Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет «Комп'ютерні технології і системи»

Кафедра «Комп'ютерні інформаційні технології»

# Пояснювальна записка
до кваліфікаційної роботи бакалавра

на тему: «Розробка мобільного додатку для перегляду розкладу занять університету»
за освітньою програмою: «12 Інженерія програмного забезпечення»
зі спеціальності: «121 Інженерія програмного забезпечення»
Виконав: студент групи «ПЗ20130»

| | | |
|---|---|---|
| | _______________ (підпис студента) | /Владислав ЗАБОЛОТНИЙ/ (Ім'я ПРІЗВИЩЕ) |
| Керівник: | _______________ (підпис) | /доц. Олександр ЖЕВАГО/ (посада, Ім'я ПРІЗВИЩЕ) |
| Нормоконтролер: | _______________ (підпис) | /доц. Світлана ВОЛКОВА/ (посада, Ім'я ПРІЗВИЩЕ) |

Засвідчую, що у цій роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _______________
(підпис)

Дніпро – 2023 рік

Ministry of Education and Science of Ukraine
Ukrainian State University of Science and Technologies

Faculty «Computer technologies and systems»

Department «Computer information technology»

# Explanatory Note

to Bachelor's Thesis

on the topic: «Development of a mobile application for viewing the university class schedule»
according to educational curriculum «Software engineering»
in the Speciality: «121 Software engineering»

| | |
|---|---|
| Done by the student of the group PZ20130: | /Vladyslav ZABOLOTNYI/ |
| Scientific Supervisor: | /Oleksandr ZHEVAHO/ |
| Normative controller: | /Svitlana VOLKOVA/ |

Dnipro – 2023

Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет: Факультет «Комп'ютерні технології і системи»
Кафедра: «Комп'ютерні інформаційні технології»
Рівень вищої освіти: бакалавр
Освітня програма: «Інженерія програмного забезпечення»
Спеціальність: «121 Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри КІТ
______ /Вадим ГОРЯЧКІН/
(підпис)

Дата ______________

**З А В Д А Н Н Я**

на кваліфікаційну роботу бакалавра
студенту Заболотному Владиславу Олександровичу
1. Тема роботи: «Розробка мобільного додатку для перегляду розкладу занять університету»
Керівник роботи: Жеваго Олександр Олександрович, доцент
затверджені наказом № 1209 ст від 07.12.2022
2. Строк подання студентом роботи: __.__.202_ р.
3. Вихідні дані до роботи: ____________________________________________

4. Зміст пояснювальної записки (перелік питань, які потрібно опрацювати):
Вступ
Збір та аналіз вимог до мобільного додатку
Проектування мобільного додатку
Розробка мобільного додатку
Розробка мобільного додатку
Тестування та відлагодження мобільного додатку
Загальні висновки
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень):
Презентація
Відео роботи програми

КАЛЕНДАРНИЙ ПЛАН

| Стадії розробки | Етапи розробки | Терміни виконання |
|---|---|---|
| 1.Технічне завдання (ТЗ) | Постановка задачі | 13.02.23 – 18.02.23 |
| | Огляд літератури та аналіз аналогів | 19.02.23 – 22.02.23 |
| | Розробка структур вхідних і вихідних даних | 23.02.23 – 04.03.23 |
| | Визначення вимог до програми. Вибір та обґрунтування мови програмування | 05.03.23 – 08.03.23 |
| | Узгодження та затвердження ТЗ | 09.03.23 – 11.03.23 |
| 2. Робочий проект | Розробка та програмування логіки програми | 12.03.23 – 02.05.23 |
| | Розробка і реалізація інтерфейсу користувача | 03.05.23 – 19.05.23 |
| | Відлагодження програми | 20.05.23 – 09.06.23 |
| | Розробка, узгодження та затвердження програмної документації | 10.06.23 – 14.06.23 |

Студент ____________________ Владислав ЗАБОЛОТНИЙ

(підпис) (Ім’я ПРІЗВИЩЕ)

Керівник роботи ____________________ доц. Олександр ЖЕВАГО

(підпис) (Ім’я ПРІЗВИЩЕ)

# РЕФЕРАТ

Пояснювальна записка складається з 8 розділів:

- вступ – у цьому розділі описується проблематика, причина та актуальність розробки. Складається з 1 сторінки;
- збір та аналіз вимог до програмного забезпечення – у даному розділі зазначений список аналогів, їх переваги та недоліки. Виділено вимоги до розроблювальної програми. Складається з 15 сторінок;
- проєктування мобільного додатку – у цьому розділі представлено функціональні характеристики, вхідні дані, вихідні дані, вибір мови програмування, парадигми програмування та проєктування екранів, інтерфейсу дизайну мобільного додатку. Складається з 13 сторінок;
- розробка мобільного додатку – у даному розділі розглянуто взаємодію користувача з програмою, ескізи екранів та опис компонентів, інтерфейс мобільного додатку. Складається з 10 сторінок;
- тестування та налагодження – у цьому розділі проведено тестування програмного додатку методами чорного та білого ящика. Проведено налагодження одного з методів програми. Складається з 21 сторінки;
- загальні висновки – висновки по ходу виконання всієї роботи.
Складається з 1 сторінки.
- список використаних джерел – бібліографіний список використаних літератури. Складається з 1 сторінки.
- додатки – даний розділ мітиться технічне завдання, текст програми, опис програми та керівництво користувача.

ЗМІСТ

# ВСТУП

Університети відіграють ключову роль у навчанні молоді та формуванні майбутніх професіоналів. Однак, з огляду на зростання числа студентів та складність розкладу занять, студентам дедалі важче відстежувати свої заняття та організовувати свій навчальний процес. Це може призводити до пропусків, невдалих планувань і, врешті-решт, погіршення академічних результатів.

Причиною виникнення розробки є вирішення цього проблемного питання та відсутність адаптивних додатків для нашого університету. Метою дипломного проєкту є розробка мобільного додатку для перегляду розкладу занять університету. Додаток буде надавати студентам та викладачам зручний та легко доступний інструмент для перегляду актуальних розкладів занять та модулів.

Основною метою проєкту є спрощення процесу отримання та оновлення інформації про розклад занять університету, забезпечуючи студентам та викладачам зручний та швидкий доступ до актуальних даних.

Для розробки додатку використані сучасні технології мобільної розробки, що дозволяють забезпечити швидку та надійну роботу програмного забезпечення.

Завдяки даному мобільному додатку, студенти та викладачі університету зможуть легко та швидко отримувати актуальну інформацію про свої заняття, уникнути пропусків та незручностей, пов'язаних з некоректним або застарілим розкладом.

Результатом розробки буде функціональний мобільний додаток, який зробить процес отримання та оновлення розкладу занять університету зручним та ефективним для всіх користувачів.

# 1 ЗБІР ТА АНАЛІЗ ВИМОГ ДО МОБІЛЬНОГО ДОДАТКУ

## 1.1 Опис та аналіз аналогів

Для пошуку аналогів використано маркет додатків Google Play Store. Пошук відбувався у розділі освіта за назвами: «Розклад занять», «Расписание», «Time table».

1. Додаток розроблений Candy Soft Inc «Мій розклад» дата випуску 23 березня 2019 р. Додаток виконує широкий спектр завдань та відображає розклади великої кількості Українських університетів.

Опис функцій:

- вибір навчального закладу зі списку доступних;
- вибір групи або викладача;
- вибір періоду часу відображення розкладу;
- відображення розкладу для груп та викладачів;
- перехід на сайт університету;
- створення заміток.

Опис інтерфейсу:

Мобільний додаток має 4 основні сторінки:

– Меню – на даній сторінці налаштовуються параметри відображення розкладу, а саме вибір університету, групи або викладача, період часу (рис 1.1 - рис 1.4);

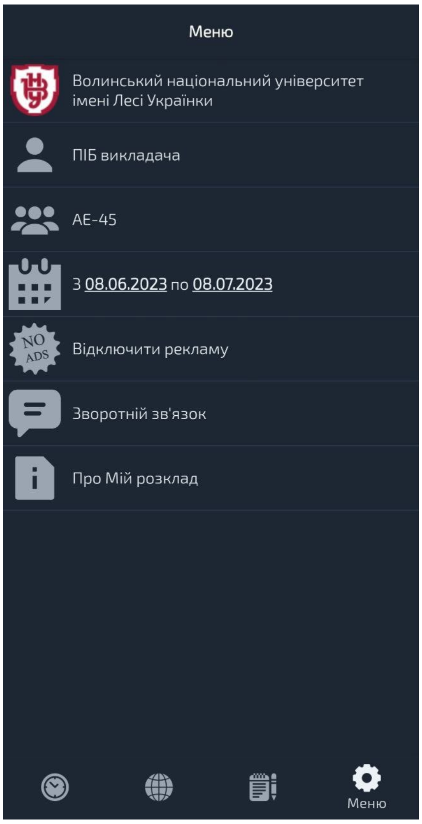



Рисунок 1.1 – Меню

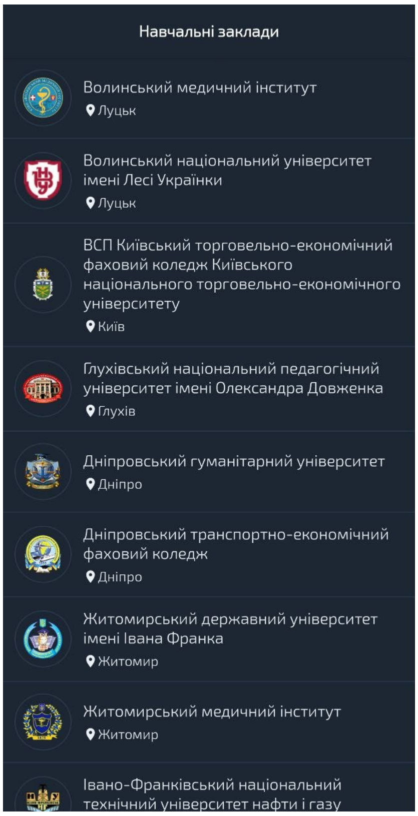



Рисунок 1.2 – Вибір навчального закладу

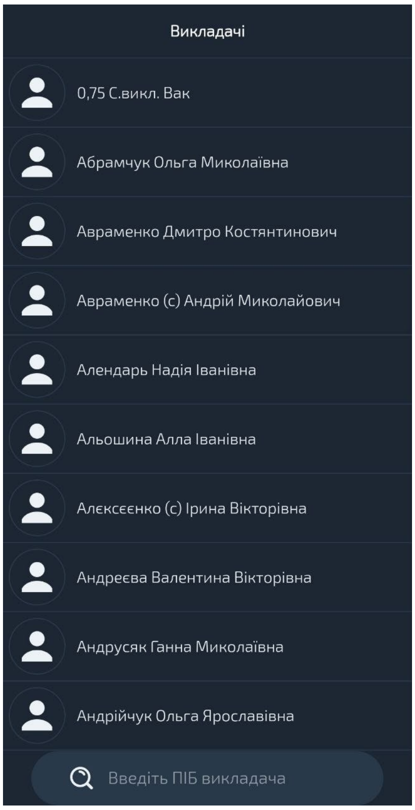



Рисунок 1.3 – Вибір викладача

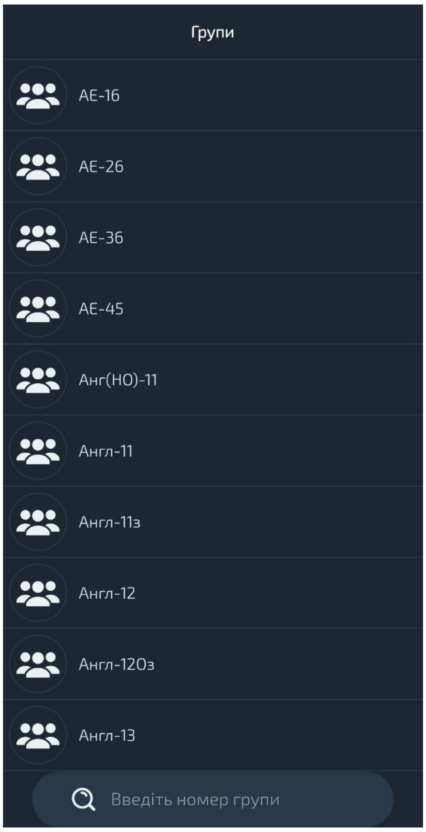



Рисунок 1.4 – Вибір групи

– Блокнот – для створення заміток (рис 1.5)

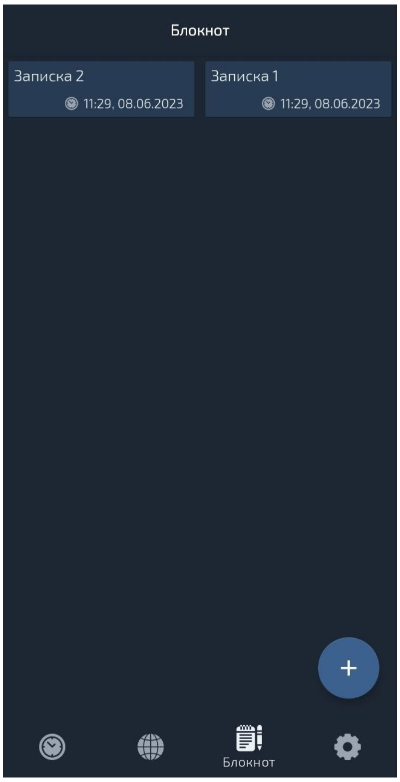



Рисунок 1.5 – Блокнот

– Сайт – сторінка відображення сайту обраного університету (рис 1.6).

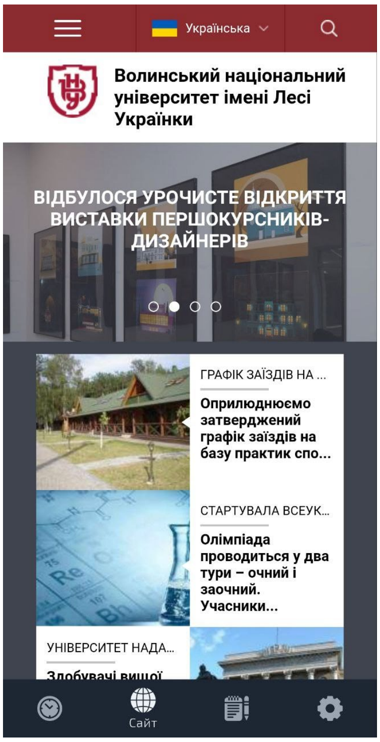



Рисунок 1.6 – Сайт

– Розклад – сторінка відображення розкладу для груп (рис 1.7) та викладачів (рис 1.8).

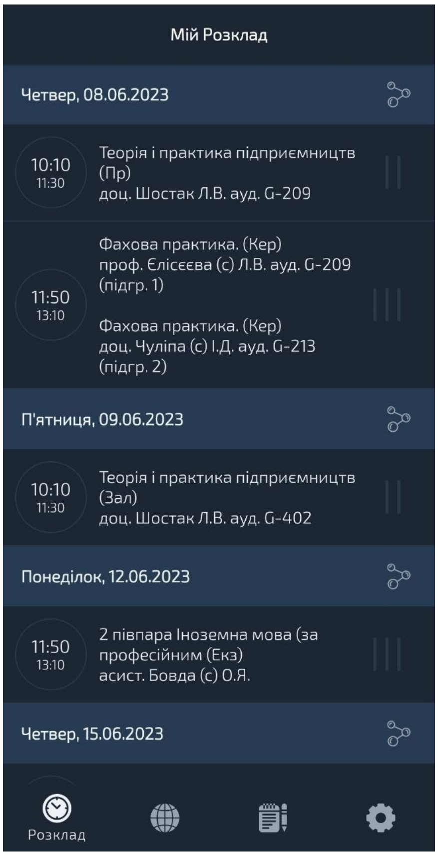



Рисунок 1.7 – Розклад для групи

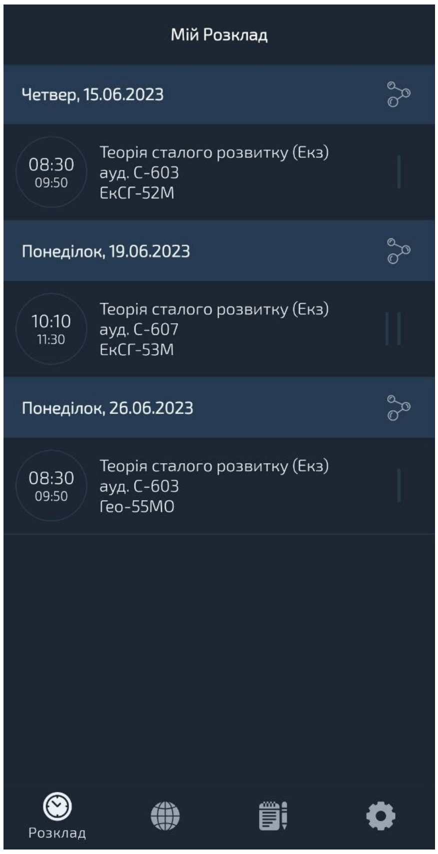



Рисунок 1.8 – Розклад для викладача

Переваги:

– вибір університетів – можна переглядати розклади великої кількості українських університетів;

– зручний інтерфейс;

– перегляд сайту обраного університету не виходячи з додатку;

– можливість створювати замітки у додатку;

Недоліки:

– перегляд тільки поточного розкладу;

– відсутність у списку університетів Українського державного університету науки і технологій;

– неможливість перегляду розкладу сесії.

2. Мобільний додаток від розробника Golden Pie Devs «Розклад КПІ» дата випуску 1 лютого 2015р. Додаток відображає розклад занять Київського політехнічного інституту.

Опис функціоналу:

– перегляд розкладу для групи та викладачів;

– перегляд викладачів до групи з рейтингом;

– перегляд розширеної інформації про заняття;

– можливість згортання додатку до меншого виду.

Опис інтерфейсу:

Програма має наступні сторінки:

– сторінка перегляду розкладу занять для групи, де здійснюється вибір календарного дня (рис 1.9) та відбувається відображення розкладу за обраним днем на тиждень (рис 1.10).

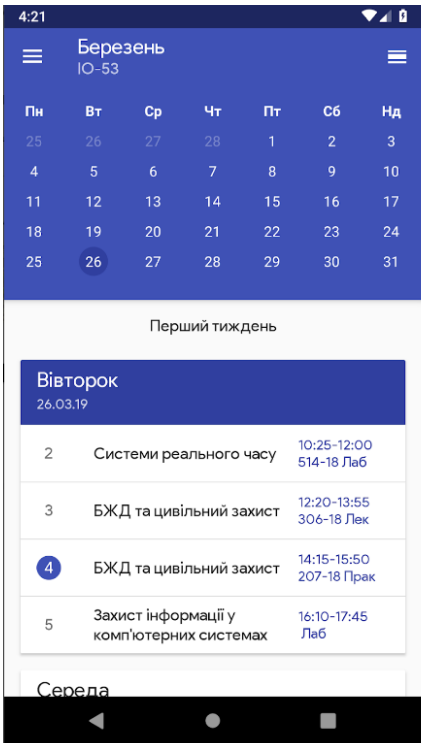



Рисунок 1.9 – Вибір календарного дня

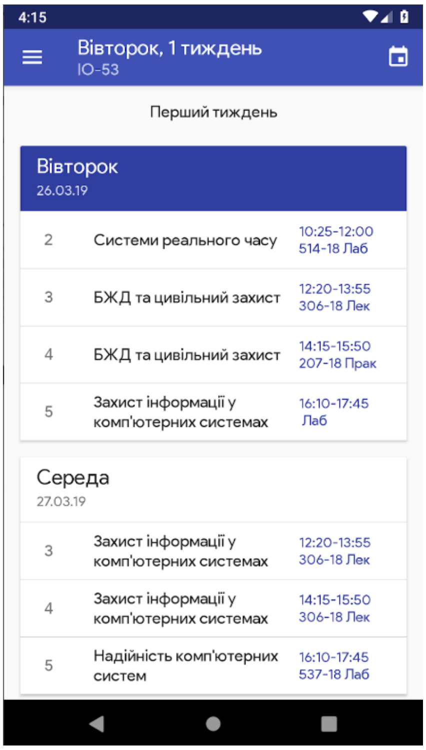



Рисунок 1.10 – Розклад занять для групи

– сторінка перегляду інформації про заняття – де відображається докладна інформація про заняття, а саме назва, час, тип та розташування з картою (рис 1.11).

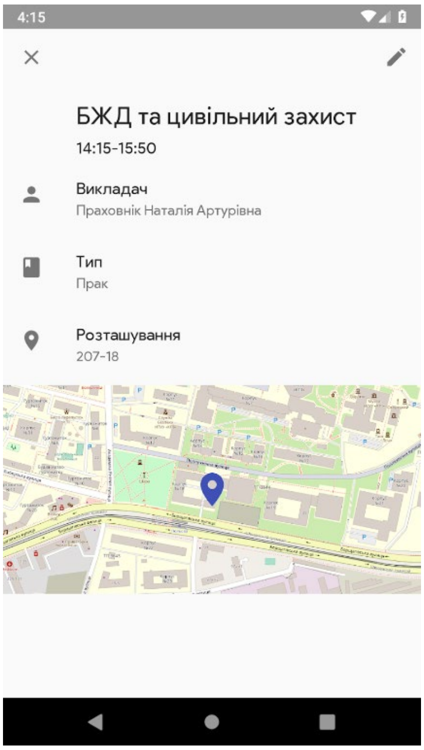



Рисунок 1.11 – Докладна інформація про заняття

– сторінка відображення викладачів з рейтингом для групи (рис 1.12).

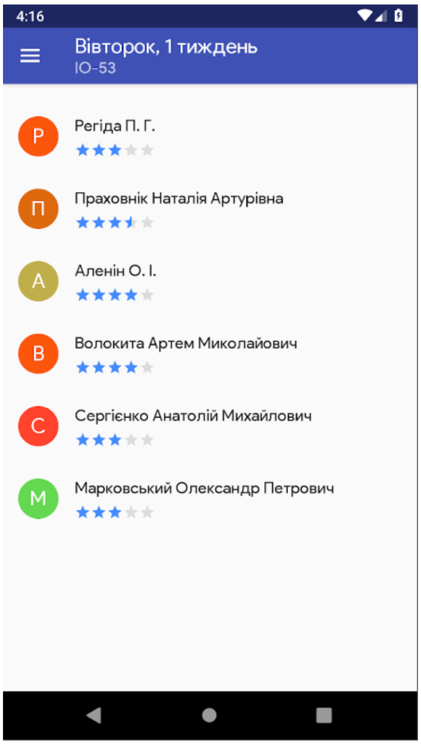



Рисунок 1.12 – Список викладачів для групи

– сторінка розкладу для викладача, на якій відображається розклад занять для обраного викладача (рис 1.13).

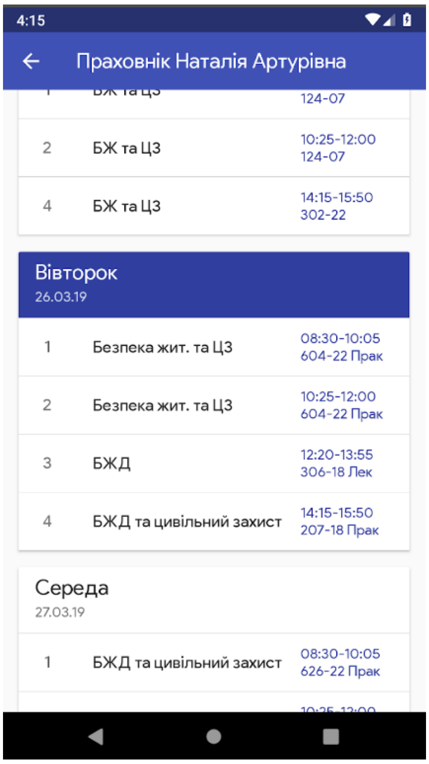



Рисунок 1.13 – Перегляд розкладу для викладача

– розклад занять у згорнутому вигляді (рис 1.14).

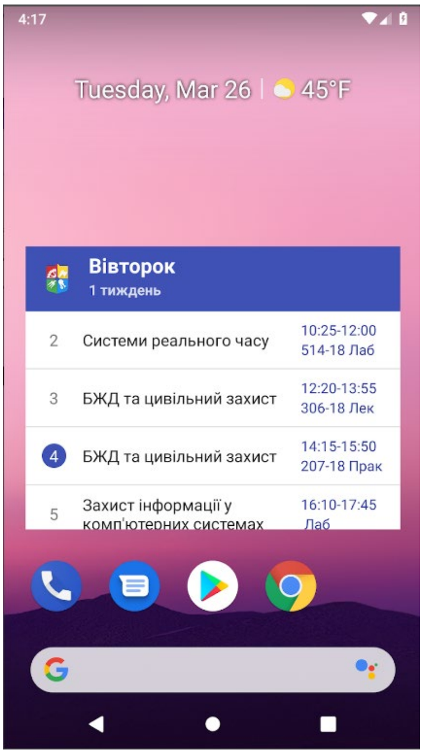



Рисунок – 1.14 Згорнутий вигляд розкладу

Переваги:

– зручний інтерфейс перегляду розкладу;

– рейтинг викладачів;

– карта з місцем знаходженням;

Недоліки:

– перегляд тільки поточного розкладу;

– неможливість перегляду розкладу сесії;

– додаток не підтримується на нових версіях операційних систем.

3. Мобільний додаток розробника Kravcov Denis «Розклад занять і Календар» дата випуску 5 серпня 2021р. Це універсальний додаток, який реалізує гнучкий варіант створення розкладу занять власноруч.

Можливості програми:

– відсутня реклама.

– декілька розкладів одночасно.

– розклад для 1, 2, 3, 4-х тижнів.

– будь-який гнучкий графік навчання.

– віджет Сьогодні з таймером занять.

– нагадування про заняття та завдання.

– фото, відео, аудіо та документи для завдань.

– простий експорт розкладу вашим друзям.

– програма доступна на Wear OS.

Опис інтерфейсу:

Програма має наступні екрани:

– головний екран розкладу, на якому відображається розклад за днем тижня та дата занять (рис 1.22)

– згорнутий екран розкладу, на якому відображається розклад, з параметрами таймеру та можливість навігації по дням (рис 1.23)

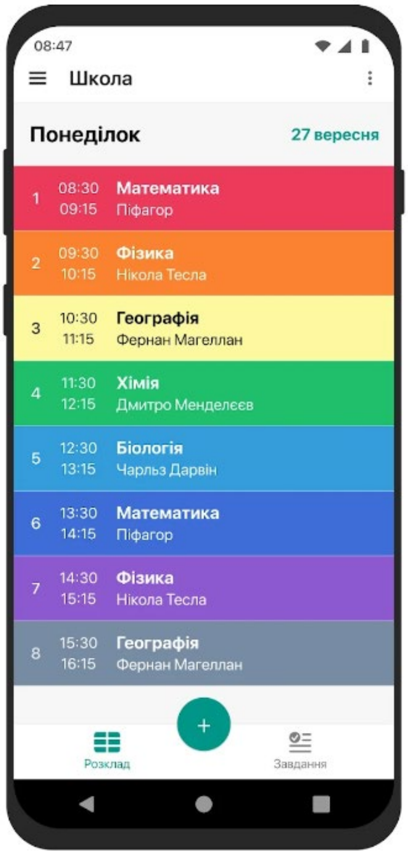



Рисунок 1.22 – Головний екран розкладу

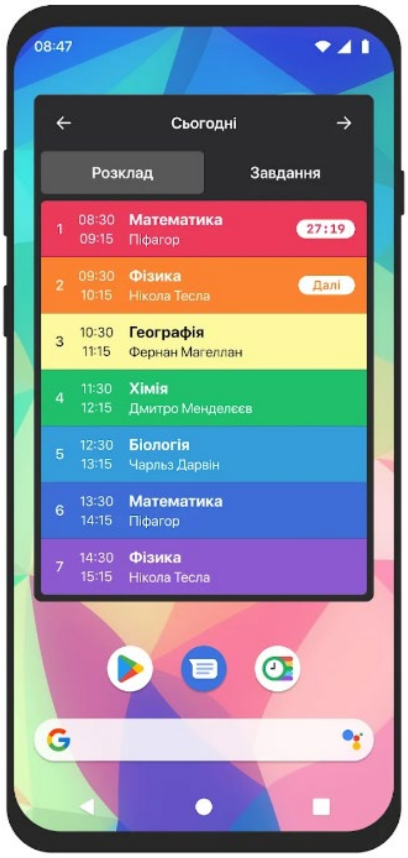



Рисунок 1.23 – Згорнутий екран розкладу

– екран вибору палітри кольорів для занять та завдань (рис 1.24)

– екран домашніх завдань на кожен день (рис 1.25)

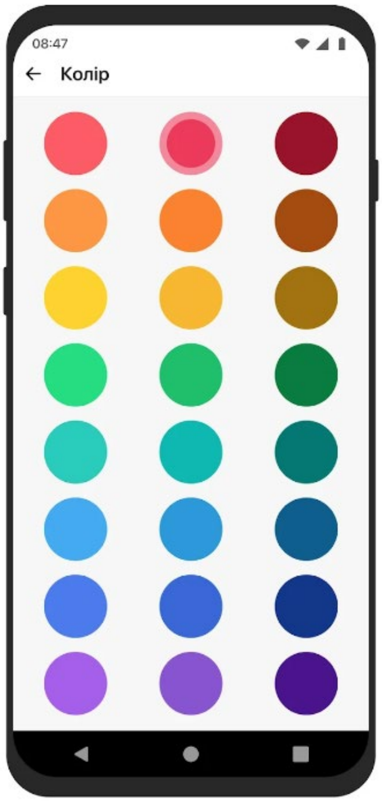



Рисунок 1.24 – Екран вибору палітри кольорів

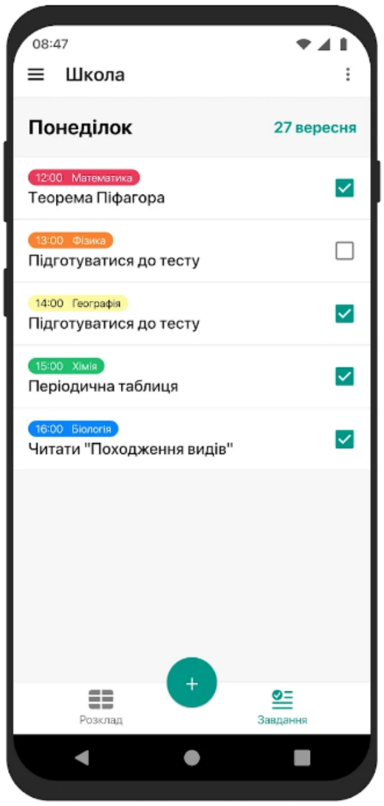



Рисунок 1.25 – Екран домашніх завдань

– екран параметрів типу розкладу, де налаштовується сам тиждень та повторюваність розкладу по тижням (рис 1.26)

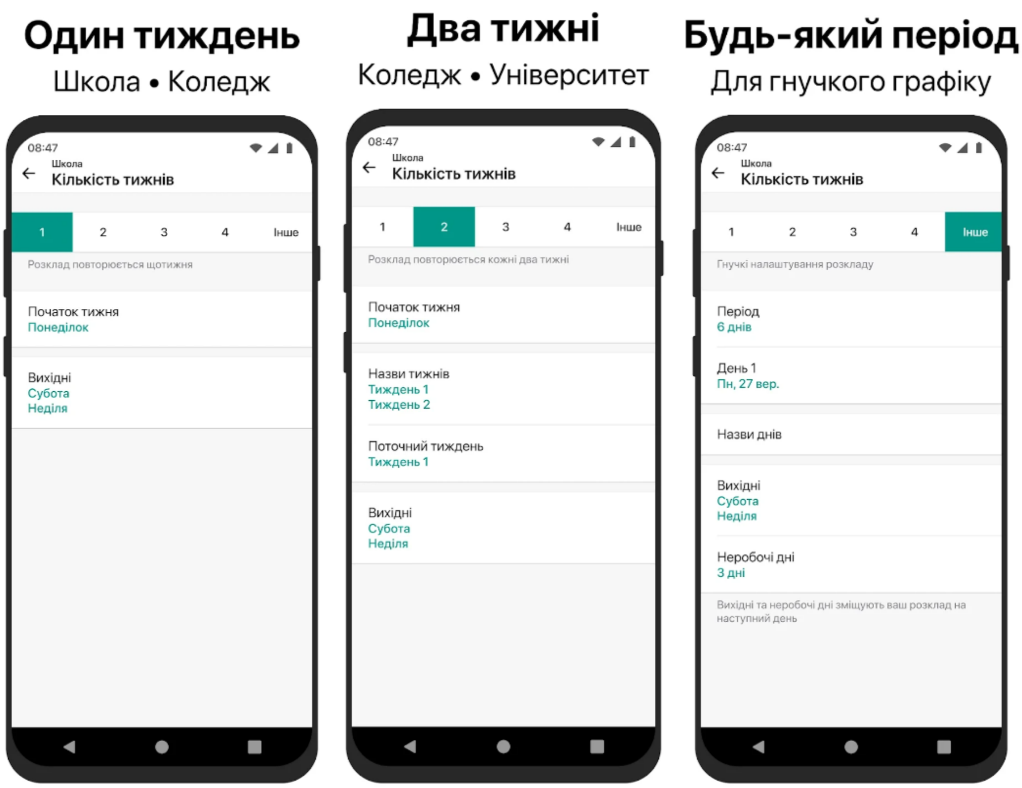



Рисунок 1.26 – Екран налаштування тижнів розкладу

– екран для надсилання розкладу, на якому можна надіслати створений розклад будь-яким зручним способом (рис 1.27)

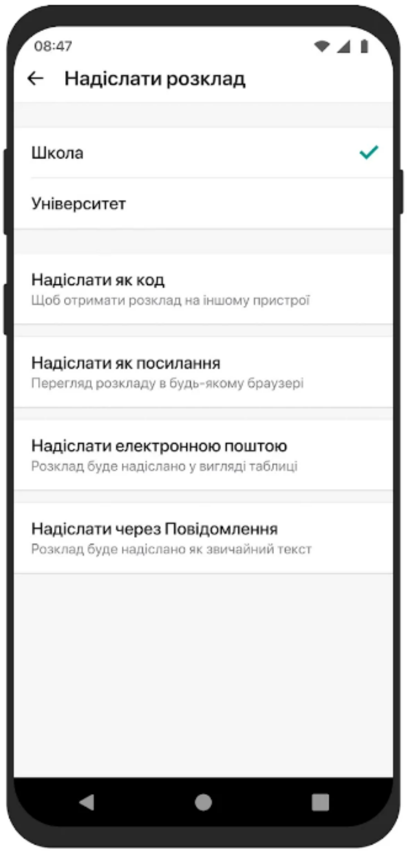



Рисунок 1.27 – Екран для надсилання розкладу

Переваги:

– зручний інтерфейс для створення та перегляду розкладу;

– можливість змінювати дизайн додатку;

– можливість отримання чи надсилання розкладу;

– гнучке налаштування розкладу.

Недоліки:

– розклади необхідно створювати вручну або отримувати розклад;

– відсутність автоматизацій отримання розкладів з освітніх установ.

## 1.2 Постановка задачі

Розробити мобільний додаток для перегляду розкладів занять та модулів Українського державного університету науки і технологій, в якому буде надаватися зручний та інтуїтивно зрозумілий інтерфейс.

Програмний продукт повинен надавати можливість:

– отримання інформації про розклади з офіційного сайту університету;
– відображення списку доступних розкладів;
– відображення обраного розкладу занять для групи;
– відображення обраного розкладу занять для викладача;
– відображення обраного розкладу модулів для групи;
– відображення обраного розкладу модулів для викладача.

Висновки по розділу:

У ході опису даного розділу було проаналізовано 3 аналоги програмного додатку і визначено їх переваги та недоліки. В результаті проведеної роботи було сформовано функціональні вимоги до розроблювального мобільного додатку.

# 2 ПРОЕКТУВАННЯ МОБІЛЬНОГО ДОДАТКУ

## 2.1 Опис функціональних характеристик

Програмний додаток призначений для перегляду доступних розкладів занять та модулів для студентів та викладачів.

Програмний продукт повинен надавати можливість надавання списку розкладів з офіційного сайту університету:

отримання інформації про розклади з офіційного сайту університету;

відображення списку доступних розкладів;

відображення обраного розкладу занять для групи;

відображення обраного розкладу занять для викладача;

відображення обраного розкладу модулів для групи;

відображення обраного розкладу модулів для викладача.

## 2.2 Вхідні дані

Вхідними даними програми, що розробляється є:

- html вміст сторінки сайту ust.edu.ua;
- файл з розкладом.

## 2.3 Вихідні дані

Результатом роботи програми є наступні вихідні дані:

- список розкладів з сайту університету;
- заняття обраного розкладу;
- сформована на основі вхідної інформації база даних (табл 2.1 – 2.2).

Таблиця 2.1 – Розклади

| Назва | Рік | Url-адреса | Контрольна сума |
|---|---|---|---|
| | | | |

Примітки:

Назва – назва розкладу, рядок.

Рік – навчальний рік розкладу, рядок.

Url-адреса – Url-адреса за якою завантажується розклад, рядок.

Контрольна сума – контрольна сума завантаженого файлу, рядок.

Таблиця 2.2 – Заняття

| Розклад | Назва | Назва групи | Альтернативна назва групи | Викладач | Аудиторія |
|---|---|---|---|---|---|
| | | | | | |

Примітки:

Розклад – назва розкладу до якого входить заняття, рядок.

Назва – назва предмету, рядок.

Назва групи – назва групи, рядок.

Альтернативна назва групи – інша назва групи, рядок.

Викладач – Прізвище та ініціали викладача, рядок.

Аудиторія – номер аудиторії, рядок.

Таблиця 2.3 – Продовження таблиці 2.2

| День тижня | Номер | Тип тижня | Тип заняття | Дата |
|---|---|---|---|---|
| | | | | |

Примітки:

День тижня – номер дня тижня, ціле число.

Номер – номер заняття, ціле число.

Тип тижня – тип тижня заняття (чисельник/знаменник/немає), рядок.

Тип заняття – тип заняття (консультація/модуль/немає), рядок.

Дата – дата та час заняття (для модулів), дата.

## 2.4 Вибір мови програмування

Для розробки мобільного додатку була обрана мова програмування C# за допомогою фреймворку Xamarin.Forms, що дозволяє створювати програми для різних операційних систем.

Xamarin дозволяє створювати одну єдину логіку програми із застосуванням C# і .NET відразу для всіх трьох платформ - Android, iOS, UWP[1].

Графічне представлення роботи Xamarin (рис 2.1)

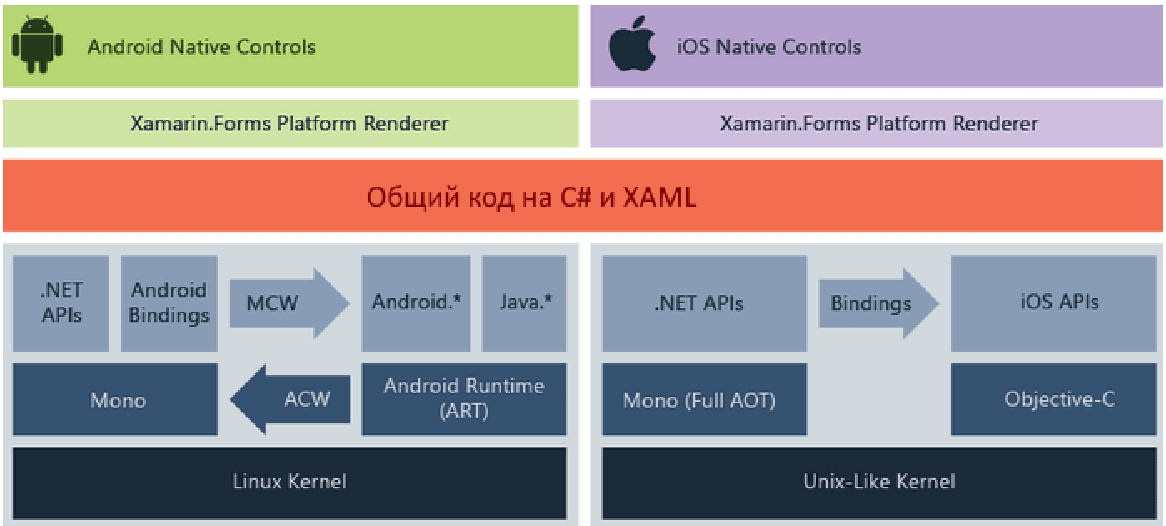



Рисунок 2.1 – Схема роботи Xamarin

Xamarin працює поверх фреймворку Mono, який надає opensource-реалізацію .NET Framework. Mono може працювати поверх різних платформ – Linix, MacOS тощо[1].

На рівні кожної окремої платформи Xamarin покладається на низку субплатформ. Зокрема:

– Xamarin.Android - бібліотеки для створення програм на ОС Android;

– Xamarin.iOS - бібліотеки для створення програм для iOS.

Ці субплатформи відіграють велику роль - через них програми можуть надавати запити до прикладних інтерфейсів на пристроях під керуванням ОС Android або iOS. Коротко це виглядає так.

За допомогою Xamarin.Android код C# з використанням Xamarin компілюється в Intermediate Language (IL), який після запуску програми компілюється в нативну збірку. Xamarin-програми запускаються в середовищі виконання Mono. Безпосередньо код не може звертатися до API Android. Для цього треба звернутися до функціональності просторів імен Android.* і Java.*, які надаються віртуальною машиною Android Runtime (ART). Спеціальний прошарок Managed Callable Wrappers (MCW) дозволяє транслювати виклики managed-коду в нативні виклики та звертатися до функціональності просторів імен Android.* та Java.*[1].

І навпаки, коли Android Runtime (ART) звертається до програми з кодом Xamarin, всі виклики проходять через обгортку Android Callable Wrappers (ACW) [1].

Програми Xamarin.iOS на відміну від Xamarin.Android, який використовує JIT-компіляцію, застосовують AOT-компіляцію (Ahead-of-Time) коду C# в нативний ARM-код. Xamarin використовує проміжний шар Selectors (селектори) для трансляції викликів коду Objective-C код на C# і шар Registrars (реєстратори) для трансляції коду C# Objective-C. У результаті шари Selectors і Registrars загалом представляють переміжний шар, який на ілюстрації вище позначений як "bindings" і який дозволяє взаємодіяти коду Objective-C з кодом C# [1].

У результаті завдяки цим платформам ми можемо створювати окремо додатки для Android, окремо для iOS, але найбільш важливою особливістю Xamarin є можливість створювати кросплатформні додатки – тобто одна логіка для всіх платформ. Дана можливість представлена технологією Xamarin.Forms і яка працює рівнем вище Xamarin.Android і Xamarin.iOS. Тобто за допомогою Xamarin.Forms ми один раз можемо визначити візуальний інтерфейс, один раз до нього прив'язати якусь логіку на C# і все це буде працювати на Android, iOS та Windows. Потім Xamarin.Forms за допомогою рендерерів (renderer) – спеціальних об'єктів для зв'язку контролів на XAML/C# з нативними контролами транслюють візуальні компоненти Xamarin.Forms у графічний інтерфейс, специфічний для кожної платформи [1].

Плюси використання Xamarin для розробки мобільних додатків

- єдиний стек технологій для розробки на всіх платформах;
- продуктивність близька до нативної;
- нативний UI;
- сумісність з обладнанням;
- Open Source-технології з корпоративною підтримкою;
- проста підтримка;

– повний пакет інструментів розробки [2].

Мінуси при розробці на Xamarin

– затримки з оновленнями платформ;
– обмежений доступ до open-source бібліотек;
– обмеженість екосистеми;
– вимога до базових знань мови програмування;
– Xamarin не підходить для додатків із високопродуктивною графікою;
– більший розмір додатків;
– складнощі з інтеграцією [2].

## 2.5 Опис об’єктно-орієнтованого підходу

Об'єктно - орієнтоване програмування (ООП) – це модель програмування яка базується на стверджені того, що програма це сукупність об’єктів які взаємодіють між собою. Кожен об’єкт в цій моделі є незалежним, і він здатний отримувати, обробляти дані та відправляти ці дані іншим об’єктам. В ООП використано моделі успадкування, модульності, поліморфізму та інкапсуляції.

Основним поняттям ООП є об'єкт. Об'єкт можна визначити як певну сукупність даних(характеристик об'єкта) та методів роботи з ними. Для класифікації об'єктів у ООП використовують класи. Клас служить зразком для створення обєкту, тобто об'єкт є нічим іншим, ніж копією класу.

Кожен об'єкт має процедури і функції(те що він уміє виконувати, наприклад, завантажувати файл, відображати картинку і т.д.), які служать для роботи з даними об'єкта. Ці процедури і функції називаються методами.

Існування ООП можливе завдяки трьом основним парадигмам на яких базується саме ООП:

- Інкапсуляція. Також відома як приховування даних. Зміст інкапсуляції полягає у приховуванні від зовнішнього користувача деталей реалізації об'єкта, замість цього надаючи інтерфейс взаємодії з ним.

- Успадкування. Це означає, що об'єкти (класи) можуть переймати деякі властивості у своїх прабатьків. Як? Це залежить від тієї мови, на якому пишеться програма. Однак у будь-якому випадку картина та ж: це призводить до повторного використання вже написаного одного разу коду. Підкласи успадковують атрибути та поведінку своїх батьківських класів, і можуть мати нові власні атрибути. Тобто утворюється ієрархія з класів, де від основного класу (так званого, предка) походять усі інші класи. Приклад такого розгалуженого "дерева" зображено на рисунку 2.2 [3].

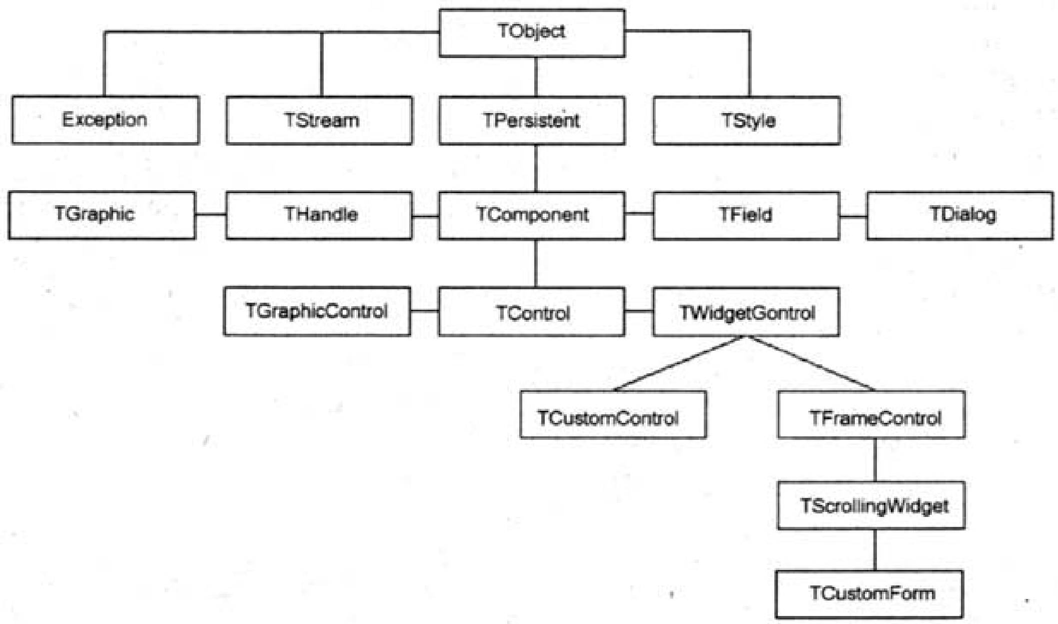



Рисунок 2.2 – Приклад дерева ієрархії класів

- Поліморфізм означає залежність поведінки від класу, в якому ця поведінка викликається, тобто, два або більше класів можуть реагувати по різному на однакові повідомлення. Це спричинене зміною в одного з класів якогось методу(процедури, функції), шляхом запису іншого алгоритму. Як приклад, деяка комп'ютерна програма при натисканні клавіші Esc завершить роботу, інша ж програма після натискання кнопки Esc тільки відкриє меню даної програми[3].

2.6 Проєктування екранів мобільного додатку

Мобільний додаток складається з двох екранів:

– екран переліку доступних розкладів;

– екран відображення обраного розкладу.

2.7 Проектування інтерфейсу користувача мобільного додатку

Для взаємодії користувача з програмою інтерфейс повинен містити наступні компоненти:

– StackLayout – компонент інтерфейсу, який необхідний для організації положення елементів на екрані;

– Frame – компонент інтерфейсу, який виконує роль групування компонентів;

– ListView – компонент інтерфейсу, який відображає список рокзладів;

– ScrollView – компонент інтерфейсу, який необхідний для гортання екрану по вертикалі;

– Picker – компонент інтерфейсу, який при натисканні на нього відображає список елементів, що дає можливість обрати один з варіантів;

– CheckBox – компонент інтерфейсу, який дає змогу обрати для яких днів тижня переглянути розклад;

– RadioButton – компонент інтерфейсу, який дає змогу обрати тип користувача, для кого відобразити розклад;

– Label – компонент інтерфейсу, який виводить текст на еркан;

– BoxView – компонент інтерфейсу, який необхідний для візуальної розмітки та створення пустого простору, у разі відсутності за чисельником або знаменником.

2.8 Проектування навігації між екранами мобільного додатку.

Навігацію між екранами у розроблювальному мобільному додатку здійснюється за допомогою класу NavigationPage. Взаємодія між екранами зображена за допомогою діаграми станів (рис 2.3).

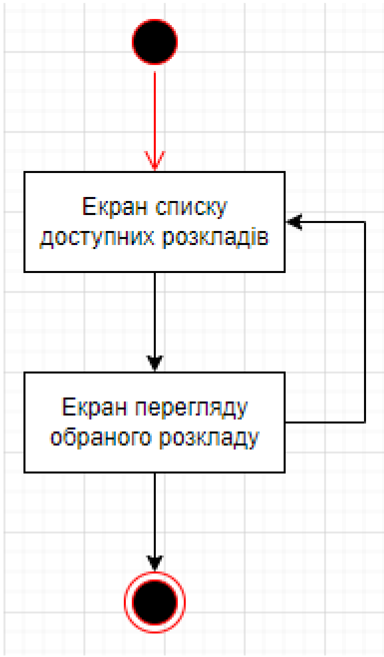



Рисунок 2.3 – Діаграма станів

## 2.9 Проектування дизайну мобільного додатку

При проектуванні дизайну мобільного додатку були розроблені ескізи для кожного компоненту програми:

– компонент типу Frame – заголовок екрану (рис 2.4)

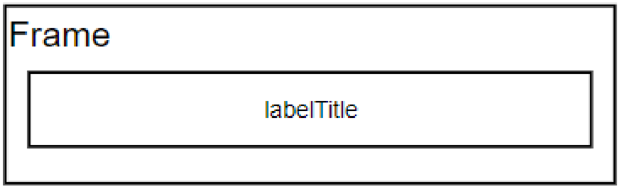



Рисунок 2.4 – Ескіз заголовку екрану розкладів

– компонент типу ListView (рис 2.5)

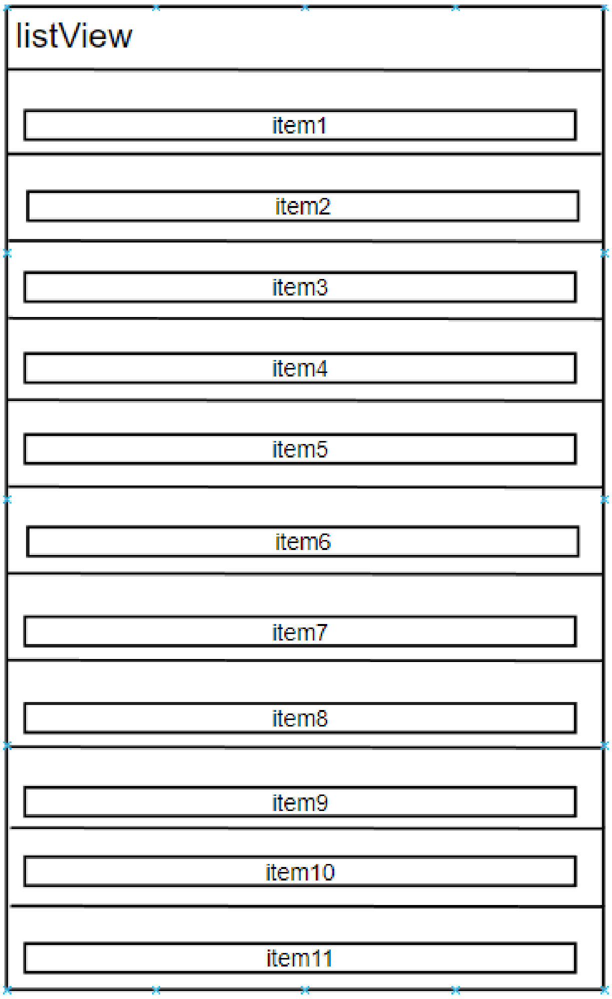



Рисунок 2.5 – Ескіз компоненту ListView

– компонент типу RadioButton (рис 2.6)


Рисунок 2.6 – Ескіз компоненту RadioButton

– компонент типу Picker (рис 2.7)


Рисунок 2.7 – Ескіз компоненту Picker

– компонент типу CheckBox (рис 2.8)

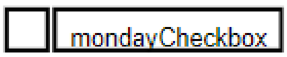



Рисунок 2.8 – Ескіз компоненту CheckBox

– компонент типу Label «Номер занять» (рис 2.9)

Рисунок 2.9 – Ескіз компоненту Label «Номер занять»

– компонент типу Label «Інформація про заняття» (рис 2.10)

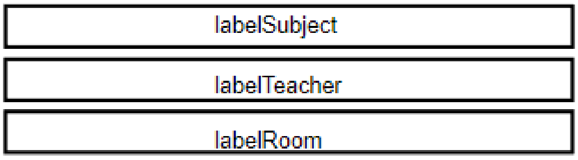



Рисунок 2.10 – Ескіз компоненту Label «Інформація про заняття»

– компонент типу StackLayout «Панель відображення розкладу» (рис 2.11)

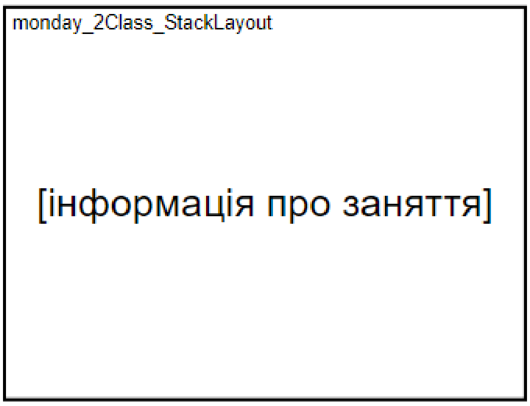



Рисунок 2.11 – Ескіз компоненту StackLayout «Панель відображення розкладу»

– компонент типу BoxView «Прямокутник розмітки» (рис 2.12)

Рисунок 2.12 – Ескіз компоненту BoxView «Прямокутник розмітки»

## 2.10 Розробка інтерфейсу користувача

Для кожного компоненту було обрано свій стиль та властивості:

– компонент типу Frame (заголовок екрану перегляду розкладів)

Програмний код опису:

```
<Frame BackgroundColor="#2196F3" Padding="24" CornerRadius="0">
    <Label Text="Оберіть розклад" HorizontalTextAlignment="Center" TextColor="White"
FontSize="32" />
</Frame>
```

Вигляд компоненту Frame зображено на рис. 2.13.

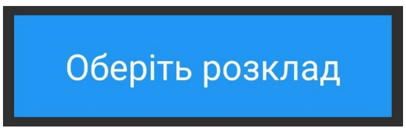


Рисунок 2.13 – Вигляд заголовку екрану перегляду розкладів

– компонент типу ListView (список розкладів)

Програмний код опису:

```
<ListView x:Name="listView"  ItemSelected="onScheduleSelected" HorizontalOptions="Center">
</ListView>
```

Вигляд компоненту ListView зображено на рис. 2.14.

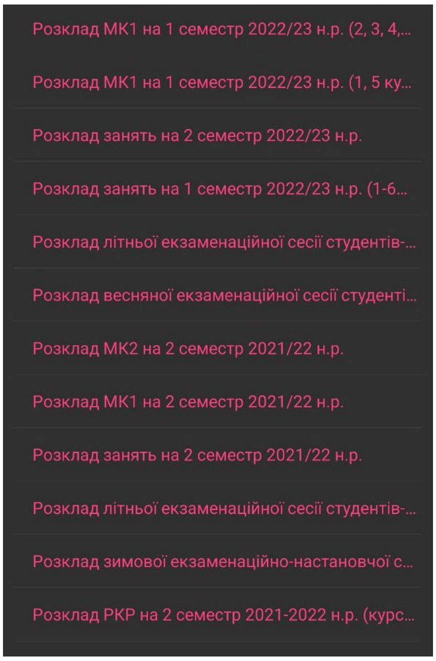



Рисунок 2.14 – Вигляд списку розкладів

– компоненти типу RadioButton (вибір критерію)

Програмний код опису:

```
<RadioButton x:Name="groupRadioButton" Content ="Студент" IsChecked="True"
CheckedChanged="onScheduleTypeChange" />
<RadioButton x:Name="teacherRadioButton" Content ="Викладач" IsChecked="False"
CheckedChanged="onScheduleTypeChange" />
```

Вигляд компонентів RadioButton зображено на рис. 2.15.


Рисунок 2.15 – Вигляд вибору критерію

– компоненти типу Picker (вибір групи/викладача)

Програмний код опису:

```
<Picker x:Name="groupPicker" SelectedIndexChanged="onSelectedGroupChange" Title="Оберіть групу"
>
      <Picker.Items>
      </Picker.Items>
</Picker>
```

```
<Picker x:Name="teacherPicker" SelectedIndexChanged="onSelectedTeacherChange" Title="Оберіть викладача" IsVisible="False">
      <Picker.Items>
      </Picker.Items>
</Picker>
```

Вигляд компонентів Picker зображено на рис. 2.16 – 2.17.


Рисунок 2.16 – Вигляд вибору групи


Рисунок 2.17 – Вигляд вибору викладача

– компоненти типу CheckBox та Label (прапорці днів тижня)

Програмний код опису:

```
<CheckBox x:Name="mondayCheckbox" CheckedChanged="onMondayCheckChange" />
<Label Text="Понеділок" FontSize="26" FontAttributes="Bold" />
   <CheckBox x:Name="tuesdayCheckbox" CheckedChanged="onTuesdayCheckChange" />
<Label Text="Вівторок" FontSize="26" FontAttributes="Bold" />
<CheckBox x:Name="wednesdayCheckbox" CheckedChanged="onWednesdayCheckChange"/>
<Label Text="Середа" FontSize="26" FontAttributes="Bold" />
   <CheckBox x:Name="thursdayCheckbox" CheckedChanged="onThursdayCheckChange" />
<Label Text="Четвер" FontSize="26" FontAttributes="Bold" />
   <CheckBox x:Name="fridayCheckbox" CheckedChanged="onFridayCheckChange" />
<Label Text="П'ятниця" FontSize="26" FontAttributes="Bold" />
```

Вигляд компонентів CheckBox та Label зображений на рис. 2.18.

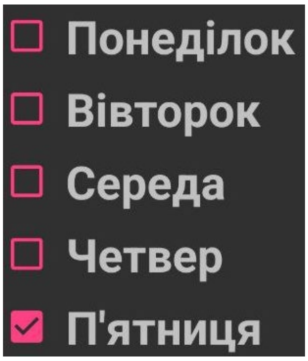



Рисунок 2.18 – Вигляд прапорців днів тижня

– компоненти типу Frame для відображення розкладу

Програмний код опису для тижня П’ятниця (для інших днів тижня застосовується аналогічний опис з відповідними іменами):

```
<Frame x:Name="fridayFrame" IsVisible="false" BackgroundColor="Gray">
   <StackLayout>
      <StackLayout Orientation="Horizontal">
         <Label Text="I" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
         <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
         <StackLayout x:Name="friday_1Class_StackLayout">
         </StackLayout>
      </StackLayout>

      <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
      <StackLayout Orientation="Horizontal">
         <Label Text="II" FontSize="22" FontAttributes="Bold" WidthRequest="25"/>
         <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
         <StackLayout x:Name="friday_2Class_StackLayout">
         </StackLayout>
      </StackLayout>

      <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
      <StackLayout Orientation="Horizontal">
         <Label Text="III" FontSize="22" FontAttributes="Bold" WidthRequest="25"/>
         <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
         <StackLayout x:Name="friday_3Class_StackLayout">
         </StackLayout>
      </StackLayout>

      <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
```

```
        <StackLayout Orientation="Horizontal">
          <Label Text="IV" FontSize="22" FontAttributes="Bold" WidthRequest="25"/>
          <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
          <StackLayout x:Name="friday_4Class_StackLayout">
          </StackLayout>
        </StackLayout>

        <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
        <StackLayout Orientation="Horizontal">
          <Label Text="V" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
          <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
          <StackLayout x:Name="friday_5Class_StackLayout">
          </StackLayout>
        </StackLayout>

        <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
        <StackLayout Orientation="Horizontal">
          <Label Text="VI" FontSize="22" FontAttributes="Bold" WidthRequest="25"/>
          <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
          <StackLayout x:Name="friday_6Class_StackLayout">
          </StackLayout>
        </StackLayout>
      </StackLayout>
    </Frame>
```

Вигляд компонентів Frame для днів тижня зображений на рис. 2.19.

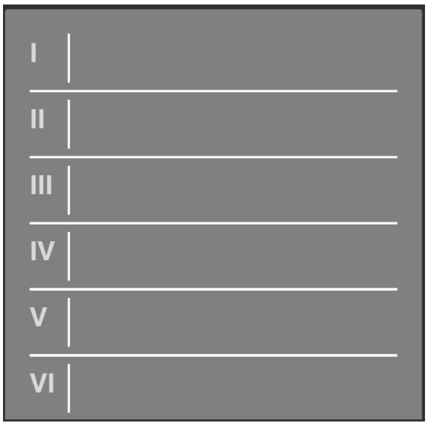



Рисунок 2.19 – Вигляд компонентів Frame для днів тижня

– компоненти типу Label для відображення інформації про заняття

Програмний код опису

```
labelSubject.FontSize = 18; labelSubject.FontAttributes = FontAttributes.Bold;
labelGroup.FontSize = 18; labelGroup.FontAttributes = FontAttributes.Italic;
labelTeacher.FontSize = 18; labelTeacher.FontAttributes = FontAttributes.Italic;
labelRoom.FontSize = 16;
labelDate.FontSize = 20; labelSubject.FontAttributes = FontAttributes.Bold;
labelSubType.FontSize = 20; labelSubject.FontAttributes = FontAttributes.Bold;
labelSeparator.Text = "--------------------------------------------";
labelSeparator.FontAttributes = FontAttributes.Bold;
```

Вигляд компонентів Label для інформації про заняття на рис. 2.20 – 2.23.

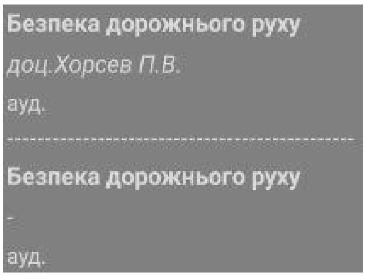



Рисунок 2.13 – Вигляд компонентів Label для розкладу занять груп

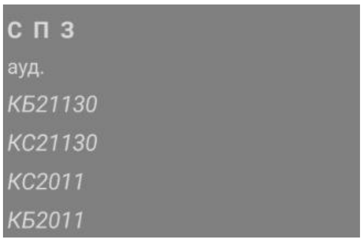



Рисунок 2.14 – Вигляд компонентів Label для розкладу занять викладачів

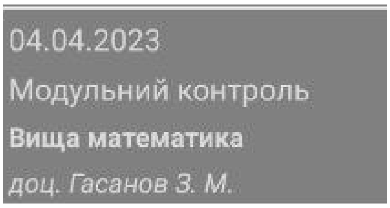



Рисунок 2.15 – Вигляд компонентів Label для модулів груп

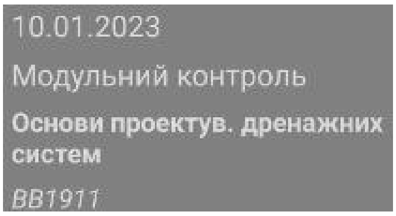



Рисунок 2.16 – Вигляд компонентів Label для модулів викладачів

Висновки по розділу:

В результаті опису аналогів, було створено план проєктування та опис функціональних вимог які супроводжується відповідним вхідним та вихідними даними.

Була обрана мова програмування та найоптимальніший підхід для кращого виконання поставленої задачі.

У процесі проєктування було спроєктовано ескізи компонентів, інтерфейс користувач, навігації між екранами та дизайн мобільного додатку.

# 3 РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ

## 3.1. Розробка динаміки системи

Для відображення динаміки системи використані діаграми діяльності (рис. 3.1) та діаграма послідовності (рис. 3.2).

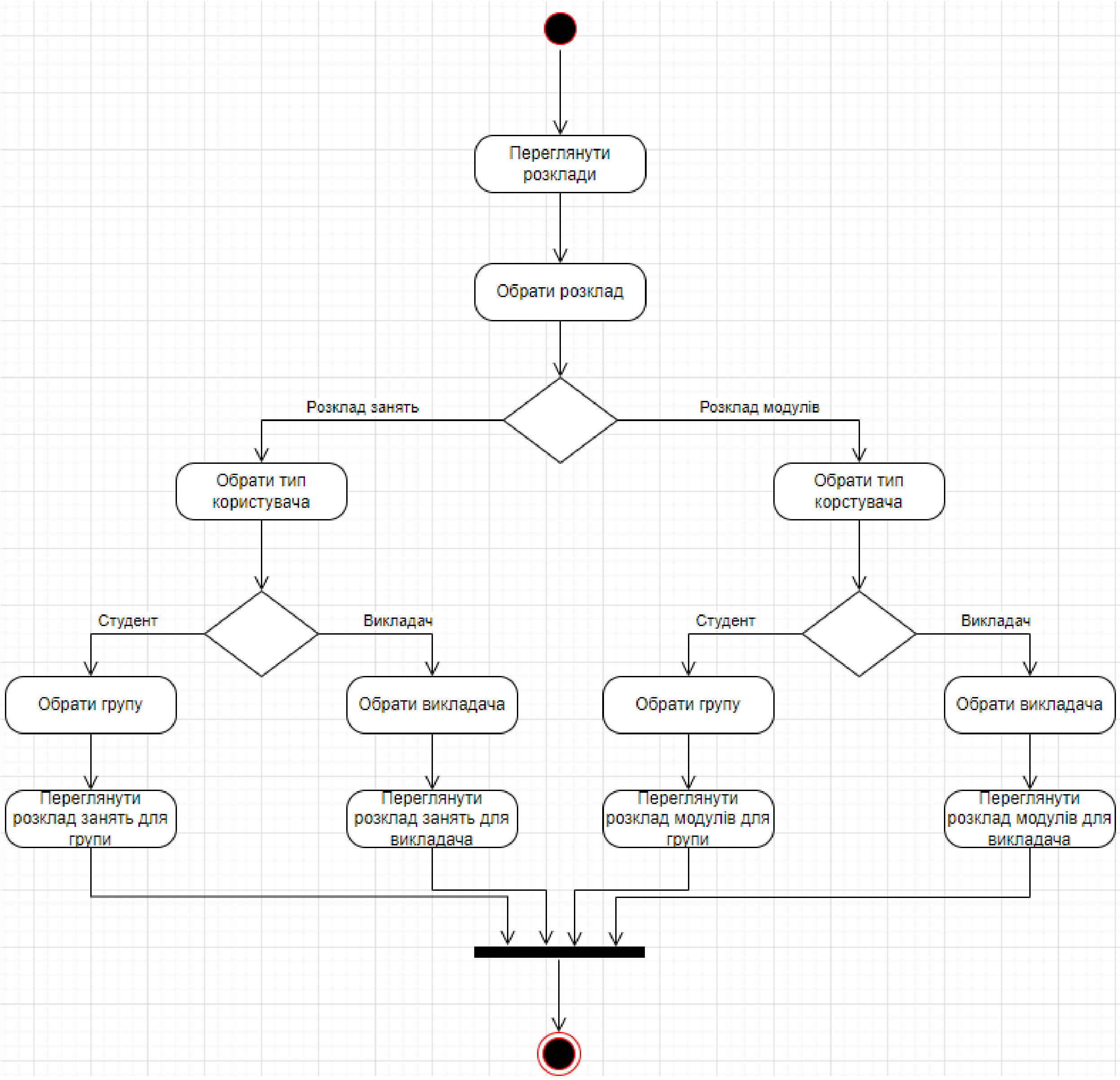



Рисунок 3.1 – Діаграма діяльності

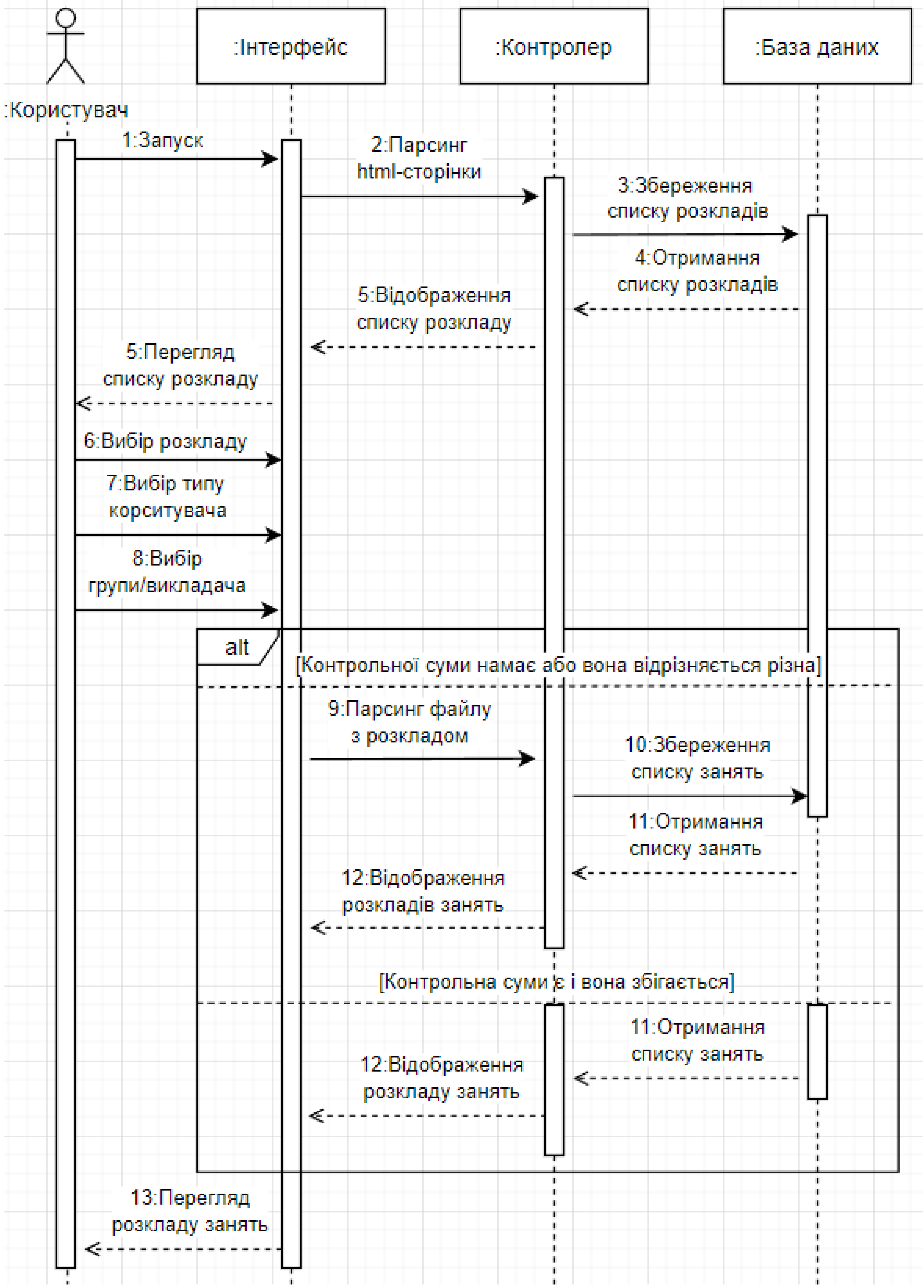



Рисунок 3.2 – Діаграма послідовності

## 3.2. Розробка екранів мобільного додатку

Мобільний додаток складається з 2 екранів:

– екран перегляду доступних розкладів;

– екран перегляду обраного розкладу.

Перелік використаних компонентів на екранах приведені в табл. 3.1 – 3.2.

Таблиця 3.1– Компоненти екрану перегляду доступних розкладів

| Компонент | Назва | Опис |
|---|---|---|
| Frame | frameTitle | Заголовок вікна |
| Label | labelTitle | Текст «Оберіть розклад» |
| ListView | listView | Список розкладів |

Таблиця 3.2 – Компоненти екрану перегляду доступних розкладів

| Компонент | Назва | Опис |
|---|---|---|
| ScrollView | | Прокрутка сторінки |
| RadioButton | groupRadioButton | Пермикач на групу |
| RadioButton | teacherRadioButton | Перемикач на викладача |
| Picker | groupPicker | Список вибору доступних груп |
| Picker | teacherPicker | Список вибору доступних викладачів |
| CheckBox | mondayCheckbox | Прапорець для відкривання розкладу на понеділок |
| Label | | Текст «Понеділок» для mondayCheckbox |
| Frame | mondayFrame | Панель для розкладу на понеділок |
| Label | | Текс «I» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | monday_1Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «II» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | monday_2Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «III» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | monday_3Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «IV» для позначення номеру занняття |
| BoxView | | Пряма лінія |

| | | |
|---|---|---|
| StackLayout | monday_4Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «V» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | monday_5Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «VI» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | monday_6Class_StackLayout | Контейнер для відображення інформації про заняття |
| CheckBox | mondayCheckbox | Прапорець для відкривання розкладу на вівторок |
| Label | | Текст «Вівторок» для tuesdayCheckbox |
| Frame | tuesdayFrame | Панель для розкладу на понеділок |
| Label | | Текс «I» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | tuesday_1Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «II» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | tuesday_2Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «III» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | tuesday_3Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «IV» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | tuesday_4Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «V» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | tuesday_5Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «VI» для позначення |

| | | |
|---|---|---|
| | | номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | tuesday_6Class_StackLayout | Контейнер для відображення інформації про заняття |
| CheckBox | wednesdayCheckbox | Прапорець для відкривання розкладу на вівторок |
| Label | | Текст «Середа» для wednesdayCheckbox |
| Frame | wednesdayFrame | Панель для розкладу на понеділок |
| Label | | Текс «I» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | wednesday_1Class_StackLayou t | Контейнер для відображення інформації про заняття |
| Label | | Текс «II» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | wednesday_2Class_StackLayou t | Контейнер для відображення інформації про заняття |
| Label | | Текс «III» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | wednesday_3Class_StackLayou t | Контейнер для відображення інформації про заняття |
| Label | | Текс «IV» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | wednesday_4Class_StackLayou t | Контейнер для відображення інформації про заняття |
| Label | | Текс «V» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | wednesday_5Class_StackLayou t | Контейнер для відображення інформації про заняття |
| Label | | Текс «VI» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | wednesday_6Class_StackLayou t | Контейнер для відображення інформації про заняття |
| CheckBox | thursdayCheckbox | Прапорець для відкривання розкладу на вівторок |
| Label | | Текст «Середа» для thursdayCheckbox |

| | | |
|---|---|---|
| Frame | thursdayFrame | Панель для розкладу на понеділок |
| Label | | Текс «I» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | thursday_1Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «II» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | thursday_2Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «III» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | thursday_3Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «IV» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | thursday_4Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «V» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | thursday_5Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «VI» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | thursday_6Class_StackLayout | Контейнер для відображення інформації про заняття |
| CheckBox | fridayCheckbox | Прапорець для відкривання розкладу на вівторок |
| Label | | Текст «Середа» для fridayCheckbox |
| Frame | thursdayFrame | Панель для розкладу на понеділок |
| Label | | Текс «I» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | friday_1Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «II» для позначення |

| | | |
|---|---|---|
| | | номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | friday_2Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «III» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | friday_3Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «IV» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | friday_4Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «V» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | friday_5Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | | Текс «VI» для позначення номеру занняття |
| BoxView | | Пряма лінія |
| StackLayout | friday_6Class_StackLayout | Контейнер для відображення інформації про заняття |
| Label | labelSubject | Назва заняття |
| Label | labelTeacher | Прізвище та ініціали викладача (для груп) |
| Label | labelGroup | Назва групи (для викладачів) |
| Label | labelRoom | Номер аудиторії |
| Label | labelSeparator | Роздільник у разі якщо заняття має тип Чисельник чи Знаменник |
| BoxView | box | Створення пустого простору якщо немає заняття по чисельнику чи знаменнику |
| Label | labelDate | Дата заняття (для розкладу модулів) |
| Label | labelSubType | Тип заняття (для розкладу модулів) |

## 3.3. Розробка інтерфейсу мобільного додатку

При розробці екранів мобільного додатку були розроблені ескізи інтерфейсу екранів (рис 3.3 – 3.4).

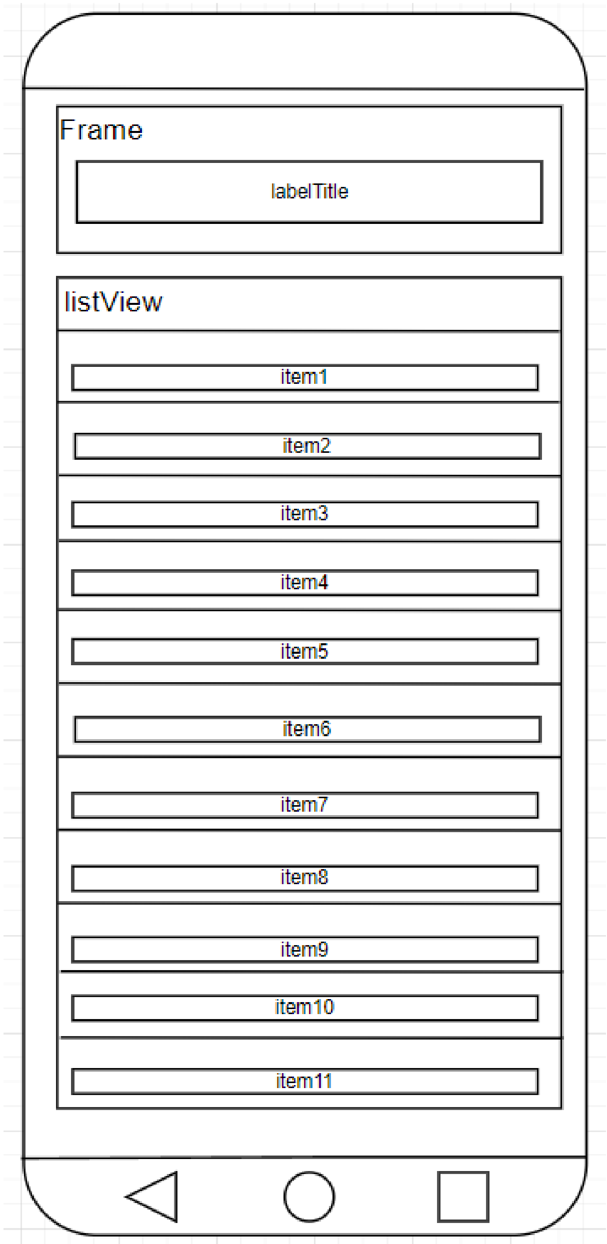



Рисунок 3.3 – Ескіз екрану перегляду доступних розкладів

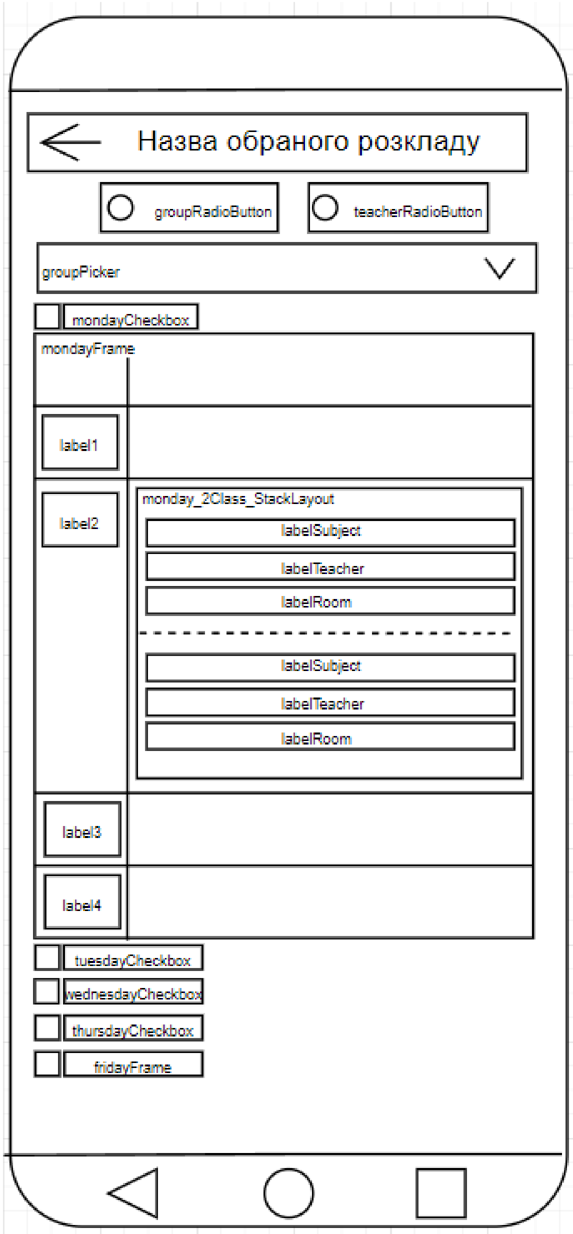



Рисунок 3.4 – Ескіз екрану перегляду обраного розкладу

## 3.4. Розробка дизайну мобільного додатку

Дизайн екранів програмного додатку зображено на рис. 3.5 – 3.6

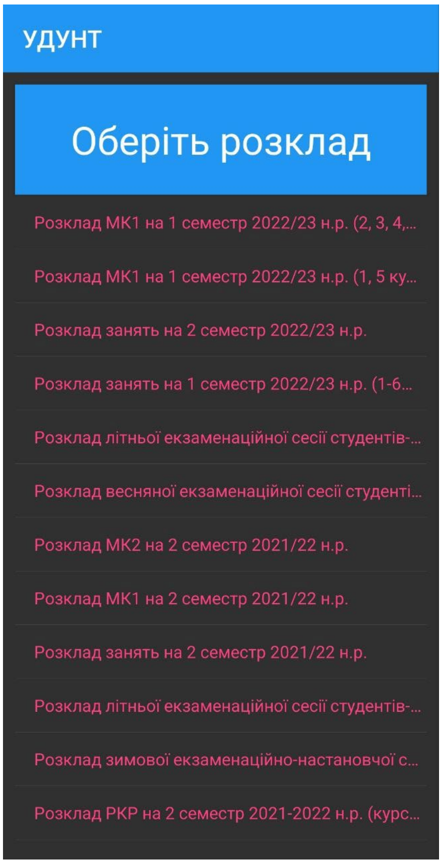



Рисунок 3.5 – Дизайн екрану перегляду обраного розкладу

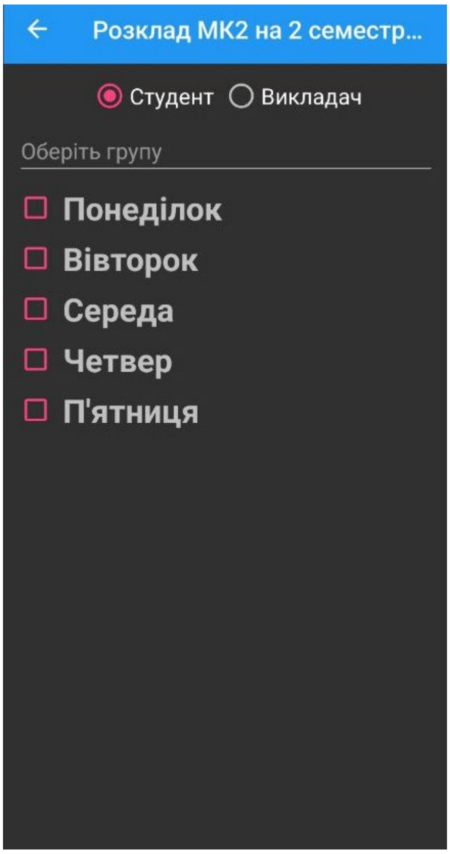



Рисунок 3.6 – Дизайн екрану перегляду обраного розкладу

Висновки по розділу:

На основі результатів проєктування було розроблено динаміку систему у вигляді діаграми діяльності та діаграми послідовності. Розроблено ескізи екранів та в результаті розроблено інтерфейс та дизайн мобільного додатку.

# 4 ТЕСТУВАННЯ ТА НАЛАГОДЖЕННЯ

## 4.1. Вибір методів тестування

Для покращення якості програмного забезпечення використовуються два основні підходи для реалізації процесу тестування: функціональний (чорний ящик) та структурний (білий ящик).

При функціональному тестуванні у тестувальників немає доступу до вихідний коду програми, і програма розглядається як «чорний ящик». Тож його суть полягає в перевірці відповідності програми своїй специфікації.

Найбільш оптимальними видами функціонального тестування є:

– випадкове (стохастичне) тестування. Створюються незалежні тести із випадково генерованих вхідних даних. Значним недоліком є велика загальна кількість тестів, які треба генерувати з огляду на надійність програми.

– тестування за класами еквівалентності. Множина вхідних і вихідних даних розбивається на класи еквівалентності. Поділ проходить таким чином, що кожен тест, що входить до певного класу, є еквівалентним будь-якому іншому тесту цього класу, тобто програма однаково реагує на всі тести одного класу.

– метод аналізу граничних умов. Перевіряються випадки, що виникають безпосередньо на межах вхідних та вихідних даних. Так як більшість помилок з'являється не в центрі множини допустимих значень, а на верхній та нижній границях, відповідно основна увага зосереджена саме на них.

Структурне тестування полягає в перевірці та аналізі вихідного коду програми.

Основними різновидами структурного тестування є:

– Тестування маршрутів. Перевірка програмного коду виконується через проходження певного зазначеного шляху. На практиці, під час тестування, часто деякі маршрути залишаються неперевіреними. Складність в тестуванні полягає в тому, що всі маршрути мають бути перевірені ще під час створення програми, що вимагає великих часових затрат.

– Тестування обробки даних. Процес виконання програми можна розуміти як роботу з певними даними, що передаються із вхідного потоку у вихідний. Із вхідних даних формуються проміжні результати, що аналізуються до моменту їх виведення. Помилки виявляються як перекручування проміжних результатів або, взагалі, їхня відсутність.

– Тестування циклів. Цикли в програмі можуть значно ускладнити весь процес тестування та відлагодження. Повне тестування повинно включати в себе перевірку всіх можливих маршрутів в кожній ітерації циклу та всіх можливих сполучень циклів із ациклічною частиною маршруту[4].

## 4.2. Тестування

### 4.2.1 Тестування методу Parse класу ClassFileParser

#### 4.2.1.1 Опис методу

Метод парсить дані .xls файлу та повертає список занять.

Вхідні дані: файл розкладу занять, тип файлу, назва обраного розкладу.

#### 4.2.1.2 Текст методу

```
public Task<ParseResult> Parse(byte[] fileData, FileType fileType, string scheduleName)
      {
         if (fileType == FileType.NotSupported || !scheduleName.Contains("Розклад занять"))//1
         {
            return Task.FromResult(ParseResult.NotSupportedFile());
         }

         var result = ParseFile(fileData, fileType);
         return Task.FromResult(result);
      }

      private ParseResult ParseFile(byte[] fileData, FileType fileType)
      {
         try
         {
            using (var ms = new MemoryStream(fileData))
            {
               var workbook = fileType == FileType.Xls ? new HSSFWorkbook(ms) : (IWorkbook)new
XSSFWorkbook(ms);
               var sheet = workbook.GetSheetAt(0);
               var columnsCount = sheet.GetRow(3).Count();
               var rowCount = sheet.LastRowNum;

               var classes = new List<Class>();

               for (int columnindex = 2; columnindex < columnsCount; columnindex++)
               {
                  var cells = new List<ICell>();
```

```
for (var rowIndex = 3; rowIndex <= rowCount; rowIndex++)
{
    var row = sheet.GetRow(rowIndex);

    if (row == null) //2
    {
        rowCount = rowIndex - 1;
        break;
    }

    var cell = row.GetCell(columnindex);
    cells.Add(cell);
}

var weekDayCounter = 1;
var numberCounter = 1;

for (var cellIndex = 0; cellIndex < cells.Count; cellIndex += rowCount)
{
    string groupName = string.Empty, groupAlternativeName = string.Empty;

    if (cells[cellIndex].IsMergedCell) //3
    {
        var stringValue = cells[cellIndex].StringCellValue;
        var groupNameParts = stringValue.Split('\n');

        groupName = groupNameParts[0];
        groupAlternativeName = groupNameParts[1];
    }
    else//4
    {
        groupName = cells[cellIndex].StringCellValue;
        groupAlternativeName = cells[cellIndex + 1].StringCellValue;
    }

    var group = new Group(groupName, groupAlternativeName);

    for (var index = cellIndex + 2; index < rowCount && index + 4 < rowCount; index += 4)
    {
        if (index + 5 > cells.Count) //5
        {
            break;
        }

        if (numberCounter > 8) //6
        {
            weekDayCounter++;
            numberCounter = 1;
        }

        if (cells[index] == null) //7
        {
            continue;
        }

        if (cells[index].IsMergedCell) //8
        {
                var teacher = !string.IsNullOrEmpty(cells[index + 3].StringCellValue) ? new
Teacher(cells[index + 3].StringCellValue) : new Teacher("-");
```

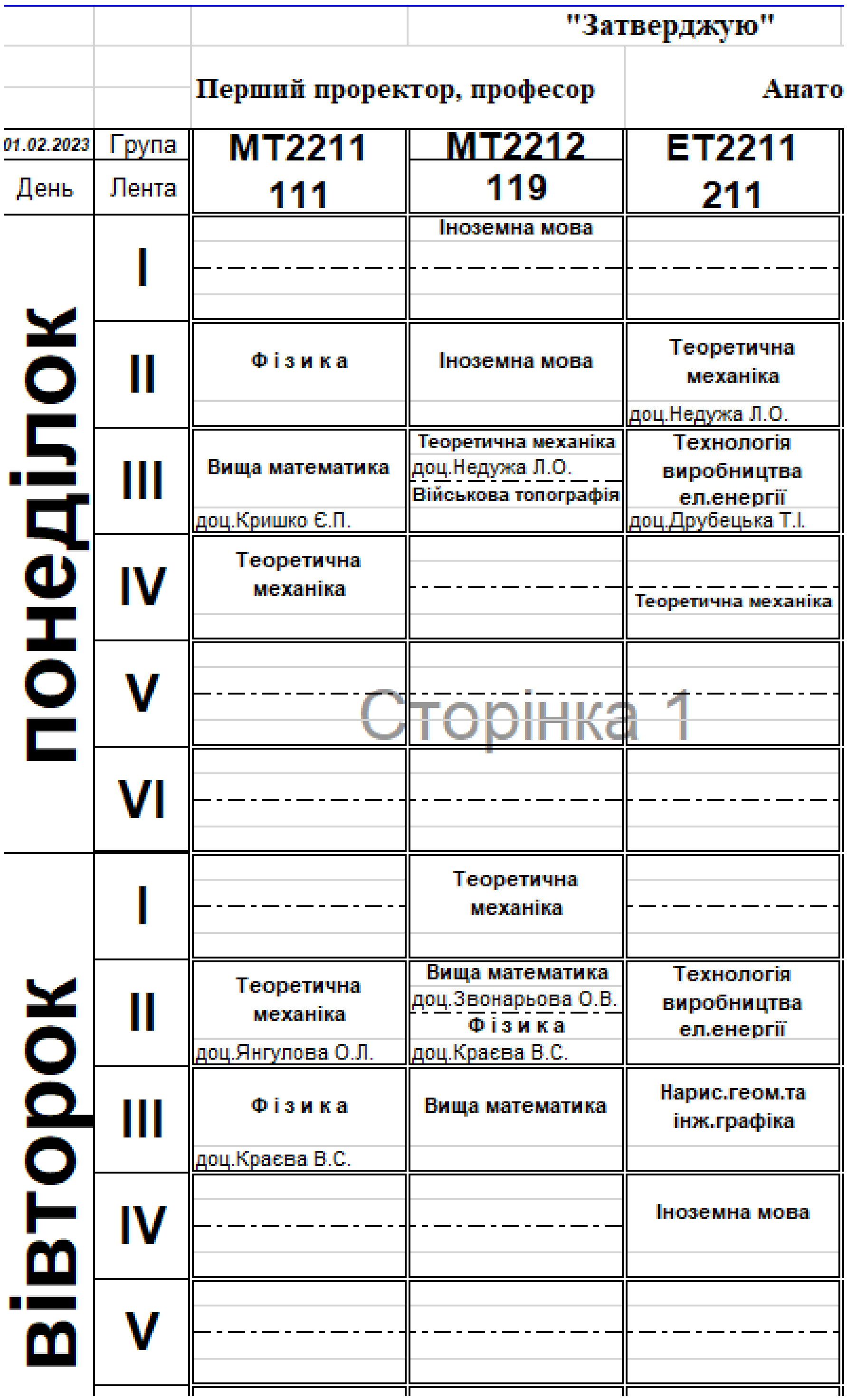
```
                        var @class = new Class(cells[index].StringCellValue, group, teacher, null,
weekDayCounter, numberCounter, WeekType.None);

                        classes.Add(@class);

                    }
                    else//9
                    {
                        var teacher = !string.IsNullOrEmpty(cells[index + 1].StringCellValue) ? new
Teacher(cells[index + 1].StringCellValue) : new Teacher("-");
                        var @class = new Class(cells[index].StringCellValue, group, teacher, null,
weekDayCounter, numberCounter, WeekType.Numerator);

                        classes.Add(@class);


                      if (cells[index + 2] == null) //10
                      {
                        continue;
                      }

                        var teacher = cells[index + 3].StringCellValue != "" ? new Teacher(cells[index +
3].StringCellValue) : new Teacher("-");
                        var @class = new Class(cells[index + 2].StringCellValue, group, teacher, null,
weekDayCounter, numberCounter, WeekType.Denominator);

                        classes.Add(@class);

                    }
                    numberCounter++;
                  }
                }
              }
              var res = new ParseResult(classes.ToArray(), null);
              return new ParseResult(classes.ToArray(), null);
            }
          }
          catch (Exception e) //11
          {
            return ParseResult.ParseFailed(e.Message);
          }
                      }
```

#### 4.2.1.3 Вибір методу тестування

Для тестування методу було обрано тестування білої скриньки, а саме методу покриття умов. Даний метод тестування краще дозволить перевірити досяжність та правильність роботи кожного оператора. Для виконання кожного оператора необхідно щоб набір тестів пройшов кожну гілку коду. Кожна гілка коду підписана коментарями з номером.

#### 4.2.1.4 Формування набору тестів

Вхідні дані набору тесту 1: файл розкладу (рис. 4.1), тип файлу: .xls, назва розкладу: «Розклад занять на 2 семестр 2022/23 н.р.»



"Затверджую"

Перший проректор, професор Анато

| 01.02.2023 День | Група Лента | МТ2211 111 | МТ2212 119 | ЕТ2211 211 |
|---|---|---|---|---|
| понеділок | I | | Іноземна мова | |
| | II | Фізика | Іноземна мова | Теоретична механіка доц.Недужа Л.О. |
| | III | Вища математика доц.Кришко Є.П. | Теоретична механіка доц.Недужа Л.О. Військова топографія | Технологія виробництва ел.енергії доц.Друбецька Т.І. |
| | IV | Теоретична механіка | | Теоретична механіка |
| | V | | | |
| | VI | | | |
| вівторок | I | | Теоретична механіка | |
| | II | Теоретична механіка доц.Янгулова О.Л. | Вища математика доц.Звонарьова О.В. Фізика доц.Краєва В.С. | Технологія виробництва ел.енергії |
| | III | Фізика доц.Краєва В.С. | Вища математика | Нарис.геом.та інж.графіка |
| | IV | | | Іноземна мова |
| | V | | | |

Сторінка 1

Рисунок 4.1 – Файл розкладу занять .xls

Вихідні дані тесту 1: список занять .

Склад списку занять:

Запис 1:

Заняття: Фізика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: -; Аудиторія: null; День тижня: 1; Номер пари: 2; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 2:

Заняття: Вища математика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Кришко Є.П.; Аудиторія: null; День тижня: 1; Номер пари: 3; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 3:

Заняття: Технічна механіка; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: -; Аудиторія: null; День тижня: 1; Номер пари: 4; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 4:

Заняття: Технічна механіка; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Янгулова О.Л.; Аудиторія: null; День тижня: 2; Номер пари: 2; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 5:

Заняття: Фізика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Краєва В.С.; Аудиторія: null; День тижня: 2; Номер пари: 3; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 6:

Заняття: Іноземна мова; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: -; Аудиторія: null; День тижня: 1; Номер пари: 1; Тип тижня: Чисельник; Тип заняття: звичайний.

Запис 7:

Заняття: Іноземна мова; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: -; Аудиторія: null; День тижня: 1; Номер пари: 2; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 8:

Заняття: Іноземна мова; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: - ; Аудиторія: null; День тижня: 1; Номер пари: 2; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 9:

Заняття: Теоретична механіка; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Недужа Л.О.; Аудиторія: null; День тижня: 1; Номер пари: 3; Тип тижня: Чисельник; Тип заняття: звичайний.

Запис 10:

Заняття: Військова топографія; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: - ; Аудиторія: null; День тижня: 1; Номер пари: 3; Тип тижня: Знаменник; Тип заняття: звичайний.

Запис 11:

Заняття: Теоретична механіка; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: - ; Аудиторія: null; День тижня: 2; Номер пари: 1; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 12:

Заняття: Вища математика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Звонарьова О.В. ; Аудиторія: null; День тижня: 2; Номер пари: 2; Тип тижня: Чисельник; Тип заняття: звичайний.

Запис 13:

Заняття: Фізика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Краєва В.С. ; Аудиторія: null; День тижня: 2; Номер пари: 2; Тип тижня: Знаменник; Тип заняття: звичайний.

Запис 14:

Заняття: Вища математика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: - ; Аудиторія: null; День тижня: 2; Номер пари: 3; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 15:

Заняття: Теоретична механіка; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: доц. Недужа Л.О.; Аудиторія: null; День тижня: 1; Номер пари: 2; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 16:

Заняття: Технологія виробництва ел. енергії; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: доц. Друбецька Т.І.; Аудиторія:

null; День тижня: 1; Номер пари: 3; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 17:

Заняття: Теоретична механіка; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: - ; Аудиторія: null; День тижня: 1; Номер пари: 4; Тип тижня: Знаменник; Тип заняття: звичайний.

Запис 18:

Заняття: Технологія виробництва ел. енергії; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: - ; Аудиторія: null; День тижня: 2; Номер пари: 2; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 19:

Заняття: Нарис. геом. та інж. графіка; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: - ; Аудиторія: null; День тижня: 2; Номер пари: 3; Тип тижня: звичайний; Тип заняття: звичайний.

Запис 20:

Заняття: Іноземна мова; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: - ; Аудиторія: null; День тижня: 2; Номер пари: 4; Тип тижня: звичайний; Тип заняття: звичайний.

Тест відпрацював вірно.

Вхідні дані набору тесту 2: файл розкладу (рис. 4.2), тип файлу: .xls, назва розкладу: «Розклад МК1 на 2 семестр 2022/23 н.р.»

"Затверджую"

Перший проректор, професор Анатолі

| МТ2211 (111) | МТ2212 (119) | ЕТ2211 (211) | ЕМ2211 (212) |
|---|---|---|---|
| 04.04.23 | 03.04.23 | 06.04.23 | 03.04.23 |
| *Консультація* | *Консультація* | *Консультація* | *Консультація* |
| Ф і з и к а | Теоретична механіка | Ф і з и к а | Загальний курс транспорту |
| доц. Краєва В. С. | доц. Недужа Л. О. | доц. Гулівець О. М. | ст.в. Черкудінов В. Е. |
| 14-30 | 13-00 | 13-00 | 13-00 |
| 06.04.23 | 04.04.23 | 10.04.23 | 04.04.23 |
| Ф і з и к а | Теоретична механіка | Ф і з и к а | Загальний курс транспорту |
| доц. Краєва В. С. | доц. Недужа Л. О. | доц. Гулівець О. М. | ст.в. Черкудінов В. Е. |
| 11-00 | 9-30 | 11-00 | 9-30 |
| 06.04.23 | 04.04.23 | 10.04.23 | 06.04.23 |
| *Консультація* | *Консультація* | *Консультація* | *Консультація* |
| Вища математика | Вища математика | Вища математика | Ф і з и к а |
| доц. Кришко Є. П. | доц. Звонарьова О. В. | доц. Мухіна Н. А. | доц. Гулівець О. М. |
| 14-30 | 13-00 | 14-00 | 13-00 |
| 10.04.23 | 06.04.23 | 12.04.23 | 10.04.23 |
| Вища математика | Вища математика | Вища математика | Ф і з и к а |
| доц. Кришко Є. П. | доц. Звонарьова О. В. | доц. Мухіна Н. А. | доц. Гулівець О. М. |
| 11-00 | 9-30 | 9-30 | 11-00 |
| 10.04.23 | 06.04.23 | | 10.04.23 |
| *Консультація* | *Консультація* | | *Консультація* |
| Інженерна геодезія | Ф і з и к а | | Вища математика |
| доц. Нікіфорова Н. А. | доц. Краєва В. С. | | доц. Мухіна Н. А. |

Рисунок 4.2 – Файл розкладу модулів .xls

Вихідні дані: пустий список.

Тест відпрацював з помилкою. Виникла помилка відповідності типу назву розкладу.

Вхідні дані набору тесту 3: файл розкладу, тип файлу: .xls, назва розкладу «Розклад МК1 на 2 семестр 2022/23 н.р.»..

Вихідні дані: пустий список.

Тест відпрацював з помилкою. Виникла помилка відкриття файлу.

У ході виконання тестів були покриті наступні умови (табл 4.1.)

Таблиця 4.1 – Таблиця покриття умов.

| Умова | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | | 7 | | 8 | | 9 | | 10 | | 11 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Тест | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - |
| 1 | | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | | * |
| 2 | * | | | | | | | | | | | | | | | | | | | | | |
| 3 | | * | | | | | | | | | | | | | | | | | | | * | |

4.2.2 Тестування методу Parse класу ModuleFileParser

4.2.2.1 Опис методу

Метод парсить дані .xls файлу та повертає список занять.

Вхідні дані: файл розкладу модулів (рис. 4.3), тип файлу, назва обраного розкладу

| | | |
|---|---|---|
| | | "Затвердж |
| Перший проректор, професор | | |
| MT2211 (111) | MT2212 (119) | ET2211 (211) |
| 04.04.23 | 03.04.23 | 06.04.23 |
| Консультація | Консультація | Консультація |
| Фізика | Теоретична механіка | Фізика |
| доц. Краєва В. С. | доц. Недужа Л. О. | доц. Гулівець О. М. |
| 8-00 | 13-00 | 13-00 |
| 06.04.23 | 04.04.23 | 10.04.23 |
| Фізика | Теоретична механіка | Фізика |
| доц. Краєва В. С. | доц. Недужа Л. О. | доц. Гулівець О. М. |
| 11-00 | 9-30 | 11-00 |
| 06.04.23 | 04.04.23 | 10.04.23 |
| Консультація | Консультація | Консультація |
| Вища математика | Вища математика | Вища математика |
| доц. Кришко Є. П. | доц. Звонарьова О. В. | доц. Мухіна Н. А. |
| 14-30 | 16-00 | 14-30 |
| 10.04.23 | 06.04.23 | 12.04.23 |
| Вища математика | Вища математика | Вища математика |
| доц. Кришко Є. П. | доц. Звонарьова О. В. | доц. Мухіна Н. А. |
| 11-00 | 9-30 | 9-30 |
| 10.04.23 | 06.04.23 | |
| Консультація | Консультація | |
| Інженерна геодезія | Фізика | |
| доц. Нікіфорова Н. А. | доц. Краєва В. С. | |
| 13-00 | 16-00 | |
| 12.04.23 | 10.04.23 | |
| Інженерна геодезія | Фізика | |
| доц. Нікіфорова Н. А. | доц. Краєва В. С. | |
| 9-30 | 9-30 | |

Рисунок 4.3 – Файл розкладу модулів .xls

#### 4.2.2.2 Текст методу

```
public Task<ParseResult> Parse(byte[] fileData, FileType fileType, string scheduleName)
    {
      if (fileType == FileType.NotSupported || !scheduleName.Contains("МК"))//1
      {
        return Task.FromResult(ParseResult.NotSupportedFile());
      }

      var result = ParseFile(fileData, fileType);
      return Task.FromResult(result);
    }

    private ParseResult ParseFile(byte[] fileData, FileType fileType)
    {
      try
      {
        using (var ms = new MemoryStream(fileData))
```

```
            {
                var workbook = fileType == FileType.Xls ? new HSSFWorkbook(ms) : (IWorkbook)new
XSSFWorkbook(ms);

                int countSheet = 2;
                var classes = new List<Class>();
                for (var sheetIndex = 0; sheetIndex < countSheet; sheetIndex++)
                {
                    var sheet = workbook.GetSheetAt(sheetIndex);

                    var columnsCount = sheet.GetRow(3).Count();
                    var rowCount = sheet.LastRowNum;



                    for (var columnIndex = 0; columnIndex < columnsCount; columnIndex += 2)
                    {
                        var cells = new List<ICell>();

                        for (var rowIndex = 3; rowIndex <= rowCount; rowIndex++)
                        {
                            var row = sheet.GetRow(rowIndex);

                            if (row == null) //2
                            {
                                rowCount = rowIndex - 1;
                                break;
                            }

                            var cell = row.GetCell(columnIndex);
                            cells.Add(cell);
                        }

                        for (var cellIndex = 0; cellIndex < cells.Count; cellIndex += rowCount)
                        {
                            var stringValue = cells[cellIndex].StringCellValue;
                            var groupNameParts = stringValue.Split('\n');
                            var group = new Group(groupNameParts[0], groupNameParts[1]);

                            for (var index = cellIndex + 1; index < rowCount && index + 5 < rowCount; index += 5)
                            {
                                if (index + 6 > cells.Count) //3
                                {
                                    break;
                                }

                                if (cells[index] != null
                                    && cells[index + 1] != null
                                    && cells[index + 2] != null
                                    && cells[index + 3] != null
                                    && cells[index + 4].StringCellValue != ""
                                    && !string.IsNullOrEmpty(cells[index + 2].StringCellValue)) //4
                                {
                                    var subject = cells[index + 2].StringCellValue;
                                    var teacher = new Teacher(cells[index + 3].StringCellValue);
                                    var date = cells[index].DateCellValue;
                                    var timeStringValue = cells[index + 4].StringCellValue;
                                    var timeParts = timeStringValue.Split('-');
                                    var fullDate = new DateTime(date.Year, date.Month, date.Day, int.Parse(timeParts[0]),
int.Parse(timeParts[1]), 0);
                                    var number = GetClassNumber(timeStringValue);
```

```
                        var subType = cells[index + 1].StringCellValue == "Консультація" ?
ClassSubType.Consultation : ClassSubType.Exam;

                        var @class = new Class(subject, group, teacher, null, (int)date.DayOfWeek, number,
WeekType.None, subType, fullDate);
                        classes.Add(@class);
                      }
                    }
                  }
                }
              }
              var res = new ParseResult(classes.ToArray(), null);
              return new ParseResult(classes.ToArray(), null);
            }
          }
          catch (Exception e) //5
          {
            return ParseResult.ParseFailed(e.Message);
          }
        }

        public static int GetClassNumber(string str)
        {
          switch (str)
          {
            case "8-00"://6
              return 1;

            case "9-30"://7
              return 2;

            case "11-00"://8
              return 3;

            case "13-00"://9
              return 4;

            case "14-30"://10
              return 5;

            case "16-00"://11
              return 6;

            default: //12
              return 0;
          }
                        }
```

#### 4.2.2.3 Вибір методу тестування

Аналогічно до попереднього методу для тестування було обрано метод покриття умов, який дозволяє перевірити досяжність та правильність роботи кожного оператора. Для виконання кожного оператора необхідно щоб набір тестів пройшов кожну гілку коду. Кожна гілка коду підписана коментарями з номером.

4.2.2.4 Формування набору тестів

Вихідні дані тесту 1: список модулів .

Склад списку модулів:

Запис 1:

Заняття: Фізика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Краєва В.С.; Аудиторія: null; День тижня: 2; Номер пари: 1; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 04.04.23.

Запис 2:

Заняття: Фізика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Краєва В.С.; Аудиторія: null; День тижня: 4; Номер пари: 3; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 06.04.23.

Запис 3:

Заняття: Вища математика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Кришко В.С.; Аудиторія: null; День тижня: 4; Номер пари: 5; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 06.04.23.

Запис 4:

Заняття: Вища математика; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Кришко В.С.; Аудиторія: null; День тижня: 1; Номер пари: 3; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 10.04.23.

Запис 5:

Заняття: Інженерна геодезія; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Нікіфорова Н.А; Аудиторія: null; День тижня: 1; Номер пари: 4; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 10.04.23.

Запис 6:

Заняття: Інженерна геодезія; Назва групи: МТ2211; Альтернативна назва групи: 111; Викладач: доц. Нікіфорова Н.А; Аудиторія: null; День тижня: 3; Номер пари: 2; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 12.04.23.

Запис 7:

Заняття: Теоретична механіка; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Недужа Л.О.; Аудиторія: null; День тижня: 1; Номер пари: 4; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 03.04.23.

Запис 8:

Заняття: Теоретична механіка; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Недужа Л.О.; Аудиторія: null; День тижня: 2; Номер пари: 2; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 04.04.23.

Запис 9:

Заняття: Вища математика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Звонарьова О. В.; Аудиторія: null; День тижня: 2; Номер пари: 6; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 04.04.23.

Запис 10:

Заняття: Вища математика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Звонарьова О. В.; Аудиторія: null; День тижня: 2; Номер пари: 6; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 06.04.23.

Запис 11:

Заняття: Фізика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Краєва В.С.; Аудиторія: null; День тижня: 4; Номер пари: 6; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 06.04.23.

Запис 12:

Заняття: Фізика; Назва групи: МТ2212; Альтернативна назва групи: 119; Викладач: доц. Краєва В.С.; Аудиторія: null; День тижня: 1; Номер пари: 2; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 10.04.23.

Запис 13:

Заняття: Фізика; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: доц. Гулівець О.М.; Аудиторія: null; День тижня: 4; Номер пари: 4; Тип тижня: звичайний; Тип заняття: Конусльтація; Дата: 06.04.23.

Запис 14:

Заняття: Фізика; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: доц. Гулівець О.М.; Аудиторія: null; День тижня: 1; Номер пари: 3; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 10.04.23.

Запис 15:

Заняття: Вища математика; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: доц. Мухіна Н.А.; Аудиторія: null; День тижня: 1; Номер пари: 5; Тип тижня: звичайний; Тип заняття: Консультація; Дата: 10.04.23.

Запис 16:

Заняття: Вища математика; Назва групи: ЕТ2211; Альтернативна назва групи: 211; Викладач: доц. Мухіна Н.А.; Аудиторія: null; День тижня: 3; Номер пари: 2; Тип тижня: звичайний; Тип заняття: Модуль; Дата: 12.04.23.

Тест відпрацював вірно.

Вхідні дані набору тесту 2: файл розкладу (рис. 4.2), тип файлу: .xls, назва розкладу: «Розклад МК1 на 2 семестр 2022/23 н.р.»

Тест відпрацював з помилкою. Виникла передбачена помилка відповідності типу назву розкладу.

Вхідні дані набору тесту 3: файл розкладу, тип файлу: .xls, назва розкладу «Розклад МК1 на 2 семестр 2022/23 н.р.»..

Вихідні дані: пустий список.

Тест відпрацював з помилкою. Виникла помилка відкриття файлу

У ході виконання тестів була створена та заповенна таблиця покриття умов (табл. 4.2)

Таблиця 4.2 – Таблиця покриття умов

| Умова | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | | 7 | | 8 | | 9 | | 10 | | 11 | | 12 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Тест | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - | + | - |
| 1 | | * | * | * | * | * | * | * | | * | * | * | * | * | * | * | * | * | * | * | * | * | | * |
| 2 | * | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | | * | | | | | | | * | | | | | | | | | | | | | | | |

4.2.3 Тестування методу ParseSiteHtml класу SiteScheduleProvider

4.2.3.1 Опис методу

У даному методі виконується запит за url-адресою https://ust.edu.ua/student/lessons_schedule, де парситься html-код сторінки з розкладами. Після чого отримані дані формуються у вигляді списку ключ значення, де ключ – назва розкладу, а значення – посилання на завантаження відповідного розкладу.

4.2.3.2 Текст методу

```
private IDictionary<string, string> ParseSiteHtml(string html)
    {
        var result = new Dictionary<string, string>();
        var doc = new HtmlAgilityPack.HtmlDocument();

        doc.LoadHtml(html);

        var nodes = doc.DocumentNode.SelectNodes(".//div[@class='col-xs-12']//div[@class='content']");

        if (nodes?.Any() != true)
        {
            return result;
        }

        foreach (var node in nodes)
        {
            var linkNodes = node.SelectNodes(".//a");

            if (linkNodes?.Any() == true)
            {
                foreach (var linkNode in linkNodes)
                {
                    var href = linkNode.GetAttributeValue("href", "");
                    if (!string.IsNullOrEmpty(href))
                    {
                        var scheduleName = linkNode.InnerText.Replace(" ", " ");
                        string scheduleUrl;
                        if (!href.StartsWith("http"))
                        {
                            if (href.Contains("////"))
                                scheduleUrl = href.Replace("////", "https://");
                            else
                                scheduleUrl = "https:" + href;
                        }
                        else
                        {
                            scheduleUrl = href;
                        }

                        if (scheduleUrl.EndsWith(".xls") || scheduleUrl.EndsWith(".xlsx"))
                        {
                            result.Add(scheduleName, scheduleUrl);
                        }
                    }
                }
            }
        }
        return result;
                }
```

4.2.3.3 Вибір методу тестування

Так як користувач ніяким чином не може вплинути на виконання даного методу, було вирішено тестувати методом припущення про помилку

4.2.3.4 Формування набору тестів

Помилка може виникнути, якщо вказана url-адреса відсутня або до неї немає доступу. Також можна припусти що список розкладів буде пустим, якщо адміністраторами була змінена структура сайту, а саме назви класів в яких знаходяться посилання на розклади.

4.3. Налагоджування програми

Під час червового запуску програми, виникло невідоме виключення. Для пошуку та усунення даної помилки необхідно провести налагодження програми.

Текст методу, що налагоджується

```
private async Task<ScheduleInfo[]> GetInfos()
    {
        try
        {
            var handler = new HttpClientHandler
            {
                AllowAutoRedirect = false,
                AutomaticDecompression = System.Net.DecompressionMethods.Deflate |
System.Net.DecompressionMethods.GZip | System.Net.DecompressionMethods.None
            };

            handler.ServerCertificateCustomValidationCallback = (sender, cert, chain, sslPolicyErrors) => { return
true; };

            using (var client = new HttpClient(handler))
            {

                var response = await client.GetAsync(siteUri);

                if (!response.IsSuccessStatusCode)
                {
                    return Array.Empty<ScheduleInfo>();
                }

                var html = await response.Content.ReadAsStringAsync();

                if (string.IsNullOrEmpty(html))
                {
                    return Array.Empty<ScheduleInfo>();
                }

                var parseResult = ParseSiteHtml(html);
                var infos = new List<ScheduleInfo>();

                foreach (var item in parseResult)
                {
```

```
                    var info = new ScheduleInfo(item.Key, GetYear(item.Key), new Uri(item.Value), null);
                    infos.Add(info);
                }

                return infos.ToArray();
            }
        }
        catch (Exception e)
        {
            return Array.Empty<ScheduleInfo>();
        }
    }
```

Для налагодження програми необхідно поставити контрольні точки (рис. 4.4) в місцях найбільш вірогідного припущення про помилку та переглянути виконання кожного оператору. Якщо станеться виключення, то програма перейде до оператору catch, де в екземплярі класу Exception і буде міститися інформація про помилку.

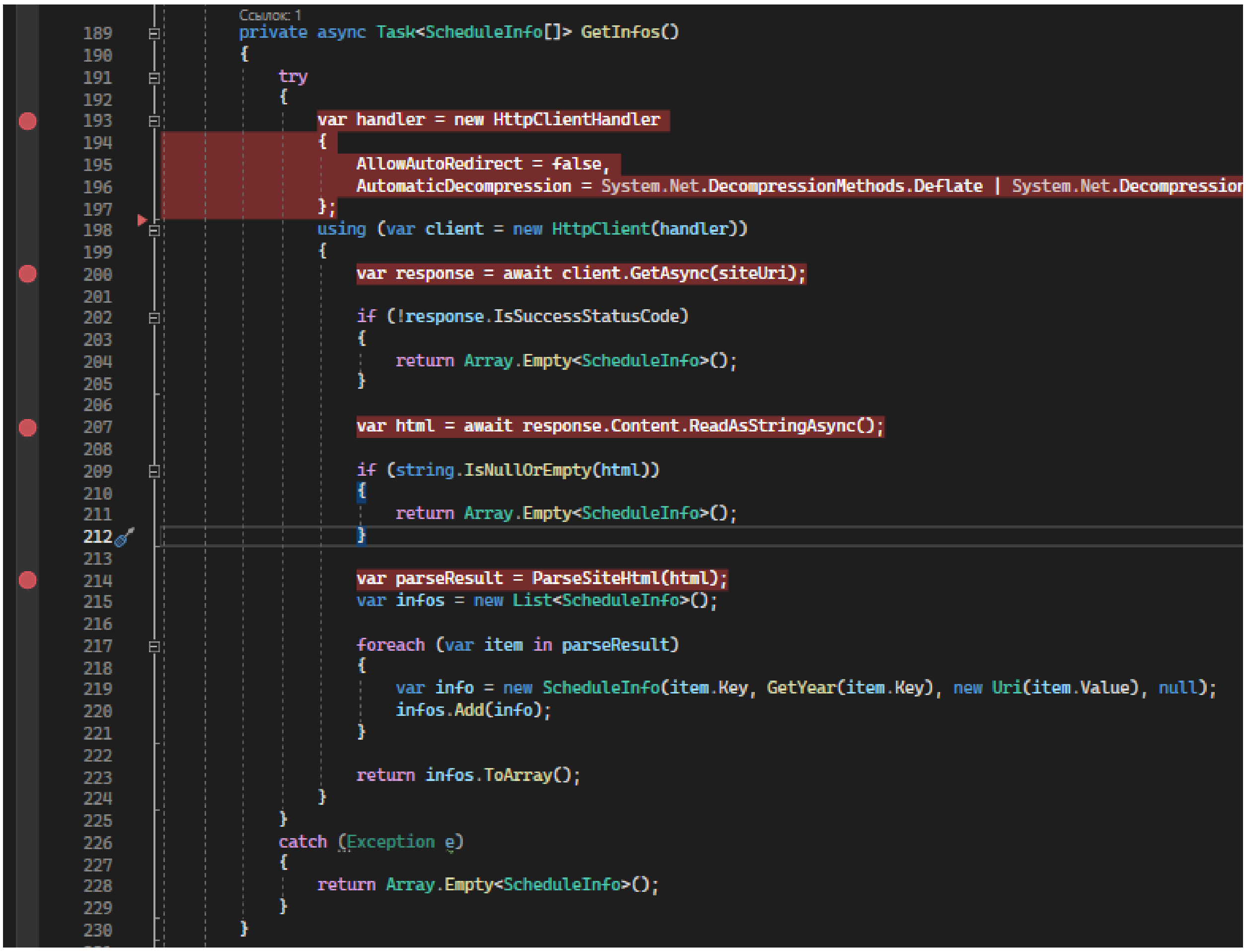



Рисунок 4.4 – Контрольні точки методу GetInfos класу SiteScheduleProvider

Виконання програми припинилося на 200 рядку та програма перейшла до оператору catch. В екземплярі класу Exception міститься наступна інформація про помилку (рис. 4.5).

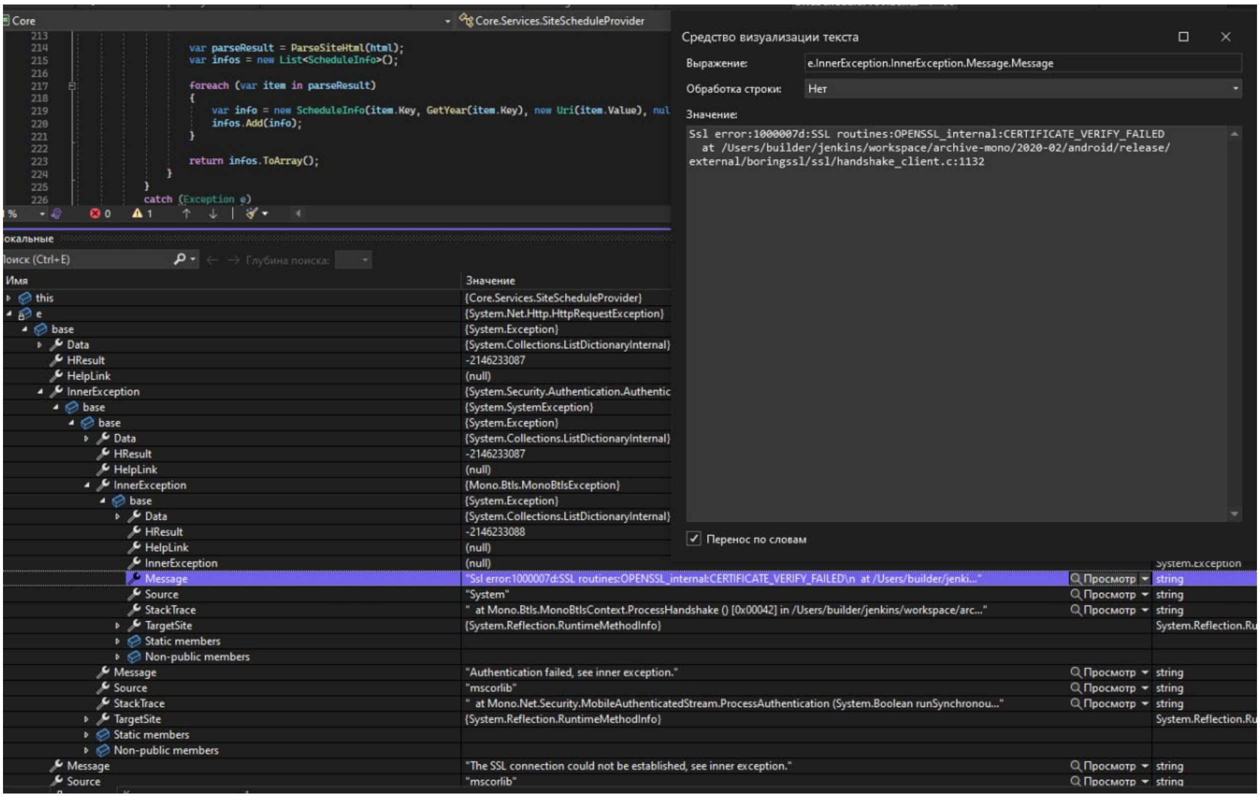



Рисунок 4.5 – Інформація про помилку

В ході пошуку інформації про причини виникнення помилки, було знайдено що дана помилка виникає в тих випадках коли програмі не вдається отримати доступ до сайту. Це трапляється у зв’язку із перевіркою SSL-сертифікату веб-сторінки, де перевірка не закінчується успіхом[5]. Після перевірки SSL-сертифікату веб-сторінки університету за допомогою. сайту thehost.ua, було виявлено що SSL-сертифікат був оновлений 04.06.2023[6].

Для вирішення перевірки було застосовано обхід перевірки SSL-сертифікату за допомогою програмного коду (рис.4.6)

```
handler.ServerCertificateCustomValidationCallback = (sender, cert, chain, sslPolicyErrors) => { return true; };
```

Рисунок 4.6 – Програмний код для обходу SSL-верифікації

У кінцевому результаті метод GetInfos класу SiteScheduleProvider має наступний вигляд:

```
private async Task<ScheduleInfo[]> GetInfos()
      {
         try
         {
            var handler = new HttpClientHandler
            {
               AllowAutoRedirect = false,
               AutomaticDecompression = System.Net.DecompressionMethods.Deflate |
System.Net.DecompressionMethods.GZip | System.Net.DecompressionMethods.None
            };

            handler.ServerCertificateCustomValidationCallback = (sender, cert, chain, sslPolicyErrors) => { return
true; };

            using (var client = new HttpClient(handler))
            {

               var response = await client.GetAsync(siteUri);

               if (!response.IsSuccessStatusCode)
               {
                  return Array.Empty<ScheduleInfo>();
               }

               var html = await response.Content.ReadAsStringAsync();

               if (string.IsNullOrEmpty(html))
               {
                  return Array.Empty<ScheduleInfo>();
               }

               var parseResult = ParseSiteHtml(html);
               var infos = new List<ScheduleInfo>();

               foreach (var item in parseResult)
               {
                  var info = new ScheduleInfo(item.Key, GetYear(item.Key), new Uri(item.Value), null);
                  infos.Add(info);
               }

               return infos.ToArray();
            }
         }
         catch (Exception e)
         {
            return Array.Empty<ScheduleInfo>();
         }
      }
```

Висновки по розділу:

Під час проходження етапу тестування було протестовано 3 основних методи програми, а саме алгоритм парсингу занять, модулів та алгоритм отримання списку розкладів з сайту університету.

В ході тестування останнього методу виникла невідома помилка. Помилка була знайдена, ідентифікована та виправлена в ході виконання процесу налагодження.

# ВИСНОВКИ

У результаті виконання проєкту було створено та розроблено мобільний додаток на мові C# з використанням платформи Xamarin, який призначений для перегляду розкладу занять університету.

У ході огляду аналогів було проведено аналіз їх переваг та недоліків та висунуто вимоги до розроблювального програмного додатку.

На етапі проектування було зазначено функціональні характеристики програми та описано вхідні і вихідні дані. Було обрано мову та парадигму програмування. Спроектовано екрани, інтерфейси та дизайн мобільного додатку.

У ході розробки мобільного додатку була описана динаміка системи та розроблено інтерфейс програми.

На останньому етапі було протестовано програмний додаток методами білого та чорного ящика.

У ході розробки програмного додатку змінився SSL-сертифікат офіційного сайту університету https://ust.edu.ua/, що посприяло виникненню труднощів до отримання списку розкладів. Метод у якому було описано отримання списку розкладів із сайту було налагоджено.

Результатом є  програмний додаток який виконує наступні функції

- перегляд списку доступних розкладів;
- перегляд розкладу занять для групи;
- перегляд розкладу занять для викладача;
- перегляд розкладу модулів для групи;
- перегляд розкладу модулів для викладача.

## ВИКОРИСТНІ ДЖЕРЕЛА

1 Інтернет-ресурс Metanit «Вступ до Xamarin. Встановлення» [Електронний ресурс] – 07.01.2021 – Режим доступу:https://metanit.com/sharp/xamarin/1.1.php

2 Стаття з Web-сайту Altexsoft «The Good and The Bad of Xamarin Mobile Developmenthttps» [Virtual Resource] – 13 nov. 2021 – Access mode: //www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/

3 Стаття з Web-сайту Програмування по-українськи «Об’єктно-орієнтоване програмування» [Електронний ресурс] - 10.01.2011 – Режим доступу: http://programming.in.ua/programming/basisprogramming/25-oop.html

4 Кучер В.В. Основні методи тестування програмного забезпечення [Електронний ресурс] – 24.06.2016 – Режим доступу: https://conf.ztu.edu.ua/wp-content/uploads/2016/06/248.pdf

5 Інтернет-ресурс StackOverflow «The SSL connection could not be established» [Електронний ресурс] – 15.12.2018 – Режим доступу: https://stackoverflow.com/questions/52939211/the-ssl-connection-could-not-be-established

6 Інтернет-ресурс TheHost «Сервіс перевірки SSL-сертифікатів» [Електронний ресурс] – Режим доступу: https://thehost.ua/ua/ssl/checkup /ust.edu.ua

ДОДАТОК А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАТВЕРДЖУЮ
Проректор
Українського державного університету
науки і технологій
Анатолій РАДКЕВИЧ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Технічне завдання
ЛИСТ ЗАТВЕРДЖЕННЯ
1116130.01318-01-ЛЗ

Завідувач кафедри КІТ
___________ Вадим ГОРЯЧКІН
Керівник розробки
___________ Олександр ЖЕВАГО
Виконавець
___________ Владислав ЗАБОЛОТНИЙ
Нормоконтролер
___________ Світлана ВОЛКОВА

2023

ЗАТВЕРДЖЕНО
1116130.01318-01-ЛЗ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Технічне завдання
1116130.01318-01

Листів 11

2023

ЗМІСТ

# 1 ВВЕДЕННЯ

Університети відіграють ключову роль у навчанні молоді та формуванні майбутніх професіоналів. Однак, з огляду на зростання числа студентів та складність розкладу занять, студентам дедалі важче відстежувати свої заняття та організовувати свій навчальний процес. Це може призводити до пропусків, невдалих планувань і, врешті-решт, погіршення академічних результатів.

Причиною виникнення розробки є вирішення цього проблемного питання та відсутність адаптивних додатків для нашого університету. Метою дипломного проєкту є розробка мобільного додатку для перегляду розкладу занять університету. Додаток буде надавати студентам та викладачам зручний та легко доступний інструмент для перегляду актуальних розкладів занять та модулів.

Основною метою проєкту є спрощення процесу отримання та оновлення інформації про розклад занять університету, забезпечуючи студентам та викладачам зручний та швидкий доступ до актуальних даних.

Для розробки додатку будуть використані сучасні технології мобільної розробки, що дозволить забезпечити швидку та надійну роботу програмного забезпечення.

Завдяки даному мобільному додатку, студенти та викладачі університету зможуть легко та швидко отримувати актуальну інформацію про свої заняття, уникнути пропусків та незручностей, пов'язаних з некоректним або застарілим розкладом.

Результатом розробки буде функціональний мобільний додаток, який зробить процес отримання та оновлення розкладу занять університету зручним та ефективним для всіх користувачів.

## 2 ПІДСТАВИ ДЛЯ РОЗРОБКИ

Підставою для розробки є наказ № 1209ст. «Про призначення керівників та затвердження тем бакалаврських робіт» за спеціальністю 121 «Інженерія програмного забезпечення» факультету «Комп’ютерних технологій і систем» по кафедрі «Комп’ютерні інформаційні технології» від 07.12.2022р. затверджений виконуючим обов’язки ректора Українського державного університету науки та технологій.

Тема проєкту «Розробка мобільного додатку для перегляду розкладу університету», керівник доц. кафедри «Комп’ютерні інформаційні технології» Жеваго О.О.

# 3 ПРИЗНАЧЕННЯ РОЗРОБКИ

Функціональне призначення - програмний додаток призначений для перегляду доступних розкладів занять та модулів для студентів та викладачів.

Експлуатаційне призначення програмного продукту:

– зручний доступ до розкладів;
– актуальність і точність;
– синхронізація з офіційним сайтом університету.

# 4 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

## 4.1 Вимоги до функціональних характеристик

Програмний продукт повинен надавати можливість:

- отримання інформації про розклади з офіційного сайту університету;
- відображення списку доступних розкладів;
- відображення обраного розкладу занять для групи;
- відображення обраного розкладу занять для викладача;
- відображення обраного розкладу модулів для групи;
- відображення обраного розкладу модулів для викладача.

Вхідними даними програми, що розробляється є:

- html вміст сторінки сайту ust.edu.ua;
- файл з розкладом.

Результатом роботи програми є наступні вихідні дані:

- список розкладів з сайту університету;
- заняття обраного розкладу;
- сформована на основі вхідної інформації база даних.

## 4.2 Вимоги до надійності

Вимоги до надійності наступні:

- програмний продукт має забезпечити стійку роботу, коректне виконання своїх основних функцій та цілісність і збереженість даних;
- кількість помилок не повинна перевищувати однієї на 1000 операторів;
- наявність архівної копії тексту програми на зовнішньому носії;
- наявність резервної копії бази даних на зовнішньому носії.

## 4.3 Умови експлуатації

Умови експлуатації повинні включати наступне:

Програмний продукт повинен використовуватись у приміщеннях, які відповідають умовам роботи ЕОМ, а саме мають такі кліматичні, санітарні та гігієнічні умови, які відповідають ДНАОП 0.00-1.31-9.

Працювати з програмою може людина, що має навички роботи з мобільними пристроями та ознайомлена з керівництвом користувача програмного продукту.

4.4 Вимоги до складу і параметрів технічних засобів

Продукт, що розробляється повинен використовуватись на мобільних пристроях, що мають наступні характеристики:

- операційна система: не нижче Android 6.0, iOS 9;
- microUSB або TypeC порт;
- оперативна пам'ять (RAM): 2 ГБ або більше;
- дисплей: HD (1280x720 пікселів) або більше;
- діагональ екрану 5,5 дюймів або більше;
- вбудована пам’ять: 16 ГБ або більше;
- підключення до Інтернету через Wi-Fi або мобільну мережу.

4.5 Вимоги до інформаційної і програмної сумісності

Програмний продукт розробляється для всіх видів операційних систем сімейства “ANDROID” починаючи від версії 6 та наступні версії, а також iOS 9.

4.6 Вимоги до маркування і упаковки

Упаковка програмного продукту, включаючи документацію повинна бути захищена від пошкоджень різного роду (механічних, кліматичних).

Упаковка повинна мати маркування:

МОБІЛЬНИЙ ДОДАТОК
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ
Розробник: Заболотний В.
УДУНТ, кафедра КІТ
2023

4.7 Вимоги до транспортування і зберігання

Умови транспортування та зберігання повинні забезпечувати захист носія від пошкоджень.

Програмний виріб міститься на фізичному носії та переданий через microUSB.

# 5 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

До складу програмної документації повинні входити: технічне завдання і робочий проект у складі:

- специфікація;
- текст програми;
- опис програми;
- керівництво користувача.

Вся документація до програми повинна задовольняти вимогам державного стандарту до оформлення програмних документів.

# 6 СТАДІЇ ТА ЕТАПИ РОЗРОБКИ

Стадії та етапи розробки програмного продукту представлені у табл. 6.1.

Таблиця 6.1 – Стадії та етапи розробки

| Стадії розробки | Етапи розробки | Терміни виконання |
|---|---|---|
| 1.Технічне завдання (ТЗ) | Постановка задачі | 13.02.23 – 18.02.23 |
| | Огляд літератури та аналіз аналогів | 19.02.23 – 22.02.23 |
| | Розробка структур вхідних і вихідних даних | 23.02.23 – 04.03.23 |
| | Визначення вимог до програми. Вибір та обґрунтування мови програмування | 05.03.23 – 08.03.23 |
| | Узгодження та затвердження ТЗ | 09.03.23 – 11.03.23 |
| 2. Робочий проект | Розробка та програмування логіки програми | 12.03.23 – 02.05.23 |
| | Розробка і реалізація інтерфейсу користувача | 03.05.23 – 19.05.23 |
| | Тестування та відлагодження програми | 20.05.23 – 09.06.23 |
| | Розробка, узгодження та затвердження програмної документації | 10.06.23 – 14.06.23 |

# 7 ПОРЯДОК КОНТРОЛЮ І ПРИЙМАННЯ

Контроль здійснюється за допомогою виконання набору тестів з метою знаходження помилок в програмному продукті та його специфікації. Контроль виконання роботи забезпечується керівником розробки Жеваго О.О.

Прийом програмного продукту здійснюється уповноваженою комісією.

# 8 БІБЛІОГРАФІЧНИЙ СПИСОК

1. Основи стандартизації програмних систем [Текст]: методичні вказівки до дипломного проектування та лабораторних робіт / уклад.: Ю. М. Івченко, В. І. Шинкаренко, В. Г. Івченко; Дніпропетр. нац. ун-т залізн. трансп. ім. акад. В. Лазаряна. – Д.: Вид-во Дніпропетр. нац. ун-ту залізн. трансп. ім. акад. В. Лазаряна, 2009. – 38 с.

ДОДАТОК Б

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАТВЕРДЖУЮ

Проректор

Українського державного університету

науки і технологій

Анатолій РАДКЕВИЧ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ

ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Робочий проєкт

ЛИСТ ЗАТВЕРДЖЕННЯ

1116130.01318-01-ЛЗ

Завідувач кафедри КІТ

___________ Вадим ГОРЯЧКІН

Керівник розробки

___________ Олександр ЖЕВАГО

Виконавець

___________ Владислав ЗАБОЛОТНИЙ

Нормоконтролер

___________ Світлана ВОЛКОВА

2023

ЗАТВЕРДЖЕНО
1116130.01318-01-ЛЗ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Специфікація

1116130.01318-01

Листів 2

2023

| Позначення | Найменування | Примітка |
|---|---|---|
| | Документація | |
| 1116130.01318-01-ЛЗ | Лист затвердження | |
| 1116130.01318-01 12 01-ЛЗ | Лист затвердження | |
| 1116130.01318-01 12 01 | Текст програми | |
| 1116130.01318-01 13 01-ЛЗ | Лист затвердження | |
| 1116130.01318-01 13 01 | Опис програми | |
| 1116130.01318-01 І3 01-ЛЗ | Лист затвердження | |
| 1116130.01318-01 І3 01 | Керівництво користувача | |

ЗАТВЕРДЖЕНО
1116130.01318-01 12 01-ЛЗ

# РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Текст програми

1116130.01318-01 12 01

Листів 26

2023

# АНОТАЦІЯ

Документ 1116130.01318-01 12 01 «Мобільний додаток для перегляду розкладу занять університету. Текст програми» входить до складу програмної документації на додаток, що реалізує мобільний додаток для відображення розкладів занять університету.

У даному документі представлено текст програми написаний на мові програмування C#. Об'єм пам'яті, що займає програмний продукт, складає 29,3 Мб. Конфігурація телефона стандартна. Комплекс функціонує в середовищі Android 6.0 та вище або iOS 9.0 та вище.

Програма розроблена в середовищі Visual Studio 2022 за допомогою технологій Xamarin. В якості СУБД використовується SQLite.

# ЗМІСТ

# 1 СХЕМА ВЗАЄМОДІЇ МОДУЛІВ

Програма побудована на основі Onion архітектури, яка складається з наступних модулів:

- Core – рівень бізнес логіки, у якому описується логіка взаємодії між даними;
- Domain – рівень домену, у якому описуються моделі даних;
- Persistence – рівень доступу до даних, який служить для організації збереження даних;
- Mobile – рівень представлення даних, який служить для відображення даних;
- Mobile.Android – Android специфічний рівень, у якому особливі налаштування для операційної системи Android;
- Mobile.iOS – iOS специфічний рівень, у якому особливі налаштування для операційної системи Android;

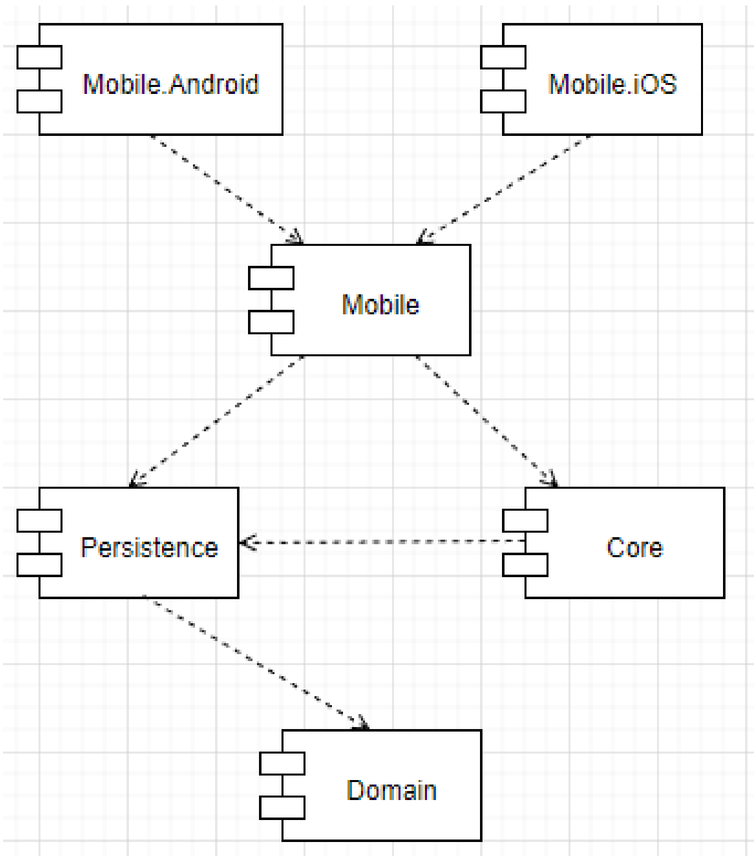



Рисунок 1 – Архітектура взаємодії між модулями

# 2 ТЕКСТ ПРОГРАМИ

## 2.1 Текст модуля Core

### 2.1.1 Файл ClassFileParser.cs

У даному файлі виконується алгоритм парсингу розкладу занять.

```
using Domain.Enums;
using Domain.Models;
using NPOI.HSSF.UserModel;
using NPOI.SS.UserModel;
using NPOI.XSSF.UserModel;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Threading.Tasks;

namespace Core.Parsers
{
    public class ClassFileParser : IFileScheduleParser
    {
        public Task<ParseResult> Parse(byte[] fileData, FileType, string
scheduleName)
        {
            if (fileType == FileType.NotSupported ||
!scheduleName.Contains("Розклад занять"))
            {
                return Task.FromResult(ParseResult.NotSupportedFile());
            }

            var result = ParseFile(fileData, fileType);
            return Task.FromResult(result);
        }

        private ParseResult ParseFile(byte[] fileData, FileType fileType)
        {
            try
            {
                using (var ms = new MemoryStream(fileData))
                {
                    var workbook = fileType == FileType.Xls ? new
HSSFWorkbook(ms) : (IWorkbook)new XSSFWorkbook(ms);
                    var sheet = workbook.GetSheetAt(0);
                    var columnsCount = sheet.GetRow(3).Count();
                    var rowCount = sheet.LastRowNum;

                    var classes = new List<Class>();

                    for (int columnindex = 2; columnindex < columnsCount;
columnindex++)
                    {
                        var cells = new List<ICell>();

                        for (var rowIndex = 3; rowIndex <= rowCount;
rowIndex++)
                        {
                            var row = sheet.GetRow(rowIndex);

                            if (row == null)
                            {
                                rowCount = rowIndex - 1;
                                break;
                            }

                            var cell = row.GetCell(columnindex);
                            cells.Add(cell);
                        }

                        var weekDayCounter = 1;
                        var numberCounter = 1;

                        for (var cellIndex = 0; cellIndex < cells.Count; cellIndex
+= rowCount)
                        {
                            string groupName = string.Empty,
groupAlternativeName = string.Empty;

                            if (cells[cellIndex].IsMergedCell)
                            {
                                var stringValue = cells[cellIndex].StringCellValue;
                                var groupNameParts = stringValue.Split('\n');

                                groupName = groupNameParts[0];
                                groupAlternativeName = groupNameParts[1];
                            }
                            else
                            {
                                groupName = cells[cellIndex].StringCellValue;
                                groupAlternativeName = cells[cellIndex +
1].StringCellValue;
                            }

                            var group = new Group(groupName,
groupAlternativeName);

                            for (var index = cellIndex + 2; index < rowCount &&
index + 4 < rowCount; index += 4)
                            {
                                if (index + 5 > cells.Count)
                                {
                                    break;
                                }

                                if (numberCounter > 8)
                                {
                                    weekDayCounter++;
                                    numberCounter = 1;
                                }

                                if (cells[index] == null)
                                {
                                    continue;
                                }

                                if (cells[index].IsMergedCell)
                                {
                                    if
(!string.IsNullOrEmpty(cells[index].StringCellValue))
                                    {
                                        var teacher = !string.IsNullOrEmpty(cells[index
+ 3].StringCellValue) ? new Teacher(cells[index + 3].StringCellValue) :
new Teacher("-");
                                        var @class = new
Class(cells[index].StringCellValue, group, teacher, null,
weekDayCounter, numberCounter, WeekType.None);

                                        classes.Add(@class);
                                    }
                                }
                                else
                                {
                                    if
(!string.IsNullOrEmpty(cells[index].StringCellValue) )
                                    {
                                        var teacher = !string.IsNullOrEmpty(cells[index
+ 1].StringCellValue) ? new Teacher(cells[index + 1].StringCellValue) :
new Teacher("-");
                                        var @class = new
Class(cells[index].StringCellValue, group, teacher, null,
weekDayCounter, numberCounter, WeekType.Numerator);

                                        classes.Add(@class);
                                    }
```

```
                        if (cells[index + 2] == null)
                        {
                            continue;
                        }

                        if (!string.IsNullOrEmpty(cells[index +
2].StringCellValue))
                        {
                            var teacher = cells[index + 3].StringCellValue !=
"" ? new Teacher(cells[index + 3].StringCellValue) : new Teacher("-");
                            var @class = new Class(cells[index +
2].StringCellValue, group, teacher, null, weekDayCounter,
numberCounter, WeekType.Denominator);

                            classes.Add(@class);
                        }
                    }
                    numberCounter++;
                }
            }
        }
        var res = new ParseResult(classes.ToArray(), null);
        return new ParseResult(classes.ToArray(), null);
    }
}
catch (Exception e)
{
    return ParseResult.ParseFailed(e.Message);
}
}
}
}
```

### 2.1.2 Файл IFileScheduleParser.cs

У даному файлі створений інтерфейс для класів парсингу.

```
using System.Threading.Tasks;

namespace Core.Parsers
{
    public interface IFileScheduleParser
    {
        Task<ParseResult> Parse(byte[] fileData, FileType, string
scheduleName);
    }
}
```

### 2.1.3 Файл ModuleFileParser.cs

У даному файлі виконується алгоритм парсингу розкладу модулів.

```
using Domain.Enums;
using Domain.Models;
using NPOI.HSSF.UserModel;
using NPOI.SS.UserModel;
using NPOI.XSSF.UserModel;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Threading.Tasks;

namespace Core.Parsers
{
    public class ModuleFileParser : IFileScheduleParser
    {
        public Task<ParseResult> Parse(byte[] fileData, FileType, string
scheduleName)
        {
            if (fileType == FileType.NotSupported ||
!scheduleName.Contains("МК"))
            {
                return Task.FromResult(ParseResult.NotSupportedFile());
            }

            var result = ParseFile(fileData, fileType);
            return Task.FromResult(result);
        }

        private ParseResult ParseFile(byte[] fileData, FileType fileType)
        {
            try
            {
                using (var ms = new MemoryStream(fileData))
                {
                    var workbook = fileType == FileType.Xls ? new
HSSFWorkbook(ms) : (IWorkbook)new XSSFWorkbook(ms);

                    int countSheet = 2;
                    var classes = new List<Class>();
                    for (var sheetIndex = 0; sheetIndex < countSheet;
sheetIndex++)
                    {
                        var sheet = workbook.GetSheetAt(sheetIndex);

                        var columnsCount = sheet.GetRow(3).Count();
                        var rowCount = sheet.LastRowNum;

                        for (var columnIndex = 0; columnIndex < columnsCount;
columnIndex += 2)
                        {
                            var cells = new List<ICell>();

                            for (var rowIndex = 3; rowIndex <= rowCount;
rowIndex++)
                            {
                                var row = sheet.GetRow(rowIndex);

                                if (row == null)
                                {
                                    rowCount = rowIndex - 1;
                                    break;
                                }

                                var cell = row.GetCell(columnIndex);
                                cells.Add(cell);
                            }

                            for (var cellIndex = 0; cellIndex < cells.Count; cellIndex
+= rowCount)
                            {
                                var stringValue = cells[cellIndex].StringCellValue;
                                var groupNameParts = stringValue.Split('\n');
                                var group = new Group(groupNameParts[0],
groupNameParts[1]);

                                for (var index = cellIndex + 1; index < rowCount &&
index + 5 < rowCount; index += 5)
                                {
                                    if (index + 6 > cells.Count)
                                    {
                                        break;
                                    }

                                    if (cells[index] != null
                                        && cells[index + 1] != null
                                        && cells[index + 2] != null
                                        && cells[index + 3] != null
                                        && cells[index + 4].StringCellValue != ""
                                        && !string.IsNullOrEmpty(cells[index +
2].StringCellValue))
                                    {
                                        var subject = cells[index + 2].StringCellValue;
```

```
                    var teacher = new Teacher(cells[index +
3].StringCellValue);
                    var date = cells[index].DateCellValue;
                    var timeStringValue = cells[index +
4].StringCellValue;
                    var timeParts = timeStringValue.Split('-');
                    var fullDate = new DateTime(date.Year,
date.Month, date.Day, int.Parse(timeParts[0]), int.Parse(timeParts[1]), 0);
                    var number =
GetClassNumber(timeStringValue);
                    var subType = cells[index + 1].StringCellValue
== "Консультація" ? ClassSubType.Consultation : ClassSubType.Exam;

                    var @class = new Class(subject, group, teacher,
null, (int)date.DayOfWeek, number, WeekType.None, subType,
fullDate);
                    classes.Add(@class);
                  }
                }
              }
            }
          }
          var res = new ParseResult(classes.ToArray(), null);
          return new ParseResult(classes.ToArray(), null);
        }
      }
      catch (Exception e)
      {
        return ParseResult.ParseFailed(e.Message);
      }
    }

    public static int GetClassNumber(string str)
    {
      switch (str)
      {
        case "8-00":
          return 1;

        case "9-30":
          return 2;

        case "11-00":
          return 3;

        case "13-00":
          return 4;

        case "14-30":
          return 5;

        case "16-00":
          return 6;

        default:
          return 0;
      }
    }
  }
}
```

### 2.1.4 Файл IScheduleProvider.cs

У даному файлі відображений інтерфейс для отримання розкладів із сайту.

```
using Domain.Models;
using System.Threading.Tasks;

namespace Core.Services
{
  public interface IScheduleProvider
  {
    Task<ScheduleInfo[]> GetScheduleInfos();

    Task<ScheduleInfo> ExtendScheduleInfo(ScheduleInfo
initialScheduleInfo);

    Task<Schedule> Get(ScheduleInfo scheduleInfo);
  }
}
```

### 2.1.5 Файл ScheduleService.cs

У даному файлі відображена логіка взаємодії компонентів системи.

```
using Domain;
using Domain.Models;
using Domain.PersistenceInterfaces;
using System.Collections.Concurrent;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace Core.Services
{
  public class ScheduleService
  {
    private readonly IScheduleProvider scheduleProvider;
    private readonly IScheduleRepository scheduleRepository;
    private readonly ITeacherRepository teacherRepository;
    private readonly IGroupRepository groupRepository;

    #region Cache

    private static readonly ConcurrentDictionary<string, string[]>
teacherNamesCache = new ConcurrentDictionary<string, string[]>();
    private static readonly ConcurrentDictionary<string, string[]>
groupNamesCache = new ConcurrentDictionary<string, string[]>();

    #endregion Cache

    public ScheduleService(
      IScheduleProvider scheduleProvider,
      IScheduleRepository scheduleRepository,
      ITeacherRepository teacherRepository,
      IGroupRepository groupRepository)
    {
      this.scheduleProvider = scheduleProvider;
      this.scheduleRepository = scheduleRepository;
      this.teacherRepository = teacherRepository;
      this.groupRepository = groupRepository;
    }

    public async Task<ScheduleInfo[]> GetScheduleInfos()
    {
      var scheduleInfos = await scheduleProvider.GetScheduleInfos();

      if (scheduleInfos?.Any() != true)
      {
        scheduleInfos = await scheduleRepository.Get();
      }

      return scheduleInfos.OrderByDescending(x =>
x.Year).ThenByDescending(x => x.Name).ToArray();
    }

    public async Task LoadSchedule(ScheduleInfo scheduleInfo)
    {
      var extendedScheduleInfo = await
scheduleProvider.ExtendScheduleInfo(scheduleInfo) ?? scheduleInfo;
      var dbScheduleInfo = await
scheduleRepository.GetInfo(scheduleInfo.Name);

      if (dbScheduleInfo == null)
      {
        var schedule = await
scheduleProvider.Get(extendedScheduleInfo);
```

```
            await scheduleRepository.Create(schedule);
        }
        else if (!string.Equals(dbScheduleInfo.Checksum, extendedScheduleInfo.Checksum))
        {
            var schedule = await scheduleProvider.Get(extendedScheduleInfo);

            await scheduleRepository.Delete(extendedScheduleInfo.Name);
            await scheduleRepository.Create(schedule);
        }
    }

    public Task<IEnumerable<Class>> GetClasses(SearchCriteria searchCriteria) => scheduleRepository.GetClasses(searchCriteria);

    public async Task<string[]> GetTeachers(string scheduleName)
    {
        if (teacherNamesCache.TryGetValue(scheduleName, out var teacherNames))
        {
            return teacherNames;
        }

        var teachers = await teacherRepository.Get(scheduleName);
        var sortedResult = teachers.OrderBy(x => x.Name).Select(t => t.Name).ToArray();

        teacherNamesCache.TryAdd(scheduleName, sortedResult);

        return sortedResult;
    }

    public async Task<string[]> GetGroups(string scheduleName)
    {
        if (groupNamesCache.TryGetValue(scheduleName, out var groupNames))
        {
            return groupNames;
        }

        var groups = await groupRepository.Get(scheduleName);
        var sortedResult = groups.OrderBy(x => x.Name).Select(t => t.Name).ToArray();

        groupNamesCache.TryAdd(scheduleName, sortedResult);

        return sortedResult;
    }
  }
}
```

### 2.1.6 Файл SiteScheduleProvider

У даному файлі відбувається отримання розкладів з сайту.

```
using Core.Parsers;
using Domain.Models;
using System;
using System.Collections.Concurrent;
using System.Collections.Generic;
using System.Linq;
using System.Net.Http;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Core.Services
{
  public class SiteScheduleProvider : IScheduleProvider
  {
    private readonly IEnumerable<IFileScheduleParser> fileScheduleParsers;
    private static readonly Uri siteUri = new Uri("https://ust.edu.ua/student/lessons_schedule");

    #region Cache

    private static readonly ConcurrentDictionary<Uri, byte[]> fileCache = new ConcurrentDictionary<Uri, byte[]>();
    private static ScheduleInfo[] siteSchedulesCache;
    private static readonly ConcurrentDictionary<string, ScheduleInfo> scheduleInfoCache = new ConcurrentDictionary<string, ScheduleInfo>();
    private static readonly ConcurrentDictionary<string, Schedule> scheduleCache = new ConcurrentDictionary<string, Schedule>();

    #endregion Cache

    public SiteScheduleProvider(IEnumerable<IFileScheduleParser> fileScheduleParsers)
    {
        this.fileScheduleParsers = fileScheduleParsers;
    }

    public async Task<ScheduleInfo[]> GetScheduleInfos()
    {
        if (siteSchedulesCache != null)
        {
            return siteSchedulesCache;
        }

        var result = await GetInfos();

        if (result.Any())
        {
            siteSchedulesCache = result;
        }

        return result;
    }

    public async Task<ScheduleInfo> ExtendScheduleInfo(ScheduleInfo initialScheduleInfo)
    {
        if (scheduleInfoCache.TryGetValue(initialScheduleInfo.Name, out var scheduleInfo))
        {
            return scheduleInfo;
        }

        if (initialScheduleInfo.Url == null)
        {
            return initialScheduleInfo;
        }

        var fileData = await GetFileData(initialScheduleInfo.Url);

        if (fileData == null)
        {
            return initialScheduleInfo;
        }

        var checksum = GetCheckSum(fileData);
        var result = new ScheduleInfo(initialScheduleInfo.Name, initialScheduleInfo.Year, initialScheduleInfo.Url, checksum);

        scheduleInfoCache.TryAdd(initialScheduleInfo.Name, result);

        return result;
    }

    public async Task<Schedule> Get(ScheduleInfo scheduleInfo)
    {
        if (scheduleCache.TryGetValue(scheduleInfo.Name, out var scheduleFromCache))
        {
            return scheduleFromCache;
        }

        var result = new Schedule(scheduleInfo, null);

        if (string.IsNullOrEmpty(scheduleInfo.Checksum) || scheduleInfo.Url == null)
```

```
            {
                return result;
            }

            var fileData = await GetFileData(scheduleInfo.Url);
            if (fileData?.Any() != true)
            {
                return result;
            }

            var fileType = GetFileType(scheduleInfo.Url.ToString());

            foreach (var fileScheduleParser in fileScheduleParsers)
            {
                var parseResult = await fileScheduleParser.Parse(fileData,
fileType, scheduleInfo.Name);

                if (!string.IsNullOrEmpty(parseResult.ErrorMessage))
                {
                    return result;
                }

                if (!parseResult.NotSupported)
                {
                    var scheduleResult = new Schedule(scheduleInfo,
parseResult.Classes);
                    scheduleCache.TryAdd(scheduleInfo.Name,
scheduleResult);

                    return scheduleResult;
                }
            }

            return result;
        }

        private static FileType GetFileType(string url)
        {
            if (url.EndsWith(".xls"))
            {
                return FileType.Xls;
            }

            if (url.EndsWith(".xlsx"))
            {
                return FileType.Xlsx;
            }

            return FileType.NotSupported;
        }

        private string GetYear(string name)
        {
            var nameParts = name.Split('/');

            if (nameParts.Length == 2)
            {
                var yearFirstPart = nameParts[0].Split(' ').LastOrDefault();
                var yearSecondPart = nameParts[1].Split(' ').FirstOrDefault();

                if (yearFirstPart != null && yearSecondPart != null)
                {
                    return $"{yearFirstPart}/{yearSecondPart}";
                }
            }

            return null;
        }

        private async Task<byte[]> GetFileData(Uri uri)
        {
            if (fileCache.TryGetValue(uri, out var data))
            {
                return data;
            }

            try
            {
                using (var client = new HttpClient())
                {
                    var response = await client.GetAsync(uri);
                    if (response.IsSuccessStatusCode)
                    {
                        var result = await
response.Content.ReadAsByteArrayAsync();
                        fileCache.TryAdd(uri, result);

                        return result;
                    }
                }
            }
            catch (Exception)
            {
            }

            return null;
        }

        private string GetCheckSum(byte[] data)
        {
            var md5 = MD5.Create();
            var hash = md5.ComputeHash(data);

            return BitConverter.ToString(hash).Replace("-", "");
        }

        private async Task<ScheduleInfo[]> GetInfos()
        {
            try
            {
                var handler = new HttpClientHandler
                {
                    AllowAutoRedirect = false,
                    AutomaticDecompression =
System.Net.DecompressionMethods.Deflate |
System.Net.DecompressionMethods.GZip |
System.Net.DecompressionMethods.None
                };

                using (var client = new HttpClient(handler))
                {
                    var response = await client.GetAsync(siteUri);

                    if (!response.IsSuccessStatusCode)
                    {
                        return Array.Empty<ScheduleInfo>();
                    }

                    var html = await response.Content.ReadAsStringAsync();

                    if (string.IsNullOrEmpty(html))
                    {
                        return Array.Empty<ScheduleInfo>();
                    }

                    var parseResult = ParseSiteHtml(html);
                    var infos = new List<ScheduleInfo>();

                    foreach (var item in parseResult)
                    {
                        var info = new ScheduleInfo(item.Key,
GetYear(item.Key), new Uri(item.Value), null);
                        infos.Add(info);
                    }

                    return infos.ToArray();
                }
            }
            catch (Exception)
            {
                return Array.Empty<ScheduleInfo>();
            }
        }

        private IDictionary<string, string> ParseSiteHtml(string html)
        {
            var result = new Dictionary<string, string>();
            var doc = new HtmlAgilityPack.HtmlDocument();
```

```
            doc.LoadHtml(html);

            var nodes = doc.DocumentNode.SelectNodes(".//div[@class='col-
xs-12']//div[@class='content']");

            if (nodes?.Any() != true)
            {
                return result;
            }

            foreach (var node in nodes)
            {
                var linkNodes = node.SelectNodes(".//a");

                if (linkNodes?.Any() == true)
                {
                    foreach (var linkNode in linkNodes)
                    {
                        var href = linkNode.GetAttributeValue("href", "");
                        if (!string.IsNullOrEmpty(href))
                        {
                            var scheduleName =
linkNode.InnerText.Replace(" ", " ");
                            string scheduleUrl;
                            if (!href.StartsWith("http"))
                            {
                                if (href.Contains("////"))
                                    scheduleUrl = href.Replace("////", "https://");
                                else
                                    scheduleUrl = "https:" + href;
                            }
                            else
                            {
                                scheduleUrl = href;
                            }

                            if (scheduleUrl.EndsWith(".xls") ||
scheduleUrl.EndsWith(".xlsx"))
                            {
                                result.Add(scheduleName, scheduleUrl);
                            }
                        }
                    }
                }
            }
            return result;
        }
    }
}
```

### 2.1.7 Файл FileType.cs

Даний файл призначений для зарезервованих типів файлу.

```
namespace Core
{
    public enum FileType
    {
        Xls,
        Xlsx,
        NotSupported
    }
}
```

### 2.1.8 Файл ParseResult.cs

Даний файл призначений для отримання результату парсингу.

```
using Domain.Models;
using System;
using System.Linq;

namespace Core
{
    public class ParseResult
    {
        public Class[] Classes { get; }
        public string ErrorMessage { get; }
        public bool NotSupported => (Classes?.Any() != true) &&
string.IsNullOrEmpty(ErrorMessage);

        public ParseResult(Class[] classes, string errorMessage)
        {
            Classes = classes;
            ErrorMessage = errorMessage;
        }

        public static ParseResult NotSupportedFile()
        {
            return new ParseResult(Array.Empty<Class>(), null);
        }

        public static ParseResult ParseFailed(string errorMessage)
        {
            return new ParseResult(Array.Empty<Class>(), errorMessage);
        }
    }
}
```

## 2.2 Текст модуля Domain

### 2.2.1 Файл ClassSubType.cs

Даний файл призначений для зарезервованих типів занять.

```
namespace Domain.Enums
{
    public enum ClassSubType
    {
        Exam,
        Consultation,
        Other
    }
}
```

### 2.2.2 Файл WeekType.cs

Даний файл призначений для зарезервованих типів тижня.

```
namespace Domain.Enums
{
    public enum WeekType
    {
        None,
        Numerator,
        Denominator
    }
}
```

### 2.2.3 Файл Class.cs

У даному файлі описується модель заняття.

```
using Domain.Enums;
using System;

namespace Domain.Models
{
    public class Class
    {
        public string Subject { get; }
        public Group Group { get; }
        public Teacher Teacher { get; }
        public string Auditory { get; }
        public int WeekDay { get; }
        public int Number { get; }
        public WeekType WeekType { get; }
        public ClassSubType SubType { get; }
        public DateTime? Date { get; }

        public Class(string subject, Group group, Teacher teacher, string auditory, int weekDay, int number, WeekType weekType, ClassSubType subType, DateTime? date)
        {
            if (string.IsNullOrEmpty(subject))
            {
                throw new ArgumentNullException(nameof(subject));
            }

            if (subType != ClassSubType.Other && date == null)
            {
                throw new ArgumentNullException(nameof(date));
            }

            Group = group ?? throw new ArgumentNullException(nameof(group));
            WeekDay = weekDay;
            Number = number;
            WeekType = weekType;
            Subject = subject;
            Teacher = teacher;
            Auditory = auditory;
            SubType = subType;
            Date = date;
        }

        public Class(string subject, Group group, Teacher teacher, string auditory, int weekDay, int number, WeekType weekType)
            : this(subject, group, teacher, auditory, weekDay, number, weekType, ClassSubType.Other, null)
        { }

        public override string ToString()
        {
            return $"{Subject} {Group.Name}";
        }
    }
}
```

### 2.2.4 Файл Group.cs

У даному файлі описується модель групи.

```
using System;
using System.Collections.Generic;

namespace Domain.Models
{
    public class Group
    {
        public string Name { get; }
        public string AlternativeName { get; }

        public Group(string name, string alternativeName)
        {
            if (string.IsNullOrEmpty(name))
            {
                throw new ArgumentNullException(nameof(name));
            }

            Name = name;
            AlternativeName = alternativeName;
        }

        public override bool Equals(object obj) => obj is Group group && Name == group.Name;

        public override int GetHashCode() => 539060726 + EqualityComparer<string>.Default.GetHashCode(Name);
    }
}
```

### 2.2.5 Файл Schedule.cs

У даному файлі описується модель розкладу.

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace Domain.Models
{
    public class Schedule
    {
        public ScheduleInfo Info { get; }
        public IEnumerable<Class> Classes { get; }

        public Schedule(ScheduleInfo info, IEnumerable<Class> classes)
        {
            if (info == null)
            {
                throw new ArgumentNullException(nameof(info));
            }

            Classes = classes?.ToList() ?? new List<Class>();
            Info = info;
        }
    }
}
```

### 2.2.6 Файл ScheduleInfo.cs

У даному файлі описується модель інформації про розклад.

```
using System;

namespace Domain.Models
{
    public class ScheduleInfo
    {
        public string Name { get; }
        public string Year { get; }
```

```
        public Uri Url { get; }
        public string Checksum { get; }

        public ScheduleInfo(string name, string year, Uri url, string checksum)
        {
            if (string.IsNullOrEmpty(name))
            {
                throw new ArgumentNullException(nameof(name));
            }

            Name = name;
            Year = year;
            Url = url;
            Checksum = checksum;
        }
    }
}
```

### 2.2.7 Файл Teacher.cs

У даному файлі описується модель викладача.

```
using System;
using System.Collections.Generic;

namespace Domain.Models
{
    public class Teacher
    {
        public string Name { get; }

        public Teacher(string name)
        {
            if (string.IsNullOrEmpty(name))
            {
                throw new ArgumentNullException(nameof(name));
            }

            Name = name;
        }

        public override bool Equals(object obj) => obj is Teacher teacher && Name == teacher.Name;

        public override int GetHashCode() => 539060726 + EqualityComparer<string>.Default.GetHashCode(Name);
    }
}
```

### 2.2.8 Файл IDbPathProvider.cs;

У даному файлі описаний інтерфейс шляху підключення бази даних.

```
namespace Domain.PersistenceInterfaces
{
    public interface IDbPathProvider
    {
        string Path { get; }
    }
}
```

### 2.2.9 Файл IGroupRepository.cs

У даному файлі описаний інтерфейс отримання списку груп.

```
using Domain.Models;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Domain.PersistenceInterfaces
{
    public interface IGroupRepository
    {
        Task<IEnumerable<Group>> Get(string scheduleName);
    }
}
```

### 2.2.10 Файл IScheduleRepository.cs

У даному файлі описаний інтерфейс створення списку розкладів.

```
using Domain.Models;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Domain.PersistenceInterfaces
{
    public interface IScheduleRepository
    {
        Task<ScheduleInfo[]> Get();

        Task<ScheduleInfo> GetInfo(string name);

        Task<IEnumerable<Class>> GetClasses(string name);

        Task<IEnumerable<Class>> GetClasses(SearchCriteria searchCriteria);

        Task Delete(string name);

        Task Create(Schedule schedule);
    }
}
```

### 2.2.11 Файл ITeacherRepository.cs

У даному файлі описаний інтерфейс створення списку викладачів.

```
using Domain.Models;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Domain.PersistenceInterfaces
{
    public interface ITeacherRepository
    {
        Task<IEnumerable<Teacher>> Get(string scheduleName);
    }
}
```

### 2.2.12 Файл SearchCriteria.cs

У даному файлі описаний клас для пошуку занять за критерієм.

```
using System;

namespace Domain
{
  public class SearchCriteria
  {
    public string ScheduleName { get; }
    public string TeacherName { get; }
    public string GroupName { get; }

    public SearchCriteria(string scheduleName, string teacherName,
string groupName)
    {
      if (string.IsNullOrEmpty(scheduleName))
      {
        throw new ArgumentNullException(nameof(scheduleName));
      }

      ScheduleName = scheduleName;
      TeacherName = teacherName;
      GroupName = groupName;
    }
  }
}
```

### 2.3 Текст модуля Persistence

#### 2.3.1 Файл Class.cs

У даному файлі описується структура таблиці Class.

```
using SQLite;

namespace Persistence.SQLiteDb.Entities
{
  [Table(TableNames.ClassTableName)]
  public class Class
  {
    [NotNull]
    public string ScheduleName { get; set; }

    [NotNull]
    public string Subject { get; set; }

    [NotNull]
    public string GroupName { get; set; }
    public string GroupAlternativeName { get; set; }
    public string TeacherName { get; set; }
    public string Auditory { get; set; }
    public int WeekDay { get; set; }
    public int Number { get; set; }

    [NotNull]
    public string WeekType { get; set; }

    [NotNull]
    public string SubType { get; set; }

    public long? Timestamp { get; set; }
  }
}
```

#### 2.3.2 Файл Schedule.cs

У даному файлі описується структура таблиці Schedule.

```
using SQLite;

namespace Persistence.SQLiteDb.Entities
{
  [Table(TableNames.ScheduleTableName)]
  public class Schedule
  {
    [PrimaryKey]
    public string Name { get; set; }

    public string Year { get; set; }

    public string Url { get; set; }

    public string Checksum { get; set; }
  }
}
```

#### 2.3.3 Файл SQLiteDataBase.cs

У даному файлі описується ініціалізація бази даних.

```
using Domain.PersistenceInterfaces;
using Persistence.SQLiteDb.Entities;
using SQLite;

namespace Persistence.SQLiteDb
{
  public class SQLiteDatabase
  {
    public SQLiteAsyncConnection Connection { get; }

    public SQLiteDatabase(IDbPathProvider dbPathProvider)
    {
      Connection = new SQLiteAsyncConnection(dbPathProvider.Path,
SQLiteOpenFlags.Create | SQLiteOpenFlags.ReadWrite |
SQLiteOpenFlags.FullMutex);

      InitializeDatabase();
    }

    private void InitializeDatabase()
    {
      Connection.CreateTableAsync<Class>().Wait();
      Connection.CreateTableAsync<Schedule>().Wait();
    }
  }
}
```

#### 2.3.4 Файл SQLiteGroupRepository.cs

Даний файл призначений для отримання списку груп обраного розкладу.

```
using Domain.Models;
using Domain.PersistenceInterfaces;
```

```
using SQLite;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace Persistence.SQLiteDb
{
   public class SQLiteGroupRepository : IGroupRepository
   {
      private readonly SQLiteAsyncConnection database;

      public SQLiteGroupRepository(SQLiteDatabase sQLiteDatabase)
      {
         database = sQLiteDatabase.Connection;
      }

      public async Task<IEnumerable<Group>> Get(string
scheduleName)
      {
         var groups = await
database.QueryAsync<GroupData>($"SELECT DISTINCT
{nameof(GroupData.GroupName)} FROM
{TableNames.ClassTableName} WHERE
{nameof(Entities.Class.ScheduleName)} = ?", scheduleName);
         return groups.Select(g => new Group(g.GroupName,
null)).ToArray();
      }
   }

   public class GroupData
   {
      public string GroupName { get; set; }
   }
}
```

## 2.3.5 Файл SQLiteScheduleRepository.cs

Даний файл призначений для заповнення таблиці Schedule розкладами.

```
using Domain;
using Domain.Enums;
using Domain.Models;
using Domain.PersistenceInterfaces;
using SQLite;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace Persistence.SQLiteDb
{
   public class SQLiteScheduleRepository : IScheduleRepository
   {
      private readonly SQLiteAsyncConnection database;

      public SQLiteScheduleRepository(SQLiteDatabase sQLiteDatabase)
      {
         database = sQLiteDatabase.Connection;
      }

      public async Task Create(Schedule schedule)
      {
         var record = new Entities.Schedule
         {
            Checksum = schedule.Info.Checksum,
            Name = schedule.Info.Name,
            Url = schedule.Info.Url?.ToString(),
            Year = schedule.Info.Year
         };

         await database.InsertAsync(record);

         var classRecords = schedule.Classes.Select(c => new
Entities.Class
         {
            Auditory = c.Auditory,
            GroupAlternativeName = c.Group.AlternativeName,
            GroupName = c.Group.Name,
            Number = c.Number,
            ScheduleName = schedule.Info.Name,
            Subject = c.Subject,
            SubType = c.SubType.ToString(),
            TeacherName = c.Teacher?.Name,
            WeekDay = c.WeekDay,
            WeekType = c.WeekType.ToString(),
            Timestamp = c.Date?.Ticks
         }).ToList();

         await database.InsertAllAsync(classRecords);
      }

      public async Task Delete(string name)
      {
         await database.Table<Entities.Class>().Where(c =>
c.ScheduleName == name).DeleteAsync();
         await database.DeleteAsync<Entities.Schedule>(name);
      }

      public async Task<ScheduleInfo[]> Get()
      {
         var records = await
database.Table<Entities.Schedule>().ToListAsync();
         return records.Select(Translate).ToArray();
      }

      public async Task<IEnumerable<Class>> GetClasses(string name)
      {
         var records = await database.Table<Entities.Class>().Where(c =>
c.ScheduleName == name).ToListAsync();
         return records.Select(Translate).ToArray();
      }

      public async Task<IEnumerable<Class>> GetClasses(SearchCriteria
searchCriteria)
      {
         var query = database.Table<Entities.Class>().Where(c =>
c.ScheduleName == searchCriteria.ScheduleName);

         if (!string.IsNullOrEmpty(searchCriteria.TeacherName))
         {
            query = query.Where(c => c.TeacherName ==
searchCriteria.TeacherName);
         }

         if (!string.IsNullOrEmpty(searchCriteria.GroupName))
         {
            query = query.Where(c => c.GroupName ==
searchCriteria.GroupName);
         }

         var records = await query.ToListAsync();
         return records.Select(Translate).ToArray();
      }

      public async Task<ScheduleInfo> GetInfo(string name)
      {
         var record = await
database.Table<Entities.Schedule>().FirstOrDefaultAsync(s => s.Name
== name);
         return record == null ? null : Translate(record);
      }

      private Class Translate(Entities.Class @class)
      {
         return new Class(
            @class.Subject,
            new Group(@class.GroupName,
@class.GroupAlternativeName),
            new Teacher(@class.TeacherName),
            @class.Auditory,
            @class.WeekDay,
```

```
            @class.Number,
            (WeekType)Enum.Parse(typeof(WeekType),
@class.WeekType),
            (ClassSubType)Enum.Parse(typeof(ClassSubType),
@class.SubType),
            @class.Timestamp == null ? null : new
DateTime(@class.Timestamp.Value));
        }

        private ScheduleInfo Translate(Entities.Schedule schedule)
        {
            return new ScheduleInfo(schedule.Name, schedule.Year,
string.IsNullOrEmpty(schedule.Url) ? null : new Uri(schedule.Url),
schedule.Checksum);
        }
    }
}
```

### 2.3.6 Файл SQLiteTeacherRepoditory.cs

Даний файл призначений для отримання списку викладачів обраного розкладу.

```
using Domain.Models;
using Domain.PersistenceInterfaces;
using SQLite;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace Persistence.SQLiteDb
{
    public class SQLiteTeacherRepository : ITeacherRepository
    {
        private readonly SQLiteAsyncConnection database;

        public SQLiteTeacherRepository(SQLiteDatabase sQLiteDatabase)
        {
            database = sQLiteDatabase.Connection;
        }

        public async Task<IEnumerable<Teacher>> Get(string
scheduleName)
        {
            var teachers = await
database.QueryAsync<TeacherData>($"SELECT DISTINCT
{nameof(TeacherData.TeacherName)} FROM
{TableNames.ClassTableName} WHERE
{nameof(Entities.Class.ScheduleName)} = ?", scheduleName);
            return teachers.Select(g => new
Teacher(g.TeacherName)).ToArray();
        }
    }

    public class TeacherData
    {
        public string TeacherName { get; set; }
    }
}
```

### 2.3.7 Файл TableName.cs

У даному файлі описуються назви таблиць бази даних.

```
namespace Persistence.SQLiteDb
{
    public static class TableNames
    {
        public const string ClassTableName = "Classes";
        public const string ScheduleTableName = "Schedules";
    }
}
```

## 2.4 Текст модуля Mobile

### 2.4.1 Файл App.xaml

Файл призначений для ініціалізації та запуску додатку.

```
<?xml version="1.0" encoding="utf-8" ?>
<Application xmlns="http://xamarin.com/schemas/2014/forms"
          xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
          x:Class="Mobile.App">
    <Application.Resources>
    </Application.Resources>
</Application>
```

### 2.4.2 Файл App.cs

Файл у якому здійснюються налаштування додатку під час його запуску та роботи.

```
using Microsoft.Extensions.DependencyInjection;
using System;
using Xamarin.Forms;

namespace Mobile
{
    public partial class App : Application
    {
        public App(Action<IServiceCollection>
addPlatformSpecificServices)
        {
            InitializeComponent();

            Startup.Init(addPlatformSpecificServices);

            MainPage = new NavigationPage(new MainPage());
        }

        protected override void OnStart()
        {
        }

        protected override void OnSleep()
        {
        }

        protected override void OnResume()
        {
        }
    }
}
```

### 2.4.3 Файл MainPage.xaml

У даному файлі відбувається налаштування інтерфейсу головної сторінки.

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
          xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
          x:Class="Mobile.MainPage"
          Title="УДУНТ">

   <StackLayout Padding="10,10,10,10">
      <Frame BackgroundColor="#2196F3" Padding="24"
CornerRadius="0">
         <Label Text="Оберіть розклад"
HorizontalTextAlignment="Center" TextColor="White" FontSize="32"
/>
      </Frame>

      <ListView x:Name="listView"  ItemSelected="onScheduleSelected"
HorizontalOptions="Center">

      </ListView>

   </StackLayout>

</ContentPage>
```

### 2.4.4 Файл MainPage.cs

Файл у якому здійснюються взаємодія інтерфейсу з процесами додатку.

```
using Core.Services;
using Domain.Models;
using Microsoft.Extensions.DependencyInjection;
using NPOI.HSSF.Record;
using System.Linq;
using Xamarin.Forms;

namespace Mobile
{
   public partial class MainPage : ContentPage
   {
      private readonly ScheduleService scheduleService;
      private ScheduleInfo[] scheduleInfos;

      public MainPage()
      {
         InitializeComponent();
         scheduleService =
Startup.ServiceProvider.GetService<ScheduleService>();
      }

      protected override async void OnAppearing()
      {

         scheduleInfos = await scheduleService.GetScheduleInfos();
         listView.ItemsSource = scheduleInfos.Select(s => s.Name);
         listView.SelectedItem = "";
      }

      private async void onScheduleSelected(object sender,
SelectedItemChangedEventArgs e)
      {
         var selectedScheduleName = e.SelectedItem.ToString();
         if (selectedScheduleName != "")
         {
            if (selectedScheduleName.Contains("Розклад занять") ||
selectedScheduleName.Contains("МК"))
            {
               var selectIndex = e.SelectedItemIndex;
               await
scheduleService.LoadSchedule(scheduleInfos[selectIndex]);
               await Navigation.PushAsync(new
SchedulePage(selectedScheduleName));
            }
            else
            {
               await DisplayAlert("Увага!", "Відображення даного
розкладу знаходиться в розробці!", "OK");
            }
         }
      }
   }
}
```

### 2.4.5 Файл SchedulePage.xaml

У даному файлу відбувається налаштування інтерфейсу сторінки відображення розкладу.

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
          xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
          x:Class="Mobile.SchedulePage"
          Title="Розклад занять"
          x:Name="schedulePage">

   <StackLayout Padding="10,10,10,10">
      <ScrollView>
         <StackLayout>
            <StackLayout>
               <StackLayout Orientation="Horizontal"
HorizontalOptions="Center">
                  <RadioButton x:Name="groupRadioButton" Content
="Студент" IsChecked="True"
CheckedChanged="onScheduleTypeChange" />
                  <RadioButton x:Name="teacherRadioButton" Content
="Викладач" IsChecked="False"
CheckedChanged="onScheduleTypeChange" />
               </StackLayout>

               <StackLayout>
                  <Picker x:Name="groupPicker"
SelectedIndexChanged="onSelectedGroupChange" Title="Оберіть
групу" >
                     <Picker.Items>
                     </Picker.Items>
                  </Picker>
                  <Picker x:Name="teacherPicker"
SelectedIndexChanged="onSelectedTeacherChange" Title="Оберіть
викладача" IsVisible="False">
                     <Picker.Items>
                     </Picker.Items>
                  </Picker>
               </StackLayout>
            </StackLayout>

            <StackLayout Orientation="Horizontal">
               <CheckBox x:Name="mondayCheckbox"
CheckedChanged="onMondayCheckChange" />
               <Label Text="Понеділок" FontSize="26"
FontAttributes="Bold" />
            </StackLayout>
```

```
<Frame x:Name="mondayFrame" IsVisible="false" BackgroundColor="Gray">
  <StackLayout>
    <StackLayout Orientation="Horizontal">
      <Label Text="I" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="monday_1Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="II" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="monday_2Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="III" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="monday_3Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="IV" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="monday_4Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="V" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="monday_5Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="VI" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="monday_6Class_StackLayout">
      </StackLayout>
    </StackLayout>
  </StackLayout>
</Frame>

<StackLayout Orientation="Horizontal">
  <CheckBox x:Name="tuesdayCheckbox" CheckedChanged="onTuesdayCheckChange" />
  <Label Text="Вівторок" FontSize="26" FontAttributes="Bold" />
</StackLayout>

<Frame x:Name="tuesdayFrame" IsVisible="false" BackgroundColor="Gray">
  <StackLayout>
    <StackLayout Orientation="Horizontal">
      <Label Text="I" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="tuesday_1Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="II" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="tuesday_2Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="III" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="tuesday_3Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="IV" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="tuesday_4Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="V" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="tuesday_5Class_StackLayout">
      </StackLayout>
    </StackLayout>

    <BoxView Color="#FCFCFC"  HeightRequest="2" HorizontalOptions="Fill" />
    <StackLayout Orientation="Horizontal">
      <Label Text="VI" FontSize="22" FontAttributes="Bold" WidthRequest="25" />
      <BoxView Color="#FCFCFC"  WidthRequest="2" HorizontalOptions="Fill" />
      <StackLayout x:Name="tuesday_6Class_StackLayout">
      </StackLayout>
    </StackLayout>
  </StackLayout>
</Frame>
```

```
            <StackLayout Orientation="Horizontal">
              <CheckBox x:Name="wednesdayCheckbox"
CheckedChanged="onWednesdayCheckChange" />
              <Label Text="Середа" FontSize="26"
FontAttributes="Bold" />
            </StackLayout>

            <Frame x:Name="wednesdayFrame" IsVisible="false"
BackgroundColor="Gray">
              <StackLayout>
                <StackLayout Orientation="Horizontal">
                  <Label Text="I" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="wednesday_1Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="II" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="wednesday_2Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="III" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="wednesday_3Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="IV" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="wednesday_4Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="V" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="wednesday_5Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="VI" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="wednesday_6Class_StackLayout">
                  </StackLayout>
                </StackLayout>
              </StackLayout>
            </Frame>

            <StackLayout Orientation="Horizontal">
              <CheckBox x:Name="thursdayCheckbox"
CheckedChanged="onThursdayCheckChange" />
              <Label Text="Четвер" FontSize="26"
FontAttributes="Bold" />
            </StackLayout>

            <Frame x:Name="thursdayFrame" IsVisible="false"
BackgroundColor="Gray">
              <StackLayout>
                <StackLayout Orientation="Horizontal">
                  <Label Text="I" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="thursday_1Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="II" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="thursday_2Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="III" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="thursday_3Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="IV" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="thursday_4Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="V" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                  <StackLayout
x:Name="thursday_5Class_StackLayout">
                  </StackLayout>
                </StackLayout>

                <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
                <StackLayout Orientation="Horizontal">
                  <Label Text="VI" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                  <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
```

```
                <StackLayout
x:Name="thursday_6Class_StackLayout">
                </StackLayout>
              </StackLayout>
            </StackLayout>
          </Frame>

          <StackLayout Orientation="Horizontal">
            <CheckBox x:Name="fridayCheckbox"
CheckedChanged="onFridayCheckChange" />
            <Label Text="П'ятниця" FontSize="26"
FontAttributes="Bold" />
          </StackLayout>

          <Frame x:Name="fridayFrame" IsVisible="false"
BackgroundColor="Gray">
            <StackLayout>
              <StackLayout Orientation="Horizontal">
                <Label Text="I" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                <StackLayout x:Name="friday_1Class_StackLayout">
                </StackLayout>
              </StackLayout>

              <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
              <StackLayout Orientation="Horizontal">
                <Label Text="II" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                <StackLayout x:Name="friday_2Class_StackLayout">
                </StackLayout>
              </StackLayout>

              <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
              <StackLayout Orientation="Horizontal">
                <Label Text="III" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                <StackLayout x:Name="friday_3Class_StackLayout">
                </StackLayout>
              </StackLayout>

              <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
              <StackLayout Orientation="Horizontal">
                <Label Text="IV" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                <StackLayout x:Name="friday_4Class_StackLayout">
                </StackLayout>
              </StackLayout>

              <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
              <StackLayout Orientation="Horizontal">
                <Label Text="V" FontSize="22" FontAttributes="Bold"
WidthRequest="25" />
                <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                <StackLayout x:Name="friday_5Class_StackLayout">
                </StackLayout>
              </StackLayout>

              <BoxView Color="#FCFCFC"  HeightRequest="2"
HorizontalOptions="Fill" />
              <StackLayout Orientation="Horizontal">
                <Label Text="VI" FontSize="22"
FontAttributes="Bold" WidthRequest="25" />
                <BoxView Color="#FCFCFC"  WidthRequest="2"
HorizontalOptions="Fill" />
                <StackLayout x:Name="friday_6Class_StackLayout">
                </StackLayout>
              </StackLayout>
            </StackLayout>
          </Frame>
        </StackLayout>
      </ScrollView>
    </StackLayout>
</ContentPage>
```

### 2.4.6 Файл SchedulePage.cs

Файл у якому здійснюються взаємодія інтерфейсу з процесами додатку.

```
using Core.Services;
using Domain;
using Domain.Enums;
using Domain.Models;
using System;
using System.Linq;
using System.Threading.Tasks;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace Mobile
{
  [XamlCompilation(XamlCompilationOptions.Compile)]
  public partial class SchedulePage : ContentPage
  {
    private readonly ScheduleService scheduleService;
    private const int maxWeekday = 5;
    private const int maxNumber = 6;
    private StackLayout[,] sls;

    public readonly string scheduleName;

    public SchedulePage(string scheduleName)
    {
      InitializeComponent();

      this.scheduleName = scheduleName;
      scheduleService =
Startup.ServiceProvider.GetService<ScheduleService>();
      sls = new StackLayout[,]
      {
        {monday_1Class_StackLayout, monday_2Class_StackLayout,
monday_3Class_StackLayout, monday_4Class_StackLayout,
monday_5Class_StackLayout, monday_6Class_StackLayout},
        {tuesday_1Class_StackLayout, tuesday_2Class_StackLayout,
tuesday_3Class_StackLayout, tuesday_4Class_StackLayout,
tuesday_5Class_StackLayout, tuesday_6Class_StackLayout},
        {wednesday_1Class_StackLayout,
wednesday_2Class_StackLayout, wednesday_3Class_StackLayout,
wednesday_4Class_StackLayout, wednesday_5Class_StackLayout,
wednesday_6Class_StackLayout},
        {thursday_1Class_StackLayout, thursday_2Class_StackLayout,
thursday_3Class_StackLayout, thursday_4Class_StackLayout,
thursday_5Class_StackLayout, thursday_6Class_StackLayout},
        {friday_1Class_StackLayout, friday_2Class_StackLayout,
friday_3Class_StackLayout, friday_4Class_StackLayout,
friday_5Class_StackLayout, friday_6Class_StackLayout}
      };
    }

    protected override async void OnAppearing()
    {
      schedulePage.Title = scheduleName;
      await InitializePickers();
    }

    private async Task InitializePickers()
    {
      var groups = await scheduleService.GetGroups(scheduleName);
```

```
        foreach (var groupName in groups)
        {
            groupPicker.Items.Add(groupName);
        }

        var teachers = await
scheduleService.GetTeachers(scheduleName);
        foreach (var teacherName in teachers)
        {
            if (teacherName == "-")
                continue;
            teacherPicker.Items.Add(teacherName);
        }
    }

    private async Task ViewScheduleGroup(string group)
    {
        CleanSL(sls);

        var searchCriteria = new SearchCriteria(scheduleName, null,
group);
        var groupClasses = await
scheduleService.GetClasses(searchCriteria);

        foreach (var groupClass in groupClasses)
        {
            for (int weekDayCounter = 1; weekDayCounter <=
maxWeekday; weekDayCounter++)
            {
                if (groupClass.WeekDay == weekDayCounter)
                {
                    for (int numberCounter = 1; numberCounter <=
maxNumber; numberCounter++)
                    {
                        if (groupClass.Number == numberCounter)
                        {
                            ViewSubjectsGroup(sls, groupClass,
weekDayCounter, numberCounter);
                        }
                    }
                }
            }
        }
    }

    public void ViewSubjectsGroup(StackLayout[,] sls, Class
groupClass, int weekDayCounter, int numberCounter)
    {
        Label labelSeparator = new Label();
        labelSeparator.Text = "--------------------------------------------";
        labelSeparator.FontAttributes = FontAttributes.Bold;
        StackLayout sl;
        BoxView box = new BoxView
        {
            WidthRequest = 150,
            HeightRequest = 90
        };
        if (groupClass.SubType == ClassSubType.Other)
        {
            if (groupClass.WeekType != WeekType.None)
            {
                if (groupClass.WeekType == WeekType.Numerator)
                {
                    sl = new StackLayout();
                    sls[weekDayCounter - 1, numberCounter -
1].Children.Add(sl);
                    ViewSubjectGroup(sl, groupClass);
                }
                else
                {
                    sl = new StackLayout();
                    if (sls[weekDayCounter - 1, numberCounter -
1].Children.Count == 0)
                    {
                        sls[weekDayCounter - 1, numberCounter -
1].Children.Add(box);
                        sls[weekDayCounter - 1, numberCounter -
1].Children.Add(labelSeparator);
                    }
                    else
                    {
                        sls[weekDayCounter - 1, numberCounter -
1].Children.RemoveAt(2);
                    }

                    sls[weekDayCounter - 1, numberCounter -
1].Children.Add(sl);
                    ViewSubjectGroup(sl, groupClass);
                }
                if (sls[weekDayCounter - 1, numberCounter -
1].Children.Count == 1)
                {
                    sls[weekDayCounter - 1, numberCounter -
1].Children.Add(labelSeparator);
                    sls[weekDayCounter - 1, numberCounter -
1].Children.Add(box);
                }
            }
            else
            {
                sl = new StackLayout();
                sls[weekDayCounter - 1, numberCounter -
1].Children.Add(sl);
                ViewSubjectGroup(sl, groupClass);
            }
        }
        else
        {
            if (sls[weekDayCounter - 1, numberCounter -
1].Children.Count > 0)
            {
                sls[weekDayCounter - 1, numberCounter -
1].Children.Add(labelSeparator);
            }
            sl = new StackLayout();
            Label labelDate = new Label();
            labelDate.Text = groupClass.Date.Value.ToShortDateString();
            Label labelSubType = new Label();
            if (groupClass.SubType == ClassSubType.Consultation)
            {
                labelSubType.Text = "Консультація";
            }
            else
            {
                labelSubType.Text = "Модульний контроль";
            }

            Label labelSubject = new Label();
            labelSubject.Text = groupClass.Subject;
            Label labelTeacher = new Label();
            labelTeacher.Text = groupClass.Teacher.Name;

            labelDate.FontSize = 20; labelSubject.FontAttributes =
FontAttributes.Bold;
            labelSubType.FontSize = 20; labelSubject.FontAttributes =
FontAttributes.Bold;
            labelSubject.FontSize = 18; labelSubject.FontAttributes =
FontAttributes.Bold;
            labelTeacher.FontSize = 18; labelTeacher.FontAttributes =
FontAttributes.Italic;

            sl.Children.Add(labelDate);
            sl.Children.Add(labelSubType);
            sl.Children.Add(labelSubject);
            sl.Children.Add(labelTeacher);

            sls[weekDayCounter - 1, numberCounter - 1].Children.Add(sl);
        }
    }

    public void ViewSubjectGroup(StackLayout sl, Class groupClass)
    {
        Label labelSubject = new Label();
        labelSubject.Text = groupClass.Subject;
        Label labelTeacher = new Label();
        labelTeacher.Text = groupClass.Teacher.Name;
        Label labelRoom = new Label();
        labelRoom.Text = "ауд." + groupClass.Auditory;
```

```
        labelSubject.FontSize = 18; labelSubject.FontAttributes =
FontAttributes.Bold;
        labelTeacher.FontSize = 18; labelTeacher.FontAttributes =
FontAttributes.Italic;
        labelRoom.FontSize = 16;

        sl.Children.Add(labelSubject);
        sl.Children.Add(labelTeacher);
        sl.Children.Add(labelRoom);
    }

    private async Task ViewScheduleTeacher(string teacher)
    {
        CleanSL(sls);

        var searchCriteriaTeacher = new SearchCriteria(scheduleName,
teacher, null);
        var teacherClasses = await
scheduleService.GetClasses(searchCriteriaTeacher);
        var sortTeacherClasses = teacherClasses.OrderBy(x =>
x.Date).ToArray();
        foreach (var teacherClass in sortTeacherClasses)
        {
            for (int weekDayCounter = 1; weekDayCounter <=
maxWeekday; weekDayCounter++)
            {
                if (teacherClass.WeekDay == weekDayCounter)
                {
                    for (int numberCounter = 1; numberCounter <=
maxNumber; numberCounter++)
                    {
                        if (teacherClass.Number == numberCounter)
                        {
                            ViewSubjectsTeacher(sls, teacherClass,
weekDayCounter, numberCounter);
                        }
                    }
                }
            }
        }
    }

    public void ViewSubjectsTeacher(StackLayout[,] sls, Class
teacherClass, int weekDayCounter, int numberCounter)
    {
        Label labelSeparator = new Label();
        labelSeparator.Text = "--------------------------------------------";
        labelSeparator.FontAttributes = FontAttributes.Bold;
        StackLayout sl;
        BoxView box = new BoxView
        {
            WidthRequest = 150,
            HeightRequest = 90
        };

        int i = weekDayCounter - 1;
        int j = numberCounter - 1;

        bool isOneGroup = true;
        if (sls[i, j].Children.Count > 0)
        {
            isOneGroup = false;
        }

        if (teacherClass.SubType == ClassSubType.Other)
        {
            if (teacherClass.WeekType != WeekType.None)
            {
                if (teacherClass.WeekType == WeekType.Numerator)
                {
                    if (isOneGroup == false)
                    {
                        sls[i, j].Children.RemoveAt(sls[i, j].Children.Count - 1);
                        sls[i, j].Children.RemoveAt(sls[i, j].Children.Count - 1);
                    }
                    sl = new StackLayout();
                    sls[i, j].Children.Add(sl);
                    ViewSubjectTeacher(sl, teacherClass, isOneGroup);
                    sls[i, j].Children.Add(labelSeparator);
                    sls[i, j].Children.Add(box);
                }
                else
                {
                    sl = new StackLayout();
                    if (sls[i, j].Children.Count == 0)
                    {
                        sls[i, j].Children.Add(box);
                        sls[i, j].Children.Add(labelSeparator);
                    }
                    else
                    {

                        if (sls[i, j].Children[sls[i, j].Children.Count - 2] is
BoxView)
                        {
                            sls[i, j].Children.RemoveAt(sls[i, j].Children.Count -
2);
                        }
                    }

                    string str = "";
                    if (sls[i, j].Children[sls[i, j].Children.Count - 1] is Label
label1)
                    {
                        str = label1.Text;
                    }
                    if (str == labelSeparator.Text)
                    {
                        isOneGroup = true;
                    }
                    else
                    {
                        isOneGroup = false;
                    }
                    sls[i, j].Children.Add(sl);
                    ViewSubjectTeacher(sl, teacherClass, isOneGroup);
                }
            }
            else
            {
                sl = new StackLayout();
                sls[i, j].Children.Add(sl);
                ViewSubjectTeacher(sl, teacherClass, isOneGroup);
            }
        }
        else
        {
            sl = new StackLayout();

            Label labelDate = new Label();
            labelDate.Text = teacherClass.Date.Value.ToShortDateString();
            Label labelSubType = new Label();
            if (teacherClass.SubType == ClassSubType.Consultation)
            {
                labelSubType.Text = "Консультація";
            }
            else
            {
                labelSubType.Text = "Модульний контроль";
            }

            Label labelSubject = new Label();
            labelSubject.Text = teacherClass.Subject;
            Label labelGroup = new Label();
            labelGroup.Text = teacherClass.Group.Name;

            labelDate.FontSize = 20; labelSubject.FontAttributes =
FontAttributes.Bold;
            labelSubType.FontSize = 20; labelSubject.FontAttributes =
FontAttributes.Bold;
            labelSubject.FontSize = 18; labelSubject.FontAttributes =
FontAttributes.Bold;
            labelGroup.FontSize = 18; labelGroup.FontAttributes =
FontAttributes.Italic;


            Label label = new Label();
```

```
            if (sls[i, j].Children.Count > 0)
                label = sls[i, j].Children.OfType<Label>().FirstOrDefault();

            if (sls[i, j].Children.Count == 0)
            {
                sls[i, j].Children.Add(labelDate);
                sl.Children.Add(labelSubType);
                sl.Children.Add(labelSubject);
                sl.Children.Add(labelGroup);
            }
            else if (label.Text != labelDate.Text)
            {
                string str = "";
                if (sls[i, j].Children[sls[i, j].Children.Count - 2] is Label
label1)
                    str = label1.Text;

                if (str != labelDate.Text)
                {
                    sls[i, j].Children.Add(labelSeparator);
                    sls[i, j].Children.Add(labelDate);
                    sl.Children.Add(labelSubType);
                    sl.Children.Add(labelSubject);
                    sl.Children.Add(labelGroup);
                }
                else
                    sl.Children.Add(labelGroup);
            }
            else
            {
                sl.Children.Add(labelGroup);
            }
            sls[i, j].Children.Add(sl);
        }
    }

    public void ViewSubjectTeacher(StackLayout sl, Class
teacherClass, bool isOneGroup)
    {
        Label labelSubject = new Label();
        labelSubject.Text = teacherClass.Subject;
        Label labelGroup = new Label();
        labelGroup.Text = teacherClass.Group.Name;
        Label labelRoom = new Label();
        labelRoom.Text = "ауд." + teacherClass.Auditory;

        labelSubject.FontSize = 18; labelSubject.FontAttributes =
FontAttributes.Bold;
        labelGroup.FontSize = 18; labelGroup.FontAttributes =
FontAttributes.Italic;
        labelRoom.FontSize = 16;

        if (isOneGroup)
        {
            sl.Children.Add(labelSubject);
            sl.Children.Add(labelRoom);
            sl.Children.Add(labelGroup);
        }
        else
        {
            sl.Children.Add(labelGroup);
        }
    }

    public void CleanSL(StackLayout[,] sls)
    {
        int weekDayCounter = 1;
        while (weekDayCounter <= maxWeekday)
        {
            int numberCounter = 1;
            while (numberCounter <= maxNumber)
            {
                sls[weekDayCounter - 1, numberCounter -
1].Children.Clear();
                numberCounter++;
            }
            weekDayCounter++;
        }
    }

    public async void ShowScheduleForGroup()
    {
        var groups = await scheduleService.GetGroups(scheduleName);
        if (groupPicker.SelectedItem != null)
        {
            foreach (var groupName in groups)
            {
                if (groupPicker.Items[groupPicker.SelectedIndex] ==
groupName)
                {
                    await ViewScheduleGroup(groupName);
                    break;
                }
            }
        }
    }

    public async void ShowScheduleForTeacher()
    {
        var teachers = await
scheduleService.GetTeachers(scheduleName);
        if (teacherPicker.SelectedItem != null)
        {
            foreach (var teacherName in teachers)
            {
                if (teacherPicker.Items[teacherPicker.SelectedIndex] ==
teacherName)
                {
                    await ViewScheduleTeacher(teacherName);
                    break;
                }
            }
        }
    }

    public void onSelectedGroupChange(object sender, EventArgs e)
    {
        if (groupRadioButton.IsChecked)
        {
            ShowScheduleForGroup();
        }
        CloseDayFrame();
    }

    public void onSelectedTeacherChange(object sender, EventArgs e)
    {
        if (teacherRadioButton.IsChecked)
        {
            ShowScheduleForTeacher();
        }
        CloseDayFrame();
    }

    private void onMondayCheckChange(object sender,
CheckedChangedEventArgs e)
    {
        mondayFrame.IsVisible = mondayCheckbox.IsChecked;
    }

    private void onTuesdayCheckChange(object sender,
CheckedChangedEventArgs e)
    {
        tuesdayFrame.IsVisible = tuesdayCheckbox.IsChecked;
    }

    private void onWednesdayCheckChange(object sender,
CheckedChangedEventArgs e)
    {
        wednesdayFrame.IsVisible = wednesdayCheckbox.IsChecked;
    }

    private void onThursdayCheckChange(object sender,
CheckedChangedEventArgs e)
    {
        thursdayFrame.IsVisible = thursdayCheckbox.IsChecked;
    }
```

```
        private void onFridayCheckChange(object sender, CheckedChangedEventArgs e)
        {
            fridayFrame.IsVisible = fridayCheckbox.IsChecked;
        }

        private void onScheduleTypeChange(object sender, CheckedChangedEventArgs e)
        {
            if (teacherPicker.SelectedItem == null || groupPicker.SelectedItem == null)
            {
                CleanSL(sls);
            }

            groupPicker.IsVisible = groupRadioButton.IsChecked;
            teacherPicker.IsVisible = !groupRadioButton.IsChecked;

            if (groupRadioButton.IsChecked)
            {
                ShowScheduleForGroup();
            }
            else
            {
                ShowScheduleForTeacher();
            }
            CloseDayFrame();
        }

        private void CloseDayFrame()
        {
            mondayFrame.IsVisible = false;
            tuesdayFrame.IsVisible = false;
            wednesdayFrame.IsVisible = false;
            thursdayFrame.IsVisible = false;
            fridayFrame.IsVisible = false;

            mondayCheckbox.IsChecked = false;
            tuesdayCheckbox.IsChecked = false;
            wednesdayCheckbox.IsChecked = false;
            thursdayCheckbox.IsChecked = false;
            fridayCheckbox.IsChecked = false;
        }
    }
}
```

### 2.4.7 Файл AssemblyInfo.cs

Даний файл надає метадані.

```
using Xamarin.Forms.Xaml;

[assembly: XamlCompilation(XamlCompilationOptions.Compile)]
```

### 2.4.8 Файл DependencyInjectionContainer.cs

У даному файлі реєструються залежності.

```
using Core.Parsers;
using Core.Services;
using Domain.PersistenceInterfaces;
using Microsoft.Extensions.DependencyInjection;
using Persistence.InMemory;
using Persistence.SQLiteDb;
using System;

namespace Mobile
{
    public static class DependencyInjectionContainer
    {
        public static IServiceCollection ConfigureServices(this IServiceCollection services, Action<IServiceCollection> addPlatformSpecificServices)
        {
            addPlatformSpecificServices.Invoke(services);

            //services = services.AddInMemoryRepositories();
            services = services.AddDbRepositories();

            services.AddScoped<IScheduleProvider, SiteScheduleProvider>();

            services.AddScoped<IFileScheduleParser, ClassFileParser>();
            services.AddScoped<IFileScheduleParser, ModuleFileParser>();

            services.AddScoped<ScheduleService>();

            return services;
        }

        private static IServiceCollection AddInMemoryRepositories(this IServiceCollection services)
        {
            services.AddScoped<ITeacherRepository, InMemoryTeacherRepository>();
            services.AddScoped<IGroupRepository, InMemoryGroupRepository>();
            services.AddScoped<IScheduleRepository, InMemoryScheduleRepository>();

            return services;
        }

        private static IServiceCollection AddDbRepositories(this IServiceCollection services)
        {
            services.AddSingleton<SQLiteDatabase>();

            services.AddScoped<ITeacherRepository, SQLiteTeacherRepository>();
            services.AddScoped<IGroupRepository, SQLiteGroupRepository>();
            services.AddScoped<IScheduleRepository, SQLiteScheduleRepository>();

            return services;
        }
    }
}
```

### 2.4.9 Startup.cs;

Даний файл призначений для ініціалізації сервісів.

```
using Microsoft.Extensions.DependencyInjection;
using System;

namespace Mobile
{
    public static class Startup
    {
        public static IServiceProvider ServiceProvider { get; private set; }

        public static IServiceProvider Init(Action<IServiceCollection> addPlatformSpecificServices)
```

```
        {
            var serviceProvider = new ServiceCollection()
                .ConfigureServices(addPlatformSpecificServices)
                .BuildServiceProvider();

            ServiceProvider = serviceProvider;
            return serviceProvider;
        }
    }
}
```

## 2.5 Текст модуля Mobile.Android

### 2.5.1 AndroidDbPathProvider.cs

У даному файлі відбувається ініціалізація шляху бази даних для Android.

```
using Domain.PersistenceInterfaces;
using System;

namespace Mobile.Android
{
    public class AndroidDbPathProvider : IDbPathProvider
    {
        public string Path =>
System.IO.Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.Personal), "schedules.db3");
    }
}
```

### 2.5.2 Файл MainActivity.cs

Даний файл є точкою входу для додатку Android та в ньому додаються сервіси.

```
using Android.App;
using Android.Content.PM;
using Android.OS;
using Android.Runtime;
using Domain.PersistenceInterfaces;
using Microsoft.Extensions.DependencyInjection;

namespace Mobile.Android
{
    [Activity(Label = "УДУНТ Розклад занять", Icon =
"@drawable/icon", Theme = "@style/MainTheme", MainLauncher =
true, ConfigurationChanges = ConfigChanges.ScreenSize |
ConfigChanges.Orientation | ConfigChanges.UiMode |
ConfigChanges.ScreenLayout | ConfigChanges.SmallestScreenSize)]
    public class MainActivity :
global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
    {
        protected override void OnCreate(Bundle savedInstanceState)
        {
            base.OnCreate(savedInstanceState);

            Xamarin.Essentials.Platform.Init(this, savedInstanceState);
            global::Xamarin.Forms.Forms.Init(this, savedInstanceState);
            LoadApplication(new App(AddServices));
        }

        public override void OnRequestPermissionsResult(int requestCode,
string[] permissions, [GeneratedEnum] Permission[] grantResults)
        {

Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode,
permissions, grantResults);

            base.OnRequestPermissionsResult(requestCode, permissions,
grantResults);
        }

        private static void AddServices(IServiceCollection services)
        {
            services.AddSingleton<IDbPathProvider,
AndroidDbPathProvider>();
        }
    }
}
```

## 2.6 Текст модуля Mobile.iOS

### 2.6.1 Файл AppDeleagte.cs

Даний файл є точкою входу для додатку iOS та в ньому додаються сервіси.

```
using Domain.PersistenceInterfaces;
using Foundation;
using Microsoft.Extensions.DependencyInjection;
using UIKit;

namespace Mobile.iOS
{
    [Register("AppDelegate")]
    public partial class AppDelegate :
global::Xamarin.Forms.Platform.iOS.FormsApplicationDelegate
    {
        public override bool FinishedLaunching(UIApplication app,
NSDictionary options)
        {
            global::Xamarin.Forms.Forms.Init();
            LoadApplication(new App(AddServices));

            return base.FinishedLaunching(app, options);
        }

        private static void AddServices(IServiceCollection services)
        {
            services.AddSingleton<IDbPathProvider, iOSDbPathProvider>();
        }
    }
}
```

### 2.6.2 Файл iOSDbPathProvider.cs

У даному файлі відбувається ініціалізація шляху до бази даних для iOS.

```
using Domain.PersistenceInterfaces;
using System;

namespace Mobile.iOS
{
```

```
    public class iOSDbPathProvider : IDbPathProvider
    {
        public string Path =>
System.IO.Path.Combine(Environment.GetFolderPath(Environment.Spec
ialFolder.Personal), "..", "Library", "schedules.db3");
    }
}
```

### 2.6.3 Файл Main.cs

Даний файл призначений для встановлення контролю над виконання програми для iOS.

```
using UIKit;

namespace Mobile.iOS
{
    public class Application
    {
        private static void Main(string[] args)
        {
            UIApplication.Main(args, null, typeof(AppDelegate));
        }
    }
}
```

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАТВЕРДЖУЮ
Проректор
Українського державного університету
науки і технологій
Анатолій РАДКЕВИЧ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Опис програми
ЛИСТ ЗАТВЕРДЖЕННЯ
1116130.01318-01 13 01-ЛЗ

Завідувач кафедри КІТ

___________ Вадим ГОРЯЧКІН

Керівник розробки

___________ Олександр ЖЕВАГО

Виконавець

___________ Владислав ЗАБОЛОТНИЙ

Нормоконтролер

___________ Світлана ВОЛКОВА

2023

ЗАТВЕРДЖЕНО
1116130.01318-01 13 01-ЛЗ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Опис програми

1116130.01318-01 13 01

Листів 21

2023

# АНОТАЦІЯ

Документ 1116130.01318-01 13 01 «Розробка мобільного додатку для перегляду розкладу занять університету. Опис програми» входить до складу програмної документації на додаток, що реалізує мобільний додаток для відображення розкладів занять університету.

У даному документі представлено опис програми: функціональне призначення, опис логічної структури, використані технічні засоби, виклик та завантаження, вхідні та вихідні дані, опис призначеного для користувача інтерфейсу та порядок роботи з програмою. Програма написана на мові C#. Об’єм пам’яті, що займає програма, складає 29,3 Мб. Конфігурація телефона стандартна. Програма кросплатформена, функціонує в середовищі Android 6.0 та вище та iOS 9.0 та вище.

Програма розроблена в середовищі Visual Studio 2022 за допомогою технологій Xamarin. В якості СУБД використовується SQLite.

ЗМІСТ

# 1 ЗАГАЛЬНІ ВІДОМОСТІ

Найменування програми: «Мобільний додаток для перегляду розкладу занять університету».

Програма була розроблена в середовищі програмування Visual Studio 2022 на мові програмування C#, за допомогою технології Xamarin.

Програмний додаток може функціонувати на операційних системах Android 6.0 і вище та iOS 9 і вище.

# 2 ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Даний мобільний додаток дозволяє студентам та викладачам Українського державного університету науки і технологій переглядати розклади занять та модулів, які доступні на офіційному сайті університету у зручному та зрозумілому вигляді.

# 3 ОПИС ЛОГІЧНОЇ СТРУКТУРИ

## 3.1 Алгоритм програми

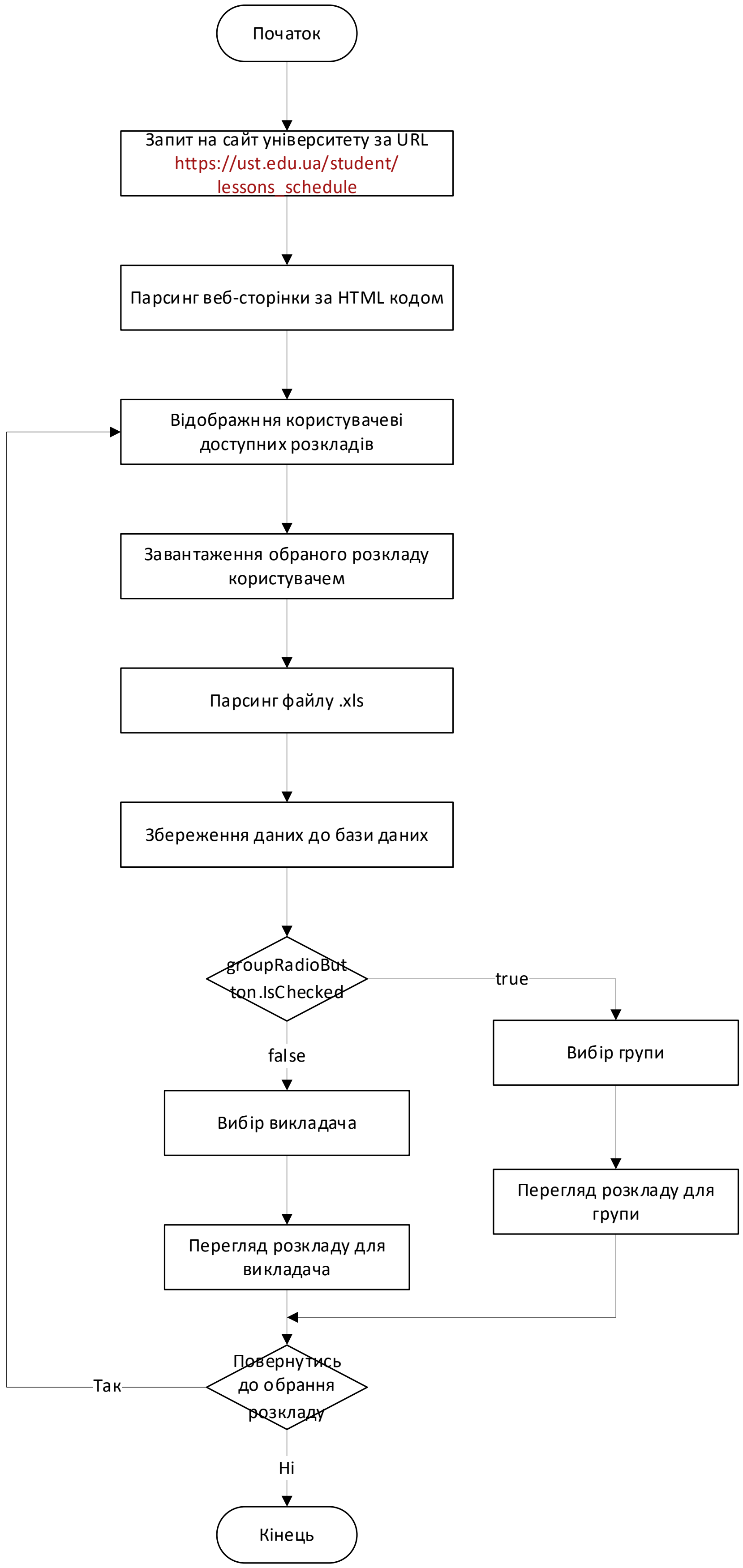



Рисунок 3.1 – Загальний алгоритм програми

3.2 Структура програми

Для розробки програмного додатку було обрано архітектуру Onion, яка ділить проєкт на наступні рівні:

– Core – у даному рівні описується логіка взаємодії між даними;

– Domain – у цьому рівні описуються моделі даних;

– Persistence – даний рівень служить для організації збереження даних;

– Mobile – рівень, який служить для відображення даних;

– специфічний рівень Mobile.Android, у якому особливі налаштування для операційної системи Android;

– специфічний рівень Mobile.iOS, у якому особливі налаштування для операційної системи iOS.

Також в структуру програми було включено патерн Dependency injection, який дозволяє зменшити залежність між класами та полегшує тестування коду.

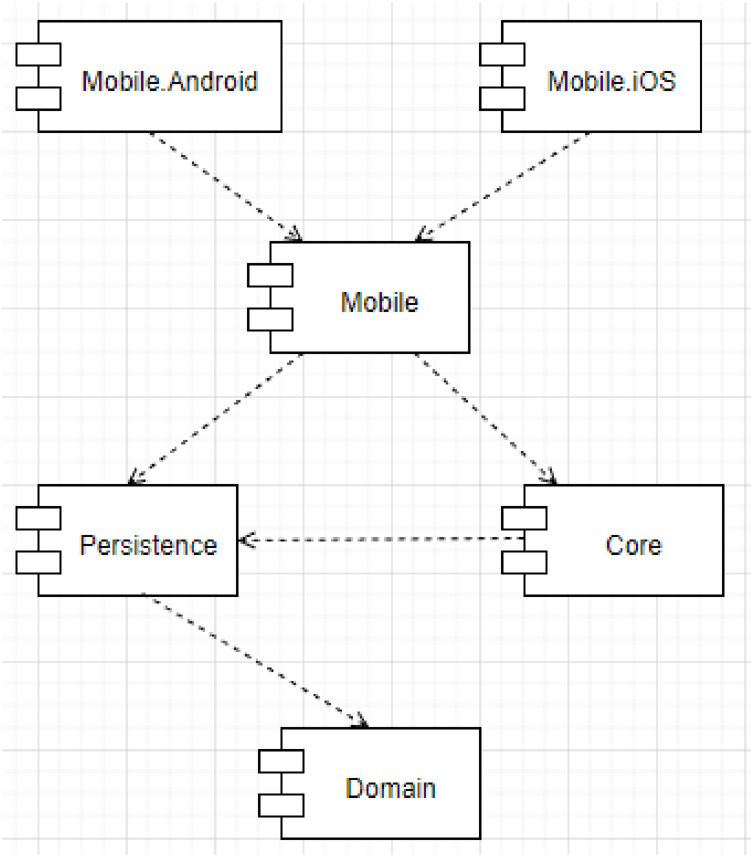



Рисунок 3.2 – Архітектура взаємодії між модулями

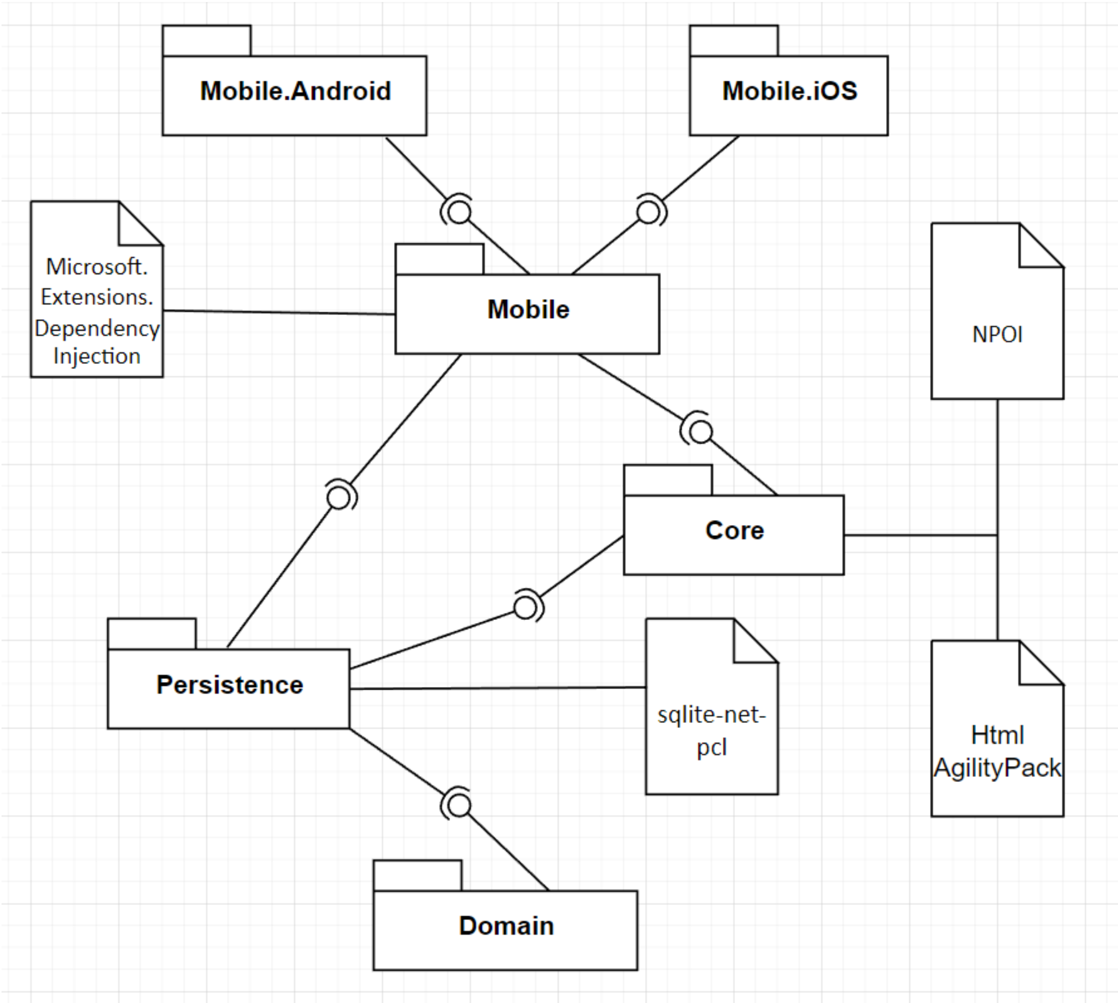



Рисунок 3.3– Діаграма компонентів

На рис 3.4 – 3.7 зображені діаграми класів програми:

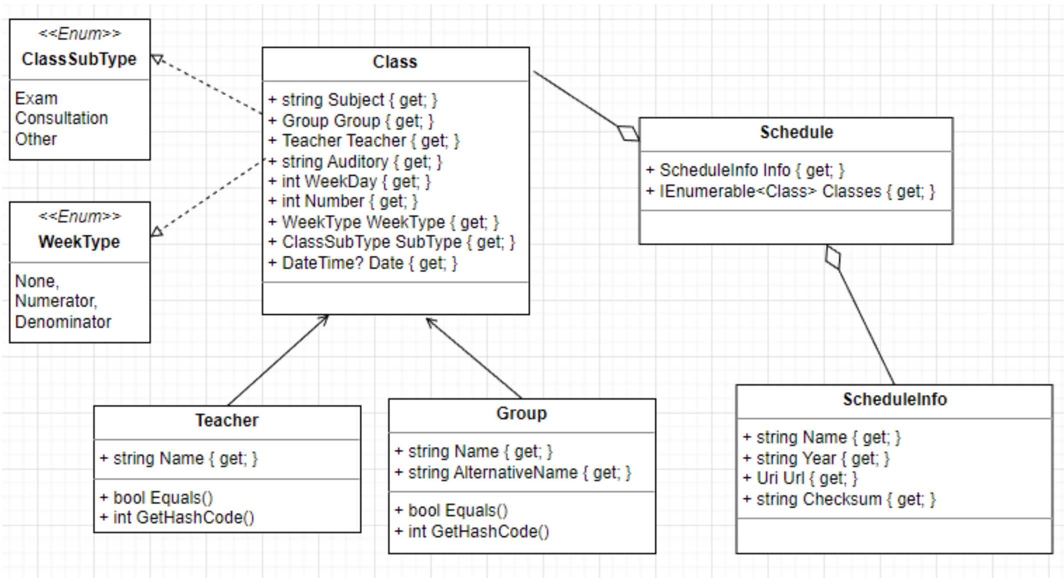



Рисунок 3.4– Діаграма класів пакету Domain

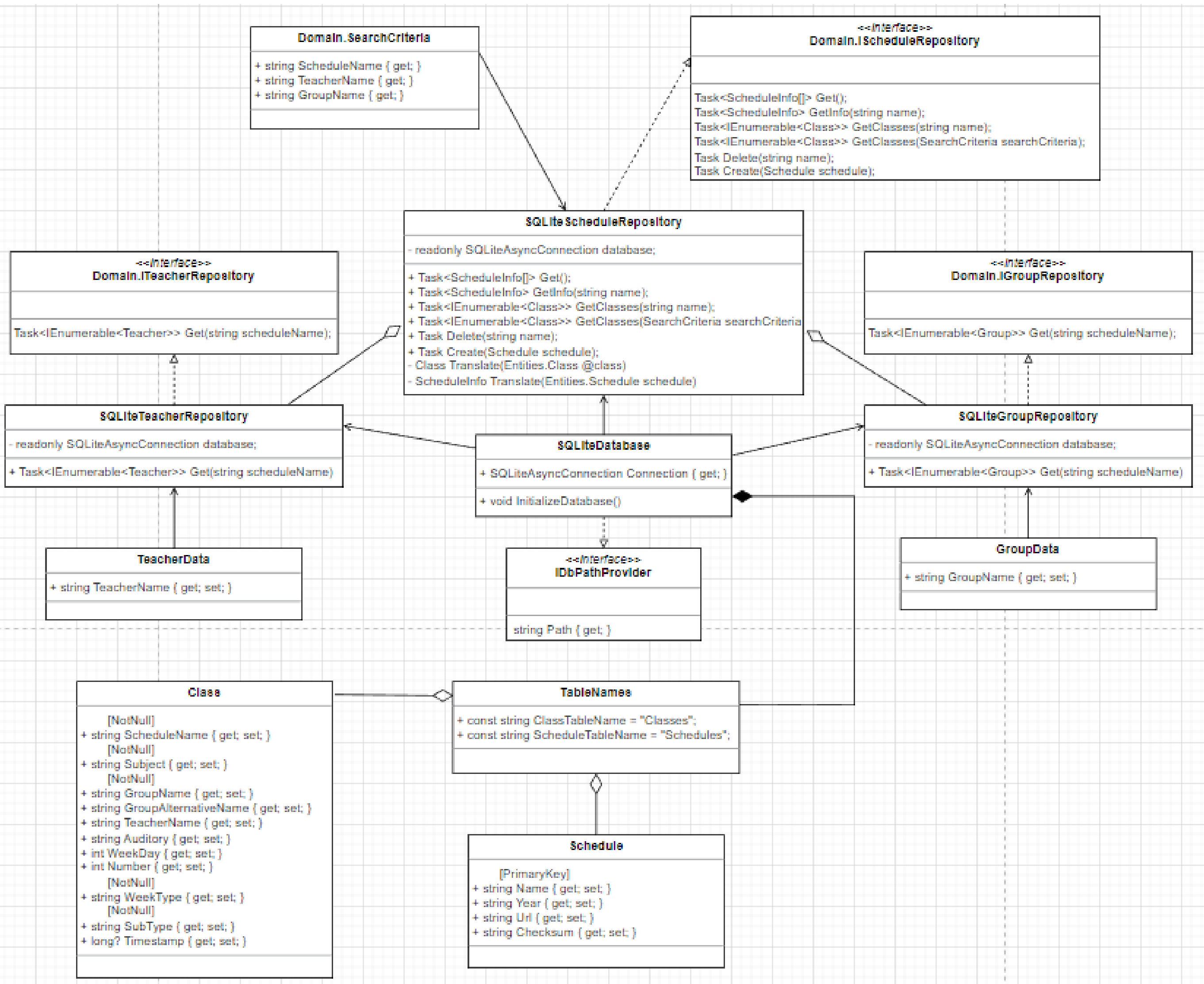



Рисунок 3.5– Діаграма класів пакету Persistence

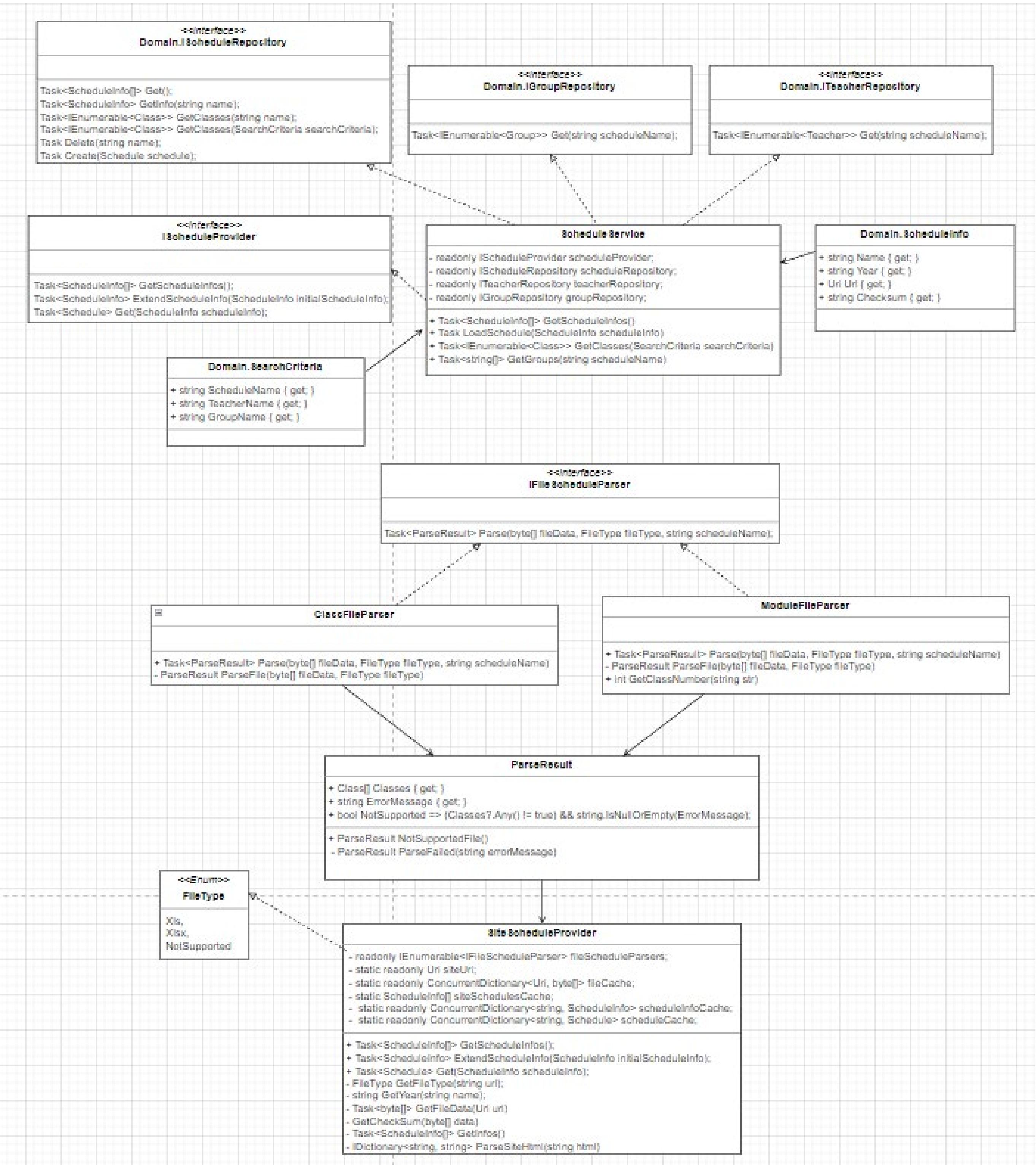



Рисунок 3.6 – Діаграма класів пакету Core

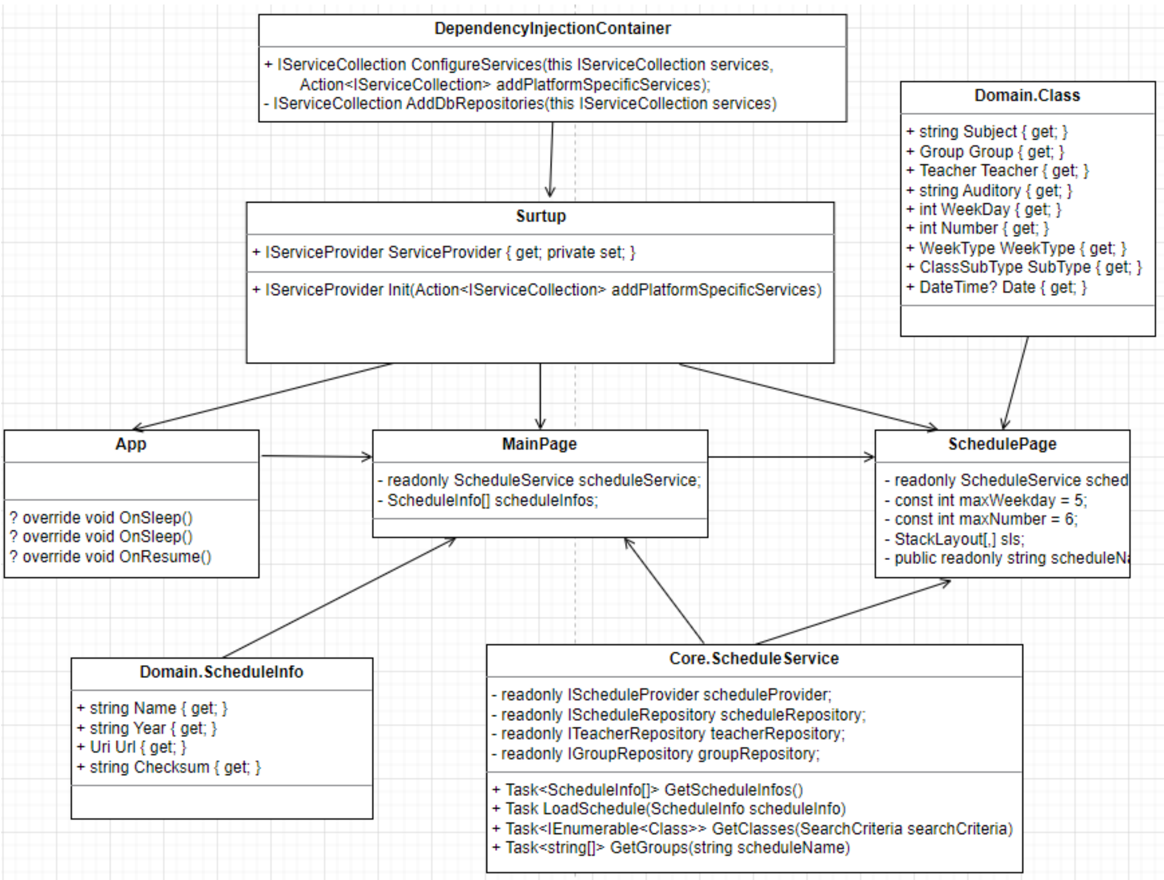



Рисунок 3.7 – Діаграма класів пакету Mobile

## 3.3 Структура бази даних

У ході розробки програми для зберігання розкладів занять була розроблена база даних SQLite.

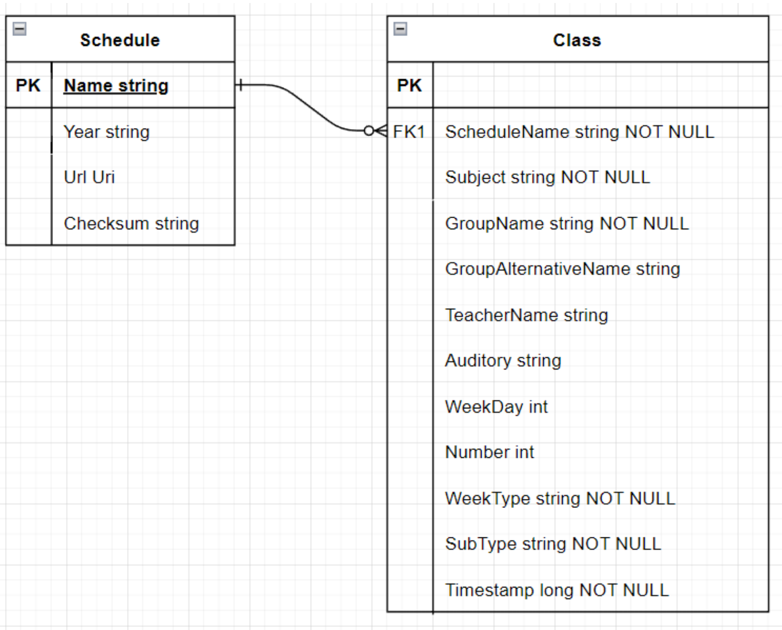



Рисунок 3.8 – Структура бази даних

# 4 ВИКОРИСТАНІ ТЕХНІЧНІ ЗАСОБИ

Програмний продукт, відлагоджувався на мобільному пристрою, що має наступні характеристики:

- операційна система: Android 12.0.
- TypeC порт;
- оперативна пам'ять (RAM): 6 ГБ;
- дисплей: 2280 x 1080 (FHD+);
- діагональ екрану 5,8 дюймів;
- вбудована пам’ять: 128 ГБ;
- підключення до Інтернету через Wi-Fi або мобільну мережу.

# 5 ВИКЛИК І ЗАВАНТАЖЕННЯ

Програму можна встановити за допомогою файлу .apk, який можна завантажити за посиланням https://github.com/vlad910099/ScheduleViewer/tree/main/bin, або завантажити через зовнішній носій інформації через microUSB або TypeC порт.

# 6 ВХІДНІ ДАНІ

Вхідними даними програми, що розробляється є:

– html вміст сторінки сайту ust.edu.ua;

– файл з розкладом.

# 7 ВИХІДНІ ДАНІ

Результатом роботи програми є наступні вихідні дані:

– список розкладів з сайту університету;

– заняття обраного розкладу;

– сформована на основі вхідної інформації база даних.

# 8 ОПИС ІНТЕРФЕЙСУ КОРИСТУВАЧА

Після запуску програми відображається головна сторінка програми зі списком всі наявних розкладів із сайту університету. Користувач повинен обрати розклад (рис. 8.1).

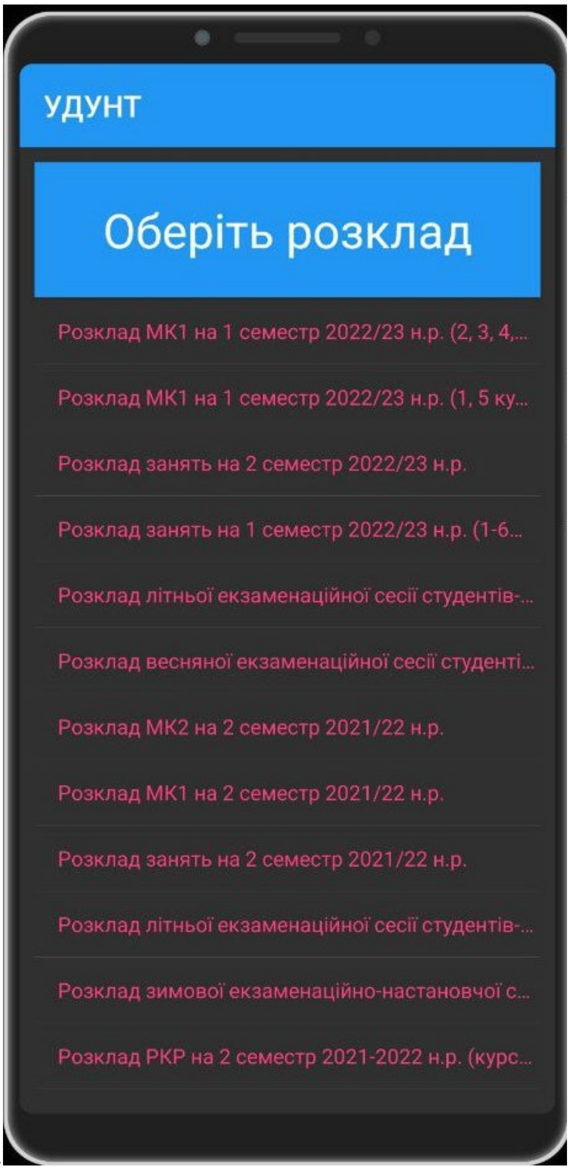



Рисунок 8.1 – Сторінка вибору розкладу

Далі користувачеві необхідно обрати розклад, після чого завантажиться відповідний розклад з’явиться вікно перегляду розкладу (рис. 8.2), де необхідно обрати для кого відобразити розклад та обрати групу чи викладача відповідно. При виборі групи відкривається список доступних груп (рис.8.3).

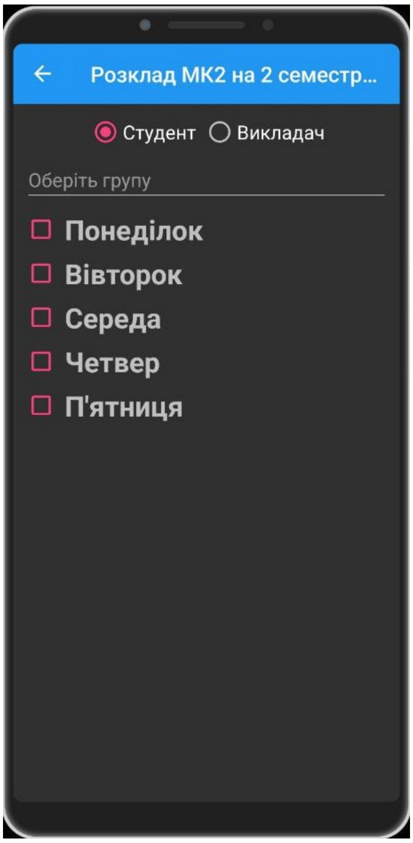



Рисунок 8.2 – Сторінка перегляду розкладу модулів


Рисунок 8.3 – Список вибору групи

Після чого користувач має змогу переглянути розклад модулів (рис. 8.4).

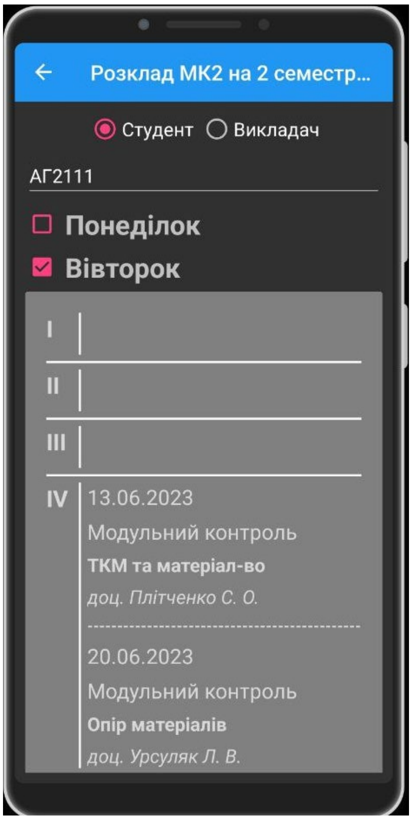



Рисунок 8.4 – Відображення розкладу модулів для групи

Якщо користувач обрав викладача, йому відповідно необхідно обрати наявного викладача зі списку доступних (рис 8.5). Після чого користувач матиме змогу переглянути розклад модулів для обраного викладача (рис 8.6).

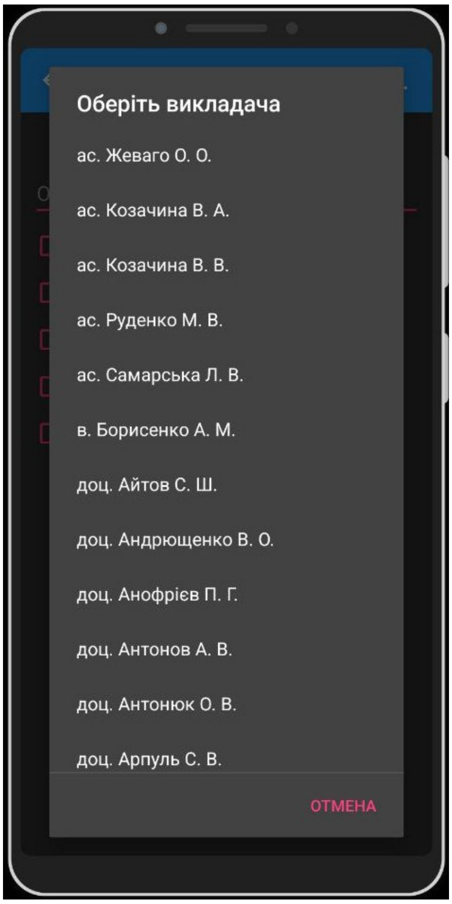



Рисунок 8.5 – Список вибору викладачів

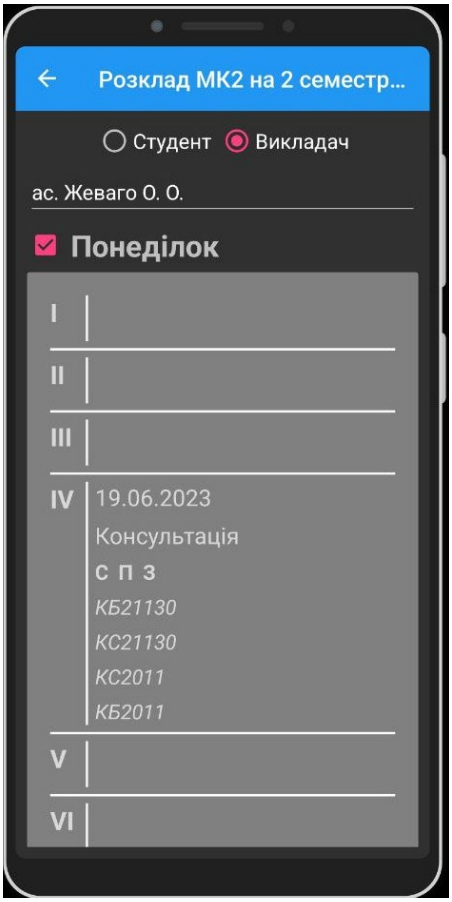



Рисунок 8.6 – Відображення розкладу модулів для викладача

Після чого користувач має змогу перейти на головну сторінку за допомогою кнопки «стрілка ліворуч» у лівому верхньому кутку вікна (рис.8.6), та обрати один з інших наявний розкладів. Якщо вибір користувача впаде на «Розклад занять», тоді з’явиться знову сторінка перегляду (рис. 8.2). Після обрання групи з’явиться відповідний розклад занять для групи (рис. 8.7), та для викладача (рис. 8.8).

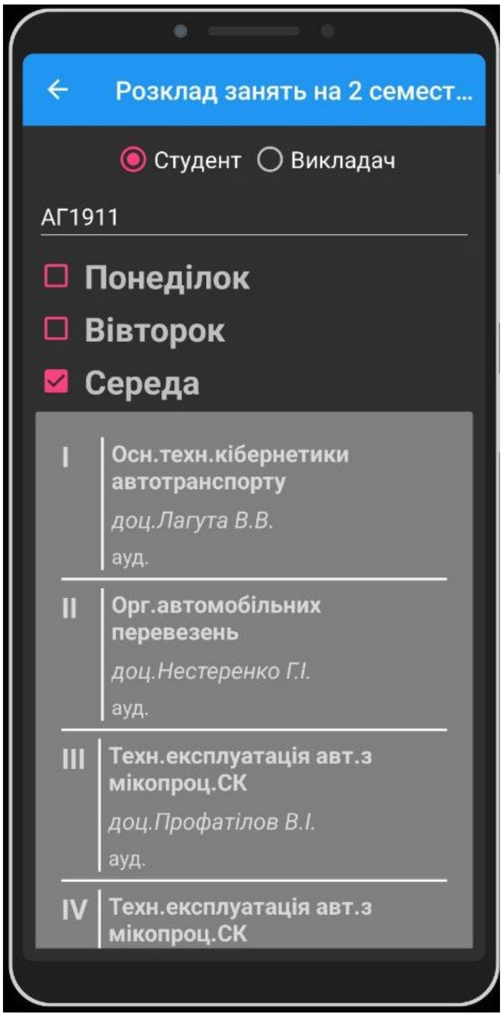



Рисунок 8.7 – Відображення розкладу занять для групи

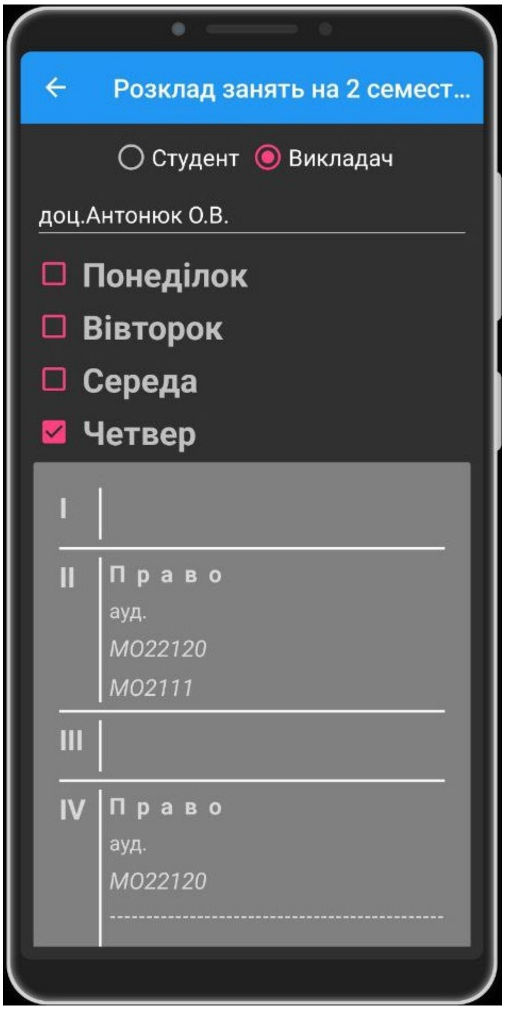



Рисунок 8.8 – Відображення розкладу занять для викладача

# 9 ПОРЯДОК РОБОТИ З ПРОГРАМОЮ

Перед початком роботи необхідно впевнитись, що на мобільному пристрої є підключення до мережі Інтернет через WIFI або мобільну мережу.

# 10 ПОВІДОМЛЕННЯ

У табл. 10.1 представлені повідомлення користувачу, що можуть з'явитися у процесі роботи програми.

Таблиця 10.1. Повідомлення, що з'являються в процесі роботи програми

| Текст повідомлення | Опис ситуації | Рекомендовані дії |
|---|---|---|
| Немає підключення до мережі Інтернет | Не вдалося завантажити список розкладів, бо мобільний пристрій немає підключення до мережі Інтернет | Підключити мобільний пристрій до WIFI або мобільної мережі |
| Увага:<br>Відображення даного розкладу знаходиться в розробці | Під час обрання розкладу, відображення якого ще не оброблено в програмі | Обрати інший розклад |
| Увага:<br>Не вдалося завантажити розклад | Якщо під час виконання програми пропав доступ до мережі Інтернет | Перевірити доступ до мережі Інтернет |

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАТВЕРДЖУЮ
Проректор
Українського державного університету
науки і технологій
Анатолій РАДКЕВИЧ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Керівництво користувача
ЛИСТ ЗАТВЕРДЖЕННЯ
1116130.01318-01 І3 01-ЛЗ

Завідувач кафедри КІТ

___________ Вадим ГОРЯЧКІН

Керівник розробки

___________ Олександр ЖЕВАГО

Виконавець

___________ Владислав ЗАБОЛОТНИЙ

Нормоконтролер

___________ Світлана ВОЛКОВА

2023

ЗАТВЕРДЖЕНО
1116130.01318-01 ІЗ 01-ЛЗ

РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ
ДЛЯ ПЕРЕГЛЯДУ РОЗКЛАДУ ЗАНЯТЬ УНІВЕРСИТЕТУ

Керівництво користувача

1116130.01318-01 ІЗ 01

Листів 13

2023

# АНОТАЦІЯ

Документ 1116130.01318-01 І3 01 «Розробка мобільного додатку для перегляду розкладу занять університету. Керівництво користувача» входить до складу програмної документації на додаток, що реалізує мобільний додаток для відображення розкладів занять університету.

У даному документі представлено керівництво користувача: призначення програми, умови використання, підготовка до роботи, опис операцій, повідомлення користувачу, аварійні ситуації. Програма написана на мові C#. Об'єм пам'яті, що займає програма, складає 29,3 Мб. Конфігурація телефона стандартна. Програма кросплатформна, функціонує в середовищі Android 6.0 та вище та iOS 9.0 та вище.

Програма розроблена в середовищі Visual Studio 2022 за допомогою технологій Xamarin. В якості СУБД використовується SQLite.

# ЗМІСТ

# 1 ПРИЗНАЧЕННЯ ПРОГРАМИ

Програма «Мобільний додаток для перегляду розкладу занять для університету» призначена для використання студентами та викладачами для перегляду розкладу занять та модулів. Програма дозволяє користувачам швидко та зручно переглядати наявні та актуальні розклади занять та модулів Українського державного університету науки і технологій.

## 2 УМОВИ ВИКОРИСТАННЯ ПРОГРАМИ

Програма функціонує в усіх мобільних пристроях, що задовольняють наступні вимоги:

- операційна система: не нижче Android 6.0, iOS 9;
- microUSB або TypeC порт;
- оперативна пам'ять (RAM): 2 ГБ або більше;
- дисплей: HD (1280x720 пікселів) або більше;
- діагональ екрану 5,5 дюймів або більше;
- вбудована пам’ять: 16 ГБ або більше;
- підключення до Інтернету через Wi-Fi або мобільну мережу.

# 3 ПІДГОТОВКА ДО РОБОТИ

3.1 Склад і зміст дистрибутивного носія даних

Дистрибутив програмної системи містить у собі:

- файли програми;
- файл readme.txt;
- завантажувальний файл УДУНТ_Розклад_занять.Android.apk.

3.2 Порядок завантаження

Для початку роботи з програмою необхідно завантажити .apk файл, за посиланням https://github.com/vlad910099/ScheduleViewer/tree/main/Download та запустити процес інсталяції на мобільному пристрої.

3.3 Перевірка працездатності

Для перевірки працездатності програми необхідно виконати наступні дії:

- підключити мобільний пристрій до мережі Інтернет;
- запустити програму;
- обрати розклад занять;
- обрати тип користувача;
- обрати групи/викладача;
- якщо наявний список груп та викладачів то програма працездатна.

# 4 ОПИС ОПЕРАЦІЙ

4.1 Процедура завантаження розкладів

Виконання процедури можливе за наявності підключення мобільного пристрою до мережі Інтернет. Для виконання необхідно запустити програмний додаток.

Після успішного завантаження наявних розкладів із сайту університету з'явиться головна сторінка програми (рис. 4.1)

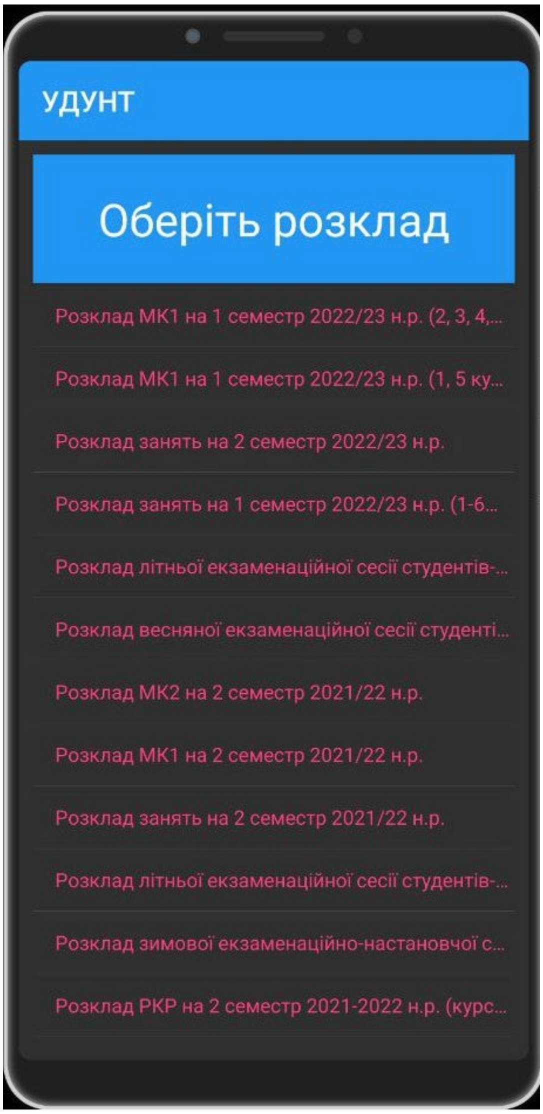



Рисунок 4.1 – Сторінка перегляду розкладів

4.2 Процедура відображення розкладу модулів для студента

Для виконання даної процедури необхідно обрати наявний розклад модулів із списку. Після чого з'явиться сторінка перегляду розкладу (рис. 4.2). Далі необхідно обрати тип користувач як «Студент» та обрати групу зі списку доступних груп (рис. 4.3).

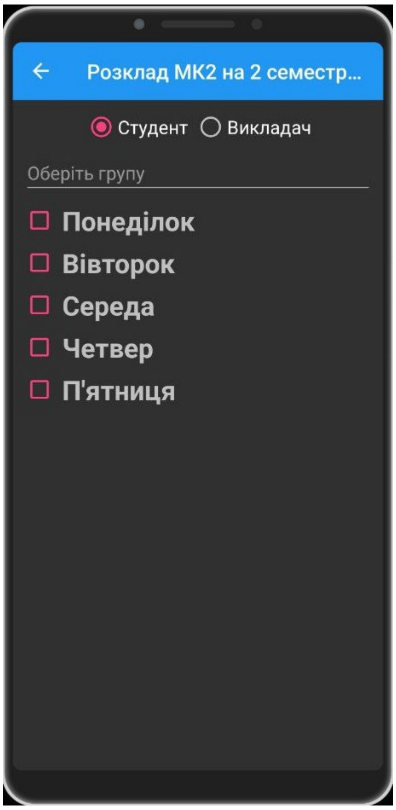



Рисунок 4.2 – Сторінка перегляду розкладу


Рисунок 4.3 – Список вибору групи

Після виконання перелічених вище дій, необхідно обрати дні тижня, де розкриється перелік модулів на обраний день тижня (рис. 4.4).

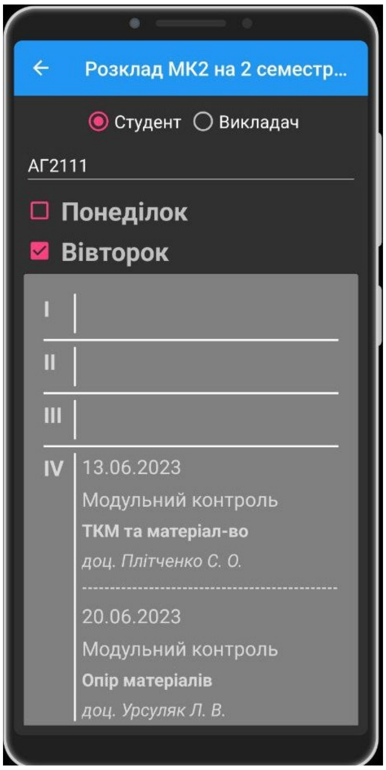



Рисунок 4.4 – Відображення розкладу модулів для групи

4.3 Процедура відображення розкладу модулів для викладача

Щоб виконати дану процедуру необхідно обрати тип користувача як «Викладач» та обрати викладача зі списку доступних викладачів (рис. 4.5). Після чого необхідно обрати дні тижня, де розкриється перелік модулів на обраний день тижня (рис. 4.6).

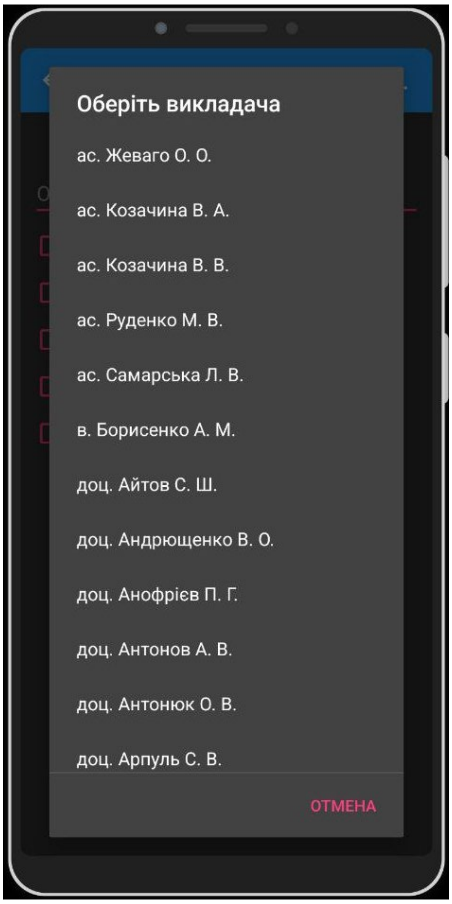



Рисунок 4.5 – Список вибору викладачів

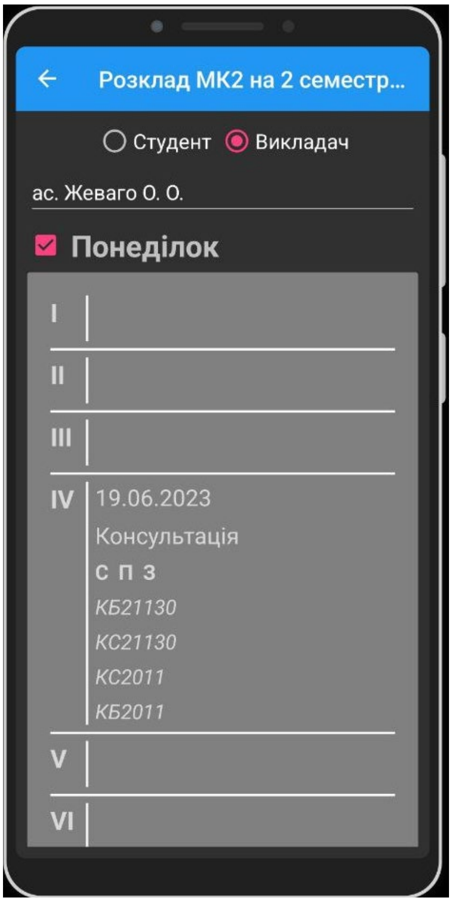



Рисунок 4.6 – Відображення розкладу модулів для викладача

4.4 Процедура відображення розкладу занять для студента

Щоб здійснити процедуру необхідно обрати зі списку доступних розкладів (рис.4.1) розклад занять, внаслідок цього відкриється сторінка перегляду розкладу (рис.4.2). На наступному етапі потрібно виконати відповідні дії з пункту 4.2 і в результаті необхідно обрати дні тижня, де розкриється перелік занять на обраний день тижня (рис. 4.7).

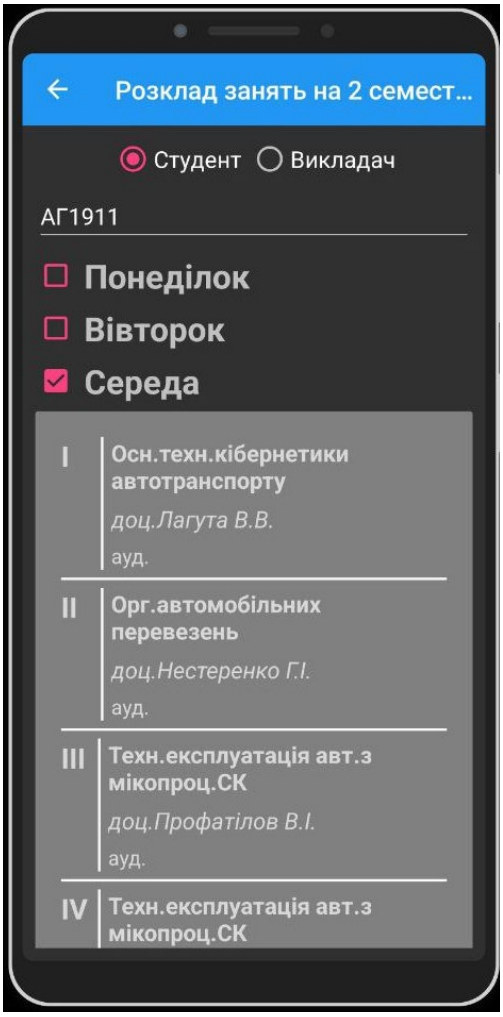



Рисунок 4.7 – Відображення розкладу занять для групи

4.5 Процедура відображення розкладу занять для викладача

Для проведення цієї процедури необхідно обрати тип користувач як «Викладач» та обрати викладача зі списку доступних викладачів. Після виконання цієї дії потрібно обрати дні тижня, де розкриється перелік модулів на обраний день тижня (рис. 4.8).

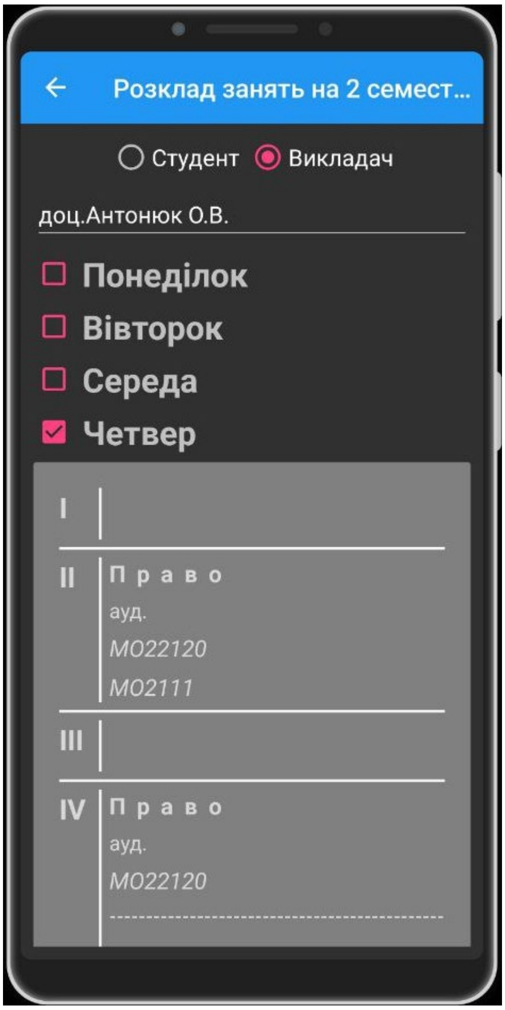



Рисунок 4.8 – Відображення розкладу занять для групи

# 5 ПОВІДОМЛЕННЯ КОРИСТУВАЧУ

У табл. 5.1 представлені повідомлення користувачу, що можуть з’явитися у процесі роботи програми.

Таблиця 5.1. Повідомлення, що з’являються в процесі роботи програми

| Текст повідомлення | Опис ситуації | Рекомендовані дії |
|---|---|---|
| Немає підключення до мережі Інтернет | Не вдалося завантажити список розкладів, бо мобільний пристрій немає підключення до мережі Інтернет | Підключити мобільний пристрій до WIFI або мобільної мережі |
| Увага:<br>Відображення даного розкладу знаходиться в розробці | Під час обрання розкладу, відображення якого ще не оброблено в програмі | Обрати інший розклад |
| Увага:<br>Не вдалося завантажити розклад | Якщо під час виконання програми пропав доступ до мережі Інтернет | Перевірити доступ до мережі Інтернет |

# 6 АВАРІЙНІ СИТУАЦІЇ

6.1 Дії у разі довгого завантаження розкладів

За тривалим часом завантаження списку розкладів необхідно перевірити якість підключення мобільного пристрою до мережі Інтернет. У разі необхідності перепідключитися або підключитися до альтернативної точки доступу до мережі.

6.2 Дії якщо мобільний додаток не запускається

Перевірити сумісність програмного забезпечення з мінімальним вимогами програмно додатку.

Оновити додаток до актуальної версії.

6.3 Дії в інших аварійних ситуаціях

В інших аварійних ситуаціях написати листа до технічної підтримки.