МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерних наук

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____Ю.П. Кондратенко
«____» _____2022 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

АВТОМАТИЗОВАНА СИСТЕМА СТВОРЕННЯ РОЗКЛАДУ ЗАНЯТЬ В УНІВЕРСИТЕТІ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.21810218

Виконав студент 4 курсу, групи 402
_____***А.В. Медвідь***
«___» червня 2022 р.

Керівник: канд. техн. наук, доцент
_____***М.В. Донченко***
«___» червня2022 р.

м. Миколаїв – 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____ Ю. П. Кондратенко
«____» _____20___р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Видано студенту групи 402 факультету комп'ютерних наук Медвідю Андрію Володимировичу.

1. Тема кваліфікаційної роботи «Автоматизована система створення розкладу занять в університеті».

Керівник роботи Донченко Михайло Васильович, канд. техн. наук, доцент.

Затв. наказом Ректора ЧНУ ім. Петра Могили від «___» ____ 20__ р. № _____

2. Строк представлення кваліфікаційної роботи студентом «__» ___ 20__р.

3. Очікуваний результат роботи: мобільний застосунок розкладу занять в університеті.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

- вибір інструментальних засобів створення застосунку;
- програмна реалізація мобільного додатка;

5. Перелік графічного матеріалу: сторінок – 80, таблиць – 3, рисунків – 53, посилань – 17, презентація.

6. Завдання до спеціальної частини: «Охорона праці при користуванні екранними пристроями».

7. Консультанти:

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охорони праці	Алексєєва А.А., канд. техн. наук, доцент	

Керівник роботи ___канд. техн. наук, доцент Донченко М.В._____
(наук. ступінь, вчене звання, прізвище та ініціали)

_____
(підпис)

Завдання прийнято до виконання _Медвідь А.В._____
(прізвище та ініціали)

_____
(підпис)

Дата видачі завдання « 07 » ____грудня____ 2021 р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: «Автоматизована система створення розкладу занять в університеті»

№	Найменування роботи	Початок	Закінчення	Примітки
1	Подання заяви на затвердження теми та керівників БКР	07.12.2021	07.12.2021	виконано
2	Отримання завдання на виконання БКР	07.12.2021	07.12.2021	виконано
3	Складання календарного плану роботи на весь період виконання БКР	08.12.2021	08.12.2021	виконано
4	Отримання завдання на переддипломну практику	23.05.2022	23.05.2022	виконано
5	Проходження переддипломної практики, збір та аналіз матеріалів до БКР	24.05.2022	02.06.2022	виконано
6	Розробка звіту з переддипломної практики	03.06.2022	05.06.2022	виконано
7	Виконання БКР: аналіз предметної сфери, вибір інструментальних засобів створення застосунку, розробка застосунку, тестування	27.02.2022	21.05.2022	виконано
8	Попередній захист БКР на засіданні комісії кафедри	30.05.2022	30.05.2022	виконано
9	Доробка та остаточне оформлення БКР	31.05.2022	24.06.2022	виконано
10	Подання БКР рецензенту	26.06.2022	26.06.2022	
11	Подання БКР, її електронної копії та інших документів (відгуку, рецензії) до захисту	26.06.2022	26.06.2022	
12	Захист БКР перед екзаменаційною комісією (ЕК)	29.06.2022	29.06.2022	

Розробив студент _____Медвідь А.В._____ _____
(прізвище та ініціали) *(підпис)*

Керівник роботи ___канд.техн.наук, доцент Донченко М.В.___ _____
(наук. ступінь, вчене звання, прізвище та ініціали) *(підпис)*

«____» _____ 202__ р.

АНОТАЦІЯ

бакалаврської кваліфікаційної роботи

студента групи 402 ЧНУ ім. Петра Могили

Медвідя Андрія Володимировича

Тема: «Автоматизована система створення розкладу занять в університеті»

Керівник: канд.техн.наук, доцент Донченко Михайло Васильович.

Об’єкт дослідження – інформаційні технології у повсякденному житті.

Предмет дослідження – мобільний застосунок розкладу занять в університеті.

Мета – створення мобільного додатка-сервісу розкладу занять для студентів, що дозволятиме переглядати поточний розклад занять, переглядати інформацію про розклад дзвінків та години роботи структур університету, знаходити викладачів з можливістю дізнатися контактні дані та переглядати мапу розташування корпусів.

У першому розділі розглядається 14. У другому – 32. У третьому – 57. В останньому спеціальному розділі було розглянуто нормативну базу охорони праці при користуванні екранними приладами.

У результаті виконання роботи було зроблено наступні висновки 72.

Сторінок – 80, таблиць – 3, рисунків – 53, посилань – 17, додатків – 0

Ключові слова: сфера мобільних застосунків-сервісів, мобільний застосунок розкладу занять в університеті.

ABSTRACT

for bachelor's qualification work

of a student of 402 group at Petro Mohyla Black Sea National University

Medvid Andrii Volodymyrovych

Topic: "Automated system for creating a schedule of classes at the university"

Supervisor: candidate of technical sciences, docent Donchenko Mykhailo Vasylovych.

The object of the research is information technologies in everyday life.

The subject of the research is a mobile application of the university schedule.

The goal is to create a mobile application-service of class schedules for students, which will allow you to view the current class schedule, view information about the schedule and hours of university structures, find teachers with the ability to find contact information and view the location map.

The first section presents 14. The second section describes 32. The third section specifies 57. In the last section the normative base of labor protection at use of screen devices was considered.

As the result of the performed work, conclusions 72

Pages – 80, tables – 3, figures – 53, links – 17, appendices – 0

Keywords: sphere of mobile applications-services, mobile application of the schedule of classes at the university.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

Держ. – державний

Див. – дивитись

ЛП – Львівська Політехніка

НТУ ХПІ – Національний технічний університет «Харківський політехнічний інститут»

Рис. – рисунок

СУБД – система управління базами даних

Та ін. – та інше(і)

AVD – Android Virtual Device

SQL – Structured query language

Пояснювальна записка

до кваліфікаційної роботи

на тему:

АВТОМАТИЗОВАНА СИСТЕМА РОЗКЛАДУ ЗАНЯТЬ В УНІВЕРСИТЕТІ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.21810218

Виконав: студент 4-го курсу, групи 402
_____*А.В. Медвідь*
(підпис, ініціали та прізвище)
«___» _____ 2022 р.

Керівник: *канд. техн. наук, доцент*_____
(наук. ступінь, вчене звання)
_____*М.В. Донченко*
(підпис, ініціали та прізвище)
«___» _____ 2022 р.

Миколаїв – 2022

ВСТУП

Інформаційні технології сьогодні є повсякденною складовою частиною нашого життя і доступ до них має переважна більшість людей. Смартфони супроводжують нас усюди, навіть у стінах університету. Тому прив'язати університетську складову до персонального смартфону є сучасною ідеєю. Студент в університеті кожного дня стикається з повсякденними потребами, такими, як дізнатися розклад занять на поточний чи наступний тиждень, яка наступна пара і в якій аудиторії вона буде проводитись, знайти викладача, або вийти з ним на зв'язок. Відповіді на всі ці питання можна об'єднати у мобільному додатку на смартфоні студента.

Використання мобільних додатків серед користувачів мобільних пристроїв стає все більш і більш популярним. Додатки-сервіси продовжують отримувати підтримку користувачів, тому що вони значно полегшують повсякденне життя. Тому розробка університетського сервісу для надання повсякденної необхідної інформації для студентів є гарною перспективою.

Мета роботи: Розробка мобільного застосунку розкладу занять в університеті.

Відповідно до мети виділені наступні **завдання дипломної роботи**:

– аналіз предметної області, аналогічного програмного забезпечення;

– обрання середовища та інструментів для розробки програмного забезпечення;

– розробка мобільного додатку розкладу занять в університеті.

1 АНАЛІЗ СФЕРИ МОБІЛЬНИХ ЗАСТОСУНКІВ-СЕРВІСІВ

1.1 Дослідження сфери мобільних застосунків-сервісів

Нині сучасні інформаційні технології не стоять на місці, а достатньо швидко розвиваються, паралельно з ними розвивається напрямок мобільних додатків. Їх впровадження сприяє покращенню якості спілкування, вирішення різноманітних завдань та покращення комунікативності.

Сьогодні існує безліч корисних застосунків-сервісів, але спираючись на локальні інтереси суспільства, можна виділити окремі, найбільш популярні застосунки серед українців.

Державний застосунок Дія

Один з найпопулярніших, а також перший представник державного застосунка у світі – Дія. Це збірник держ. документів та держ. послуг у вашому смартфоні. Кожна людина, яка володіє смартфоном має можливість переглядати свої документи (див. рис. 1.1), а також при необхідності пред’явити їх. Вони легітимні на тому ж рівні, як паперові.

Рисунок 1.1 – Вигляд документів у застосунку «Дія» [1]

Застосунок надає змогу отримувати деякі послуги онлайн (див. рис. 1.2). Наприклад, запис на заміну посвідчення водія, запис до листа очікування

вакцинації від COVID-19, отримання єПідтримки – фінансової допомоги для людей під час карантину та війни, та ін.

Рисунок 1.2 – Вигляд послуг у застосунку «Дія» [1]

Це яскравий приклад застосунку, який надає послуги для мільйонів користувачів та робить повсякденне життя легше.

Банківські застосунки

Також велику популярність мають банківські застосунки – «Приват24» (див. рис. 1.3) і трохи менш популярний «Монобанк» (див. рис. 1.4). Обидва додатка надають можливості переводити безготівкові гроші з карти на карту, відкладати їх, розраховуватись у безготівковий спосіб у місцях надання послуг. Тепер відмінності.

«Приват24» має послугу сплати комунальних платежів. Їх можна сплачувати моментально знаходячись вдома, не витрачаючи час на поїздку і черги.

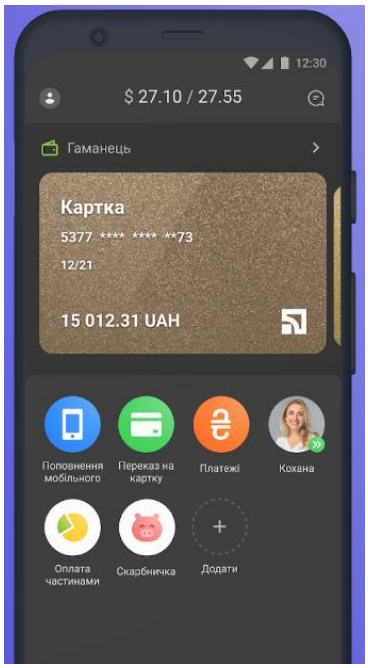

Рисунок 1.3 – Зовнішній вигляд застосунку «Приват24» [2]

«Монобанк» - це онлайн банк, на відміну від «Приват24», він не має фізичних відділень. Фізична картка отримується у різних місцях, залежно від партнерів банку. Це може бути поштове відділення, відділення оператора зв’язку та ін.

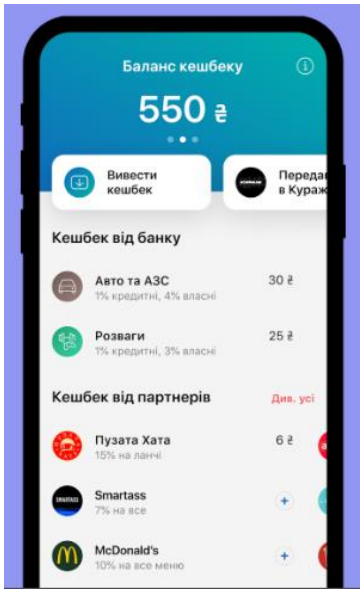

Рисунок 1.4 – Зовнішній вигляд застосунку «Монобанк» [3]

Ці банківські застосунки надзвичайно спрощують життя у фінансовій сфері для звичайних людей. Вони є одними з найпотужніших і

найпопулярніших застосунків-сервісів в Україні, також є іноземні аналоги, які також є популярними в інших країнах.

Застосунки доставок

Застосунки доставок можна розділити на декілька категорій. Перша – це пошта, друга – доставка їжі. Варто зауважити, що другий тип набув значної популярності в умовах карантину.

Найпопулярніший застосунок доставки пошти в Україні має однойменну назву «Нова пошта» (див. рис. 1.5). За допомогою цього застосунку можна відслідковувати свої доставки та відправлення, сплачувати доставку прямо у застосунку, щоб не витрачати час у відділенні пошти, знаходити відділення або поштомат та ін.

Рисунок 1.5 – Зовнішній вигляд застосунка «Нова пошта» [4]

Цей застосунок значно полегшує поштові операції, оптимізує час отримання та відправлення пошти як для працівників, так і для клієнтів, надає змогу слідкувати за доставкою.

Другий тип доставок – доставка їжі. Сьогодні в Україні популярні декілька організацій-доставок. Для аналізу обрано найпопулярніший додаток – «Glovo» (див. рис. 1.6). Додаток має невеликий, але досить корисний функціонал – замовлення доставки готової їжі, або замовлення продуктів

харчування. При замовленні можна обирати заклад або магазин, звідки отримати їжу або продукти.

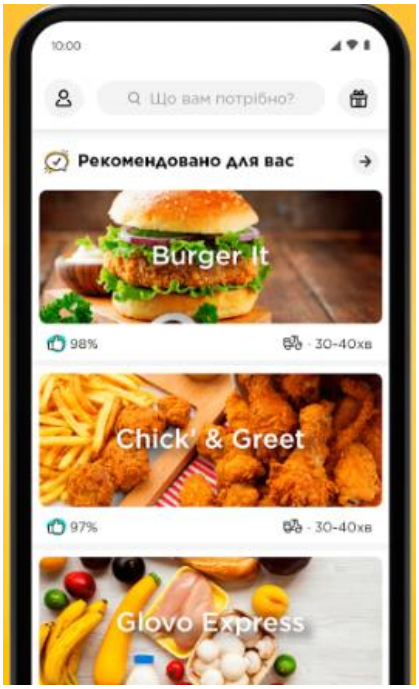

Рисунок 1.6 – Зовнішній вигляд застосунку «Glovo» [5]

Такі застосунки значно полегшують повсякденне життя у деяких ситуаціях, наприклад немає часу приготувати поїсти, або у ситуації з карантином, коли усі заклади харчування закриті і працюють тільки на виніс, а у магазини запускають обмежену кількість людей.

1.2 Аналіз існуючих застосунків для університетів

Освітня сфера та застосунки-розклади

Статистика говорить, що середньостатистичний студент університету віддає смартфону приблизно 8 годин 48 хвилин щодня [6]. Приблизно 30 хвилин кожен день студенти шукають розклад, який завантажують у таких форматах на сторінки, які ваш телефон не підтримує.

Сьогодні освітня сфера потребує власного типу застосунків які будуть полегшувати життя студентів та надавати необхідну інформацію студенту

Загалом у світі спостерігається тенденція до збільшення кількості університетів, які впроваджують електронний розклад у свій навчальний

процес щоб всі студенти завжди могли мати актуальну інформацію про своє навчання.

Розклад Львівської Політехніки

Застосунок «Розклад Львівської Політехніки» (див. рис 1.7) був розроблений для студентів Львівської Політехніки задля полегшення орієнтації в університеті та у розкладі занять.

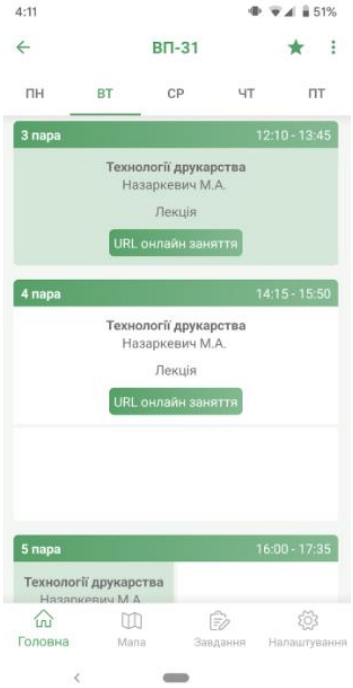

Рисунок 1.7 – Розклад занять у застосунку «Розклад ЛП» [7]

У застосунку є можливість завантажити наступні розклади:

- розклад занять для студентів;
- розклад екзаменів для студентів;
- розклад занять для викладачів;
- розклад екзаменів для викладачів;
- розклад занять для аспірантів;
- розклад екзаменів для аспірантів;
- розклад занять для студентів-заочників;
- розклад занять для аспірантів-заочників;
- розклад занять вибіркових дисциплін.

Також у застосунку є наступні можливості:

– зберігання розкладу для перегляду його без підключення до інтернету;
– обрахування середнього балу;
– перегляд розміщення корпусів на мапі для вибраної пари;
– вибір світлої або темної теми інтерфейсу і вибір кольорового акценту теми;
– відображення розкладу на поточний день у віджеті;
– мапа з позначеними навчальними корпусами;
– створення завдань.

Застосунок Timetable

Застосунок «Timetable» (див. рис. 1.8) має приємний зовнішній вигляд та надає можливість власноруч створювати розклад занять, незалежно від закладу освіти, чи то школа, чи університет та має наступні особливості:

– синхронізація інформації між усіма особистими пристроями;
– оптимізація для смартфонів та планшетів;
– темна та світла теми;
– легке зберігання особистих занять, канікул та завдань;
– перегляд розкладу у вигляді списку та у вигляді таблиці;
– пошук у розкладі та у завданнях;
– можливе включення другого, третього та четвертого навчального тижня;
– віджети 4x1 для розкладу та завдань;
– змінний (мін 2x2) віджет для розкладу та завдань;
– віджети для екрану блокування (Android 4.2);
– повідомлення про заняття та завдання на завтра;
– автоматичний беззвучний режим під час занять;
– розширення Dashclock.

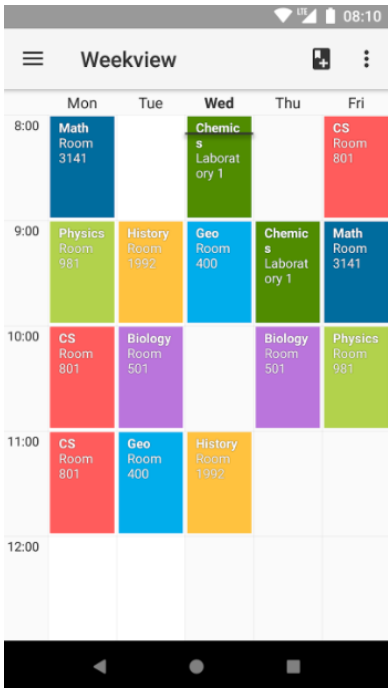

Рисунок 1.8 – Розклад занять застосунку «Timetable» [8]

Застосунок «Розклад занять НТУ ХПІ»

Застосунок «Розклад занять НТУ ХПІ» (див. рис. 1.9) розроблений студентом Харківського політехнічного інституту для університету.

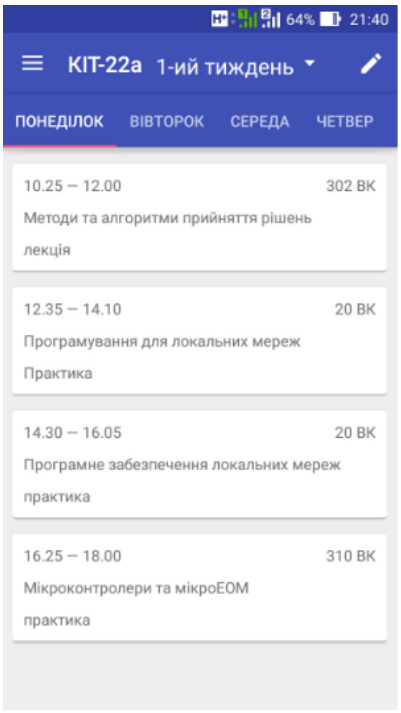

Рисунок 1.9 – Розклад занять застосунку «Розклад занять НТУ ХПІ» [9]

Застосунок має наступні можливості:

– перегляд розкладу занять обраної групи;

– створення завдань;
– перегляд мапи корпусів;
– перегляд викладачів;

1.3 Постановка задачі

Для меншої витрати часу та підтримки розвитку європейських тенденцій було вирішено створити мобільний застосунок розкладу занять в університеті.

До самого додатка було висунуто такі вимоги: наявність розкладу для всіх курсів та спеціальностей університету, пошук вчителя з його контактами для зв’язку, перегляд інформації про години роботи різних структур університету та перегляд мапи університету.

Висновки до розділу 1

У ході дослідження мобільних застосунків-сервісів було визначено, що використання мобільних застосунків серед користувачів мобільних пристроїв стає все більш і більш популярним. Застосунки-сервіси продовжують отримувати підтримку користувачів, тому що вони значно полегшують повсякденне життя. На зараз серед застосунків-сервісів велику популярність мають банківські застосунки, застосунки доставки, та застосунки надання різних послуг, починаючи з державних, закінчуючи послуги магазинів та організацій.

Також було проаналізовано освітню сферу, а саме університетську сферу та її складову застосунків для студентів. Було розглянуто декілька мобільних застосунків розкладу занять. У ході аналізу було зазначено основний функціонал застосунків:

– перегляд розкладу занять;
– функція пошуку викладача;
– перегляд інформації про години роботи структур університету;
– перегляд мапи розташування корпусів університету;

2 ВИБІР ПЛАТФОРМИ, СЕРЕДОВИЩА ТА ІНСТРУМЕНТІВ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

До самого додатка було висунуто такі вимоги: наявність розкладу для всіх курсів та спеціальностей університету, реалізація пошуку поточної аудиторії, наступної пари та пошук вчителя з його контактами для зв’язку.

Застосунок розроблявся для смартфонів на базі Android в середовищі Android Studio за допомогою мови Java, база даних реалізовувалася в СУБД (Система управління базами даних) MySQL.

2.1 Платформа Android

Android заповнює більшу частину ринку, наприклад, якщо взяти статистичні дані користувачів смартфонів в Україні за останній рік (див. рис. 2.1), то можна побачити, що Android займає майже 82% ринку [10]. Тому для більшого охоплення аудиторії було обрано саме Android, як платформу для розробки додатка.

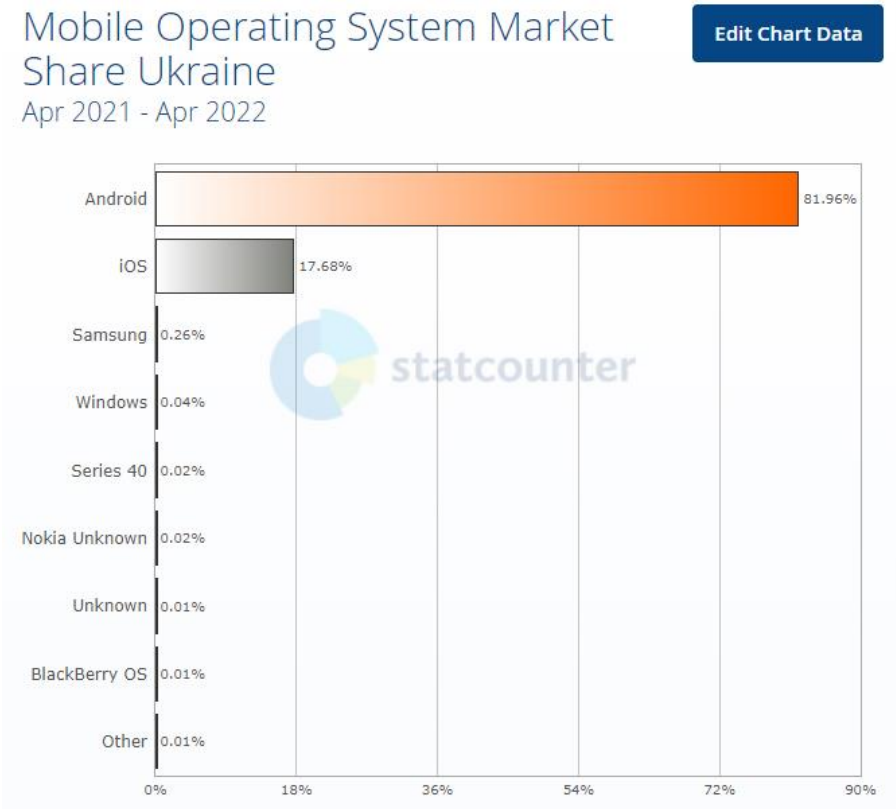

Рисунок 2.1 – Статистичні дані операційних систем користувачів смартфонів

Про Android

Android — це операційна система з відкритим кодом і на базі Linux для мобільних пристроїв, таких як смартфони та планшетні комп'ютери. Android був розроблений Open Handset Alliance на чолі з Google та іншими компаніями.

Android пропонує уніфікований підхід до розробки додатків для мобільних пристроїв, що означає, що розробникам потрібно розробляти лише для Android, а їхні програми повинні мати можливість працювати на різних пристроях на базі Android.

Перша бета-версія Android Software Development Kit (SDK) була випущена Google у 2007 році, а перша комерційна версія Android 1.0 була випущена у вересні 2008 року.

27 червня 2012 року на конференції Google I/O компанія Google анонсувала наступну версію Android 4.1 Jelly Bean. Jelly Bean — це поступове оновлення, основною метою якого є покращення інтерфейсу користувача, як з точки зору функціональності, так і продуктивності.

Вихідний код для Android доступний під безкоштовними ліцензіями на програмне забезпечення з відкритим кодом. Google публікує більшу частину коду під ліцензією Apache версії 2.0, а решту, зміни ядра Linux, під Загальною публічною ліцензією GNU версії 2.

Особливості Android

Android — це потужна операційна система, яка конкурує з Apple 4GS і підтримує чудові функції. Деякі з них перераховані нижче:

– гарний інтерфейс користувача – основний екран ОС Android забезпечує гарний та інтуїтивно зрозумілий інтерфейс користувача.

– підключення - GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, NFC і WiMAX;

– зберігання – SQLite, легка реляційна база даних, використовується для цілей зберігання даних;

– підтримка медіа – H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF і BMP;

– обмін повідомленнями - SMS і MMS;

– веб-браузер, заснований на механізмі компонування WebKit з відкритим вихідним кодом у поєднанні з механізмом JavaScript V8 Chrome, який підтримує HTML5 і CSS3;

– мультитач, Android має вбудовану підтримку мультитач, яка спочатку була доступна в таких телефонах, як HTC Hero;

– багатозадачність, користувач може переходити від одного завдання до іншого, і в той же час різні програми можуть працювати одночасно;

– віджети зі зміною розміру. Розмір віджетів можна змінювати, тому користувачі можуть розширювати їх, щоб показати більше вмісту, або зменшувати їх, щоб заощадити місце;

– багатомовність, підтримує односторонній та двонаправлений текст;

– Google Cloud Messaging, це служба, яка дозволяє розробникам надсилати короткі дані повідомлень своїм користувачам на пристроях Android, не потребуючи власного рішення для синхронізації;

– Wi-Fi Direct, технологія, яка дозволяє програмам виявляти та підключатися безпосередньо через однорангове з’єднання з високою пропускною здатністю;

– Android Beam, популярна технологія на основі NFC, яка дозволяє користувачам миттєво ділитися, просто доторкнувшись до двох телефонів із підтримкою NFC.

Програми для Android

Програми для Android зазвичай розробляються мовою Java за допомогою Android Software Development Kit.

Після розробки програми для Android можна легко упаковувати та розпродати через такі магазини, як Google Play, SlideME, Opera Mobile Store, Mobango, F-droid та Amazon Appstore.

Android працює на сотні мільйонів мобільних пристроїв у більш ніж 190 країнах світу. Це найбільша встановлена база серед будь-яких мобільних платформ і швидко зростає. Щодня в усьому світі активується понад 1 мільйон нових пристроїв Android.

Рівень API

Рівень API — це ціле значення, яке однозначно ідентифікує версію API фреймворку, яку пропонує версія платформи Android. API рівні наведені нижче.

Таблиця 2.1 – Рівні API

Platform Version	API Level	VERSION_CODE
Android 6.0	23	MARSHMALLOW
Android 5.1	22	LOLLIPOP_MR1
Android 5.0	21	LOLLIPOP
Android 4.4W	20	KITKAT_WATCH
Android 4.4	19	KITKAT
Android 4.3	18	JELLY_BEAN_MR2
Android 4.2, 4.2.2	17	JELLY_BEAN_MR1
Android 4.1, 4.1.1	16	JELLY_BEAN
Android 4.0.3, 4.0.4	15	ICE_CREAM_SANDWICH_MR1
Android 4.0, 4.0.1, 4.0.2	14	ICE_CREAM_SANDWICH
Android 3.2	13	HONEYCOMB_MR2
Android 3.1.x	12	HONEYCOMB_MR1
Android 3.0.x	11	HONEYCOMB
Android 2.3.4, 2.3.3	10	GINGERBREAD_MR1

Продовження таблиці 2.1

Platform Version	API Level	VERSION_CODE
Android 2.3.2, 2.3.1, 2.3	9	GINGERBREAD
Android 2.2.x	8	FROYO
Android 2.1.x	7	ECLAIR_MR1
Android 2.0.1	6	ECLAIR_0_1
Android 2.0	5	ECLAIR
Android 1.6	4	DONUT
Android 1.5	3	CUPCAKE
Android 1.1	2	BASE_1_1
Android 1.0	1	BASE

2.2 Середовище розробки Android Studio

Android Studio — офіційна IDE для розробки додатків Android, заснована на IntelliJ IDEA. Крім можливостей, які ви очікуєте від IntelliJ, Android Studio пропонує:

– гнучка система збірки на основі Gradle;

– варіанти збірки та створення кількох файлів apk;

– шаблони коду, які допоможуть вам створити загальні функції програми;

– розширений редактор макетів із підтримкою редагування теми перетягуванням;

– інструменти Lint для виявлення проблем продуктивності, зручності використання, сумісності версій та інших проблем;

– ProGuard і можливості підписання додатків;

– вбудована підтримка Google Cloud Platform, що спрощує інтеграцію Google Cloud Messaging і App Engine.

Структура проекту та файлу

Android Project View

За замовчуванням Android Studio відображає файли вашого профілю в поданні проекту Android. У цьому поданні показано згладжену версію структури вашого проекту, яка забезпечує швидкий доступ до ключових вихідних файлів проектів Android і допомагає вам працювати з новою системою збірки на основі Gradle. Перегляд проекту Android (див. рис 2.2):

– групує файли збірки для всіх модулів на верхньому рівні ієрархії проекту;

– показує найважливіші вихідні каталоги на верхньому рівні ієрархії модулів;

– групує всі файли маніфесту для кожного модуля;

– показує файли ресурсів з усіх вихідних наборів Gradle;

– групує файли ресурсів для різних мов, орієнтацій та типів екранів в одній групі для кожного типу ресурсу.

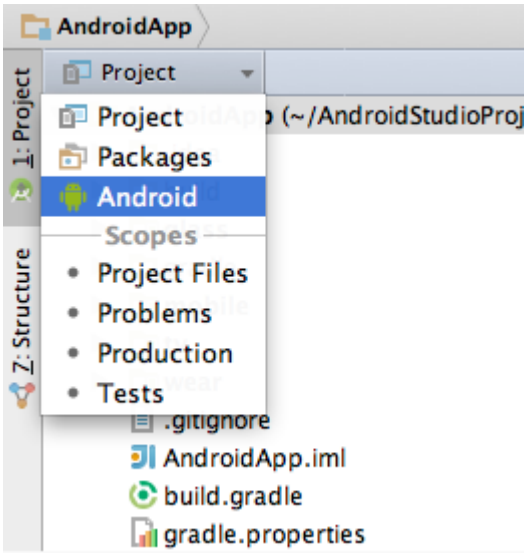

Рисунок 2.2 – Вигляд Android Project View

Подання проекту Android показує всі файли збірки на верхньому рівні ієрархії проекту в скриптах Gradle. Кожен модуль проекту відображається як папка на верхньому рівні ієрархії проекту і містить ці три елементи на верхньому рівні:

– java/ - вихідні файли для модуля;
– manifests/ - файли маніфесту для модуля;
– res/ - файли ресурсів для модуля.

Нова структура проекту та каталогу

Кожен екземпляр Android Studio містить проект (див. рис. 2.3) з одним або кількома модулями програми. Кожна папка модуля програми містить повні вихідні набори для цього модуля, включаючи каталоги src/main і src/androidTest, ресурси, файл збірки та маніфест Android.

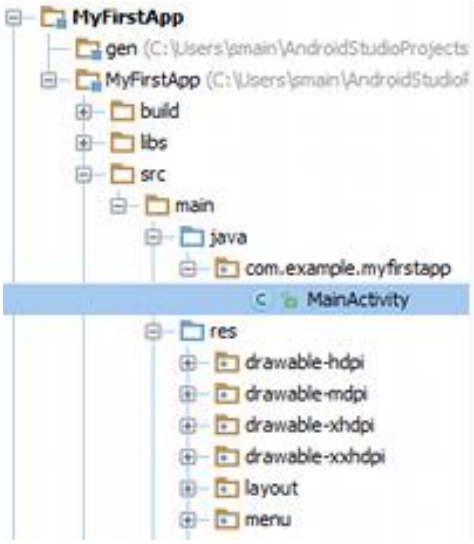

Рисунок 2.3 – Структура проекту Android Studio

Налагодження та продуктивність

Android Virtual Device (AVD) Manager

AVD Manager має оновлені екрани з посиланнями, це допомагає обрати найпопулярніші конфігурації пристрою, розміри екрана та роздільну здатність для попереднього перегляду ваших програм.

AVD Manager постачається з емуляторами для пристроїв Nexus 6 і Nexus 9, а також підтримує створення користувацьких обкладинок пристроїв Android на основі конкретних властивостей емулятора та призначення цих скінів до профілів обладнання. Android Studio встановлює прискорювач емулятора Intel® x86 Hardware Accelerated Execution Manager (HAXM) і створює емулятор за замовчуванням для швидкого прототипування програми.

Монітор пам'яті

Android Studio забезпечує перегляд монітора пам’яті (див. рис. 2.4), для полегшеного відстежування використання пам’яті програмою, щоб знаходити звільнені об’єкти, виявляти витоки пам’яті та відстежувати обсяг пам’яті, який використовує підключений пристрій.

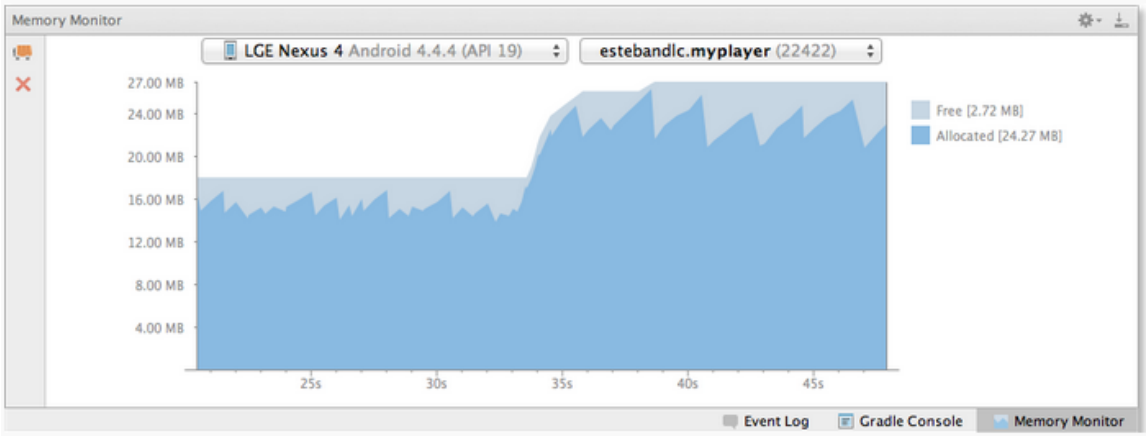

Рисунок 2.4 – Монітор пам’яті

Динамічний попередній перегляд макету

Android Studio дозволяє працювати з макетами як у режимі перегляду дизайну (див. рис. 2.5), так і в режимі перегляду тексту (див. рис. 2.6).

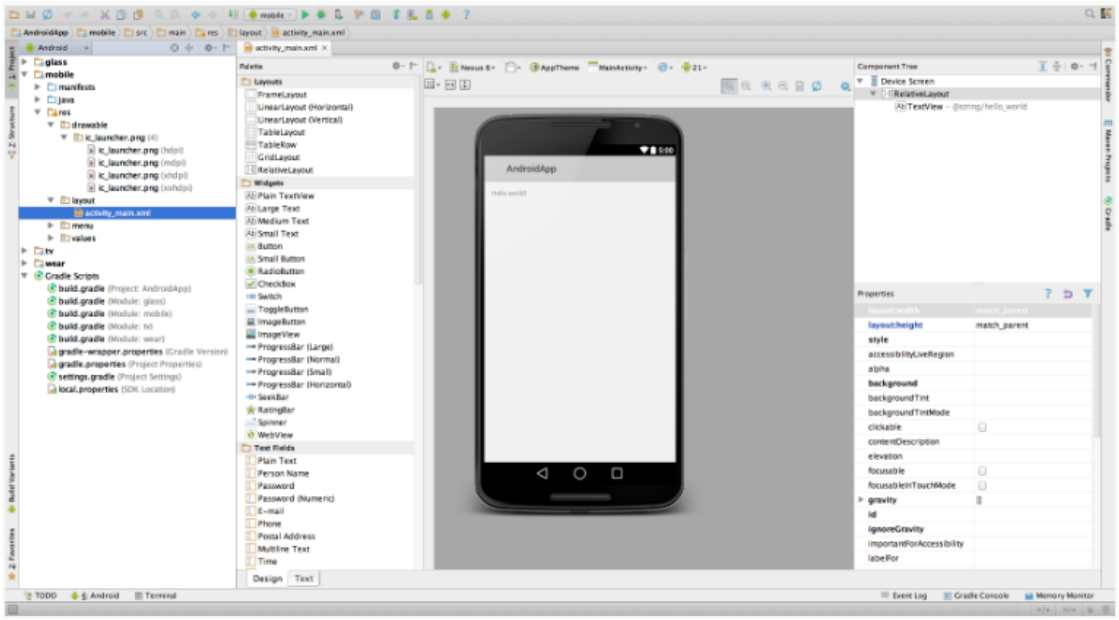

Рисунок 2.5 – Режим перегляду дизайну

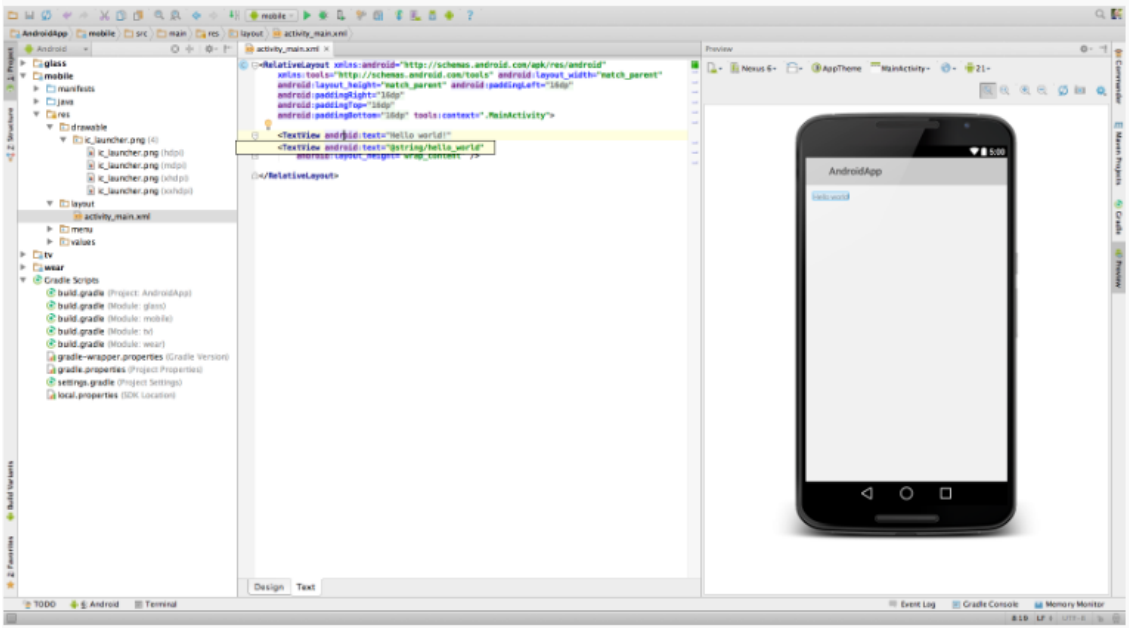

Рисунок 2.6 – Режим перегляду тексту

Журнал повідомлень

При створенні та запуску програми за допомогою Android Studio, користувач має можливість переглядати повідомлення журналу adb і пристроїв (logcat) на панелі DDMS.

2.3 Мова програмування Java

Мова програмування Java спочатку була розроблена Sun Microsystems, ініційована Джеймсом Гослінгом і випущена в 1995 році як основний компонент платформи Java Sun Microsystems (Java 1.0 [J2SE]).

Останнім випуском стандартної версії Java є Java SE 8. З розвитком Java та її широкою популярністю було створено безліч конфігурацій, щоб відповідати різним типам платформ. Наприклад: J2EE для корпоративних програм, J2ME для мобільних додатків.

Нові версії J2 були перейменовані на Java SE, Java EE та Java ME відповідно. Java гарантовано писати один раз, запускати будь-де.

Java – це:

– об’єктно-орієнтована мова програмування. У Java все є об’єктом. Java можна легко розширити, оскільки вона заснована на об'єктній моделі;

– незалежна від платформи. На відміну від багатьох інших мов програмування, включаючи C і C++, коли компілюється Java, вона не компілюється на платформу, а в незалежний від платформи байт-код. Цей байт-код поширюється в Інтернеті та інтерпретується віртуальною машиною (JVM) на будь-якій платформі, на якій він виконується;

– проста мова. Java розроблена так, щоб її легко вивчати. Якщо ви розумієте основну концепцію ООП Java, її буде легко освоїти;

– безпечна. Завдяки функції безпеки Java вона дозволяє розробляти системи без вірусів і несанкціонованого доступу. Методи автентифікації засновані на шифруванні з відкритим ключем;

– архітектурно-нейтральна мова. Компілятор Java генерує архітектурно-нейтральний формат об’єктного файлу, який робить скомпільований код виконуваним на багатьох процесорах, з наявністю системи середовища виконання Java;

– переносна мова. Будучи нейтральним до архітектури та не має аспектів специфікації, що залежать від реалізації, робить Java портативною. Компілятор на Java написаний на ANSI C з чіткою межею переносимості, яка є підмножиною POSIX;

– надійна мова. Java докладає зусиль, щоб усунути ситуації, схильні до помилок, наголошуючи головним чином на перевірці помилок під час компіляції та перевірці під час виконання;

– багатопотоковість. За допомогою багатопотокової функції Java можна писати програми, які можуть виконувати багато завдань одночасно. Ця особливість дизайну дозволяє розробникам створювати інтерактивні програми, які можуть працювати безперебійно;

– мова, що інтерпретується. Байт-код Java на льоту транслюється у власні машинні інструкції і ніде не зберігається. Процес розробки є більш швидким і аналітичним, оскільки зв'язування є поетапним і легким процесом;

– висока продуктивність. За допомогою компіляторів Just-In-Time Java забезпечує високу продуктивність;

– розподіленість. Java розроблена для розподіленого середовища Інтернету;

– динамічність. Java вважається більш динамічною, ніж C або C++, оскільки вона розроблена для адаптації до середовища, що розвивається. Програми Java можуть нести велику кількість інформації під час виконання, яку можна використовувати для перевірки та дозволу доступу до об'єктів під час виконання.

2.4 База даних MySQL

MySQL, найпопулярніша система керування базами даних SQL з відкритим вихідним кодом, розроблена, поширюється та підтримується Oracle Corporation.

База даних — це структурована колекція даних. Це може бути що завгодно, від простого списку покупок до картинної галереї або величезної кількості інформації в корпоративній мережі. Для додавання, доступу та обробки даних, що зберігаються в комп'ютерній базі даних, потрібна система керування базами даних, наприклад MySQL Server. Оскільки комп'ютери дуже добре обробляють великі обсяги даних, системи керування базами даних відіграють центральну роль у обчислювальній роботі, як самостійні утиліти або як частина інших програм.

Бази даних MySQL є реляційними. Реляційна база даних зберігає дані в окремих таблицях, а не розміщує всі дані в одному великому сховищі. Структури бази даних організовані у фізичні файли, оптимізовані для

швидкості. Логічна модель з такими об’єктами, як бази даних, таблиці, уявлення, рядки та стовпці, пропонує гнучке середовище програмування. Ви встановлюєте правила, що регулюють відносини між різними полями даних, наприклад, один до одного, один до багатьох, унікальний, обов’язковий або необов’язковий, а також «вказівники» між різними таблицями. База даних дотримується цих правил, так що за допомогою добре розробленої бази даних ваша програма ніколи не бачить неузгоджених, повторюваних, сиріт, застарілих або відсутніх даних.

Частина SQL «MySQL» означає «Мова структурованих запитів». SQL є найпоширенішою стандартизованою мовою, яка використовується для доступу до баз даних. Залежно від середовища програмування ви можете вводити SQL безпосередньо (наприклад, для створення звітів), вбудовувати оператори SQL у код, написаний іншою мовою, або використовувати спеціальний мовний API, який приховує синтаксис SQL.

SQL визначається стандартом SQL ANSI/ISO. Стандарт SQL розвивається з 1986 року і існує кілька версій. У цьому посібнику «SQL-92» відноситься до стандарту, випущеному в 1992 році, «SQL:1999» відноситься до стандарту, випущеному в 1999 році, а «SQL:2003» відноситься до поточної версії стандарту. Ми використовуємо фразу «стандарт SQL» для позначення поточної версії стандарту SQL у будь-який час.

MySQL – це програмне забезпечення з відкритим вихідним кодом

Відкритий код означає, що будь-хто може використовувати та змінювати програмне забезпечення. Будь-хто може завантажити програмне забезпечення MySQL з Інтернету та використовувати його, нічого не сплачуючи. Якщо бажаєте, ви можете вивчити вихідний код і змінити його відповідно до ваших потреб. Програмне забезпечення MySQL використовує GPL (GNU General Public License), щоб визначити, що ви можете, а що не можете робити з програмним забезпеченням у різних ситуаціях.

Сервер баз даних MySQL дуже швидкий, надійний, масштабований і простий у використанні. MySQL Server може комфортно працювати на настільному комп’ютері або ноутбуці, поряд з іншими додатками, веб-серверами тощо, не вимагаючи практичної уваги або зовсім не вимагаючи. MySQL також може масштабуватися до кластерів машин, об'єднаних разом.

MySQL Server спочатку був розроблений для роботи з великими базами даних набагато швидше, ніж існуючі рішення, і вже кілька років успішно використовується у дуже вимогливих виробничих середовищах. Хоча MySQL Server постійно розвивається, сьогодні він пропонує багатий і корисний набір функцій. Завдяки можливості підключення, швидкості та безпеки MySQL Server дуже підходить для доступу до баз даних в Інтернеті.

MySQL Server працює в клієнт/сервер або вбудованих системах. Програмне забезпечення баз даних MySQL — це система клієнт/сервер, яка складається з багатопотокового SQL-сервера, який підтримує різні серверні системи, кількох різних клієнтських програм і бібліотек, інструментів адміністрування та широкого спектру інтерфейсів прикладного програмування (API).

MySQL Server також має вбудовану багатопотокову бібліотеку, яку можна підключити до своєї програми, щоб отримати менший, швидший і простий у керуванні окремий продукт.

Головні переваги MySQL

Внутрішнє оздоблення та портативність:

– написано на C і C++;

– перевірено за допомогою широкого спектру різних компіляторів;

– працює на багатьох різних платформах;

– для портативності налаштовано за допомогою CMake;

– використовує багаторівневий дизайн сервера з незалежними модулями;

– розроблено для повної багатопотокової роботи з використанням потоків ядра, щоб легко використовувати декілька ЦП, якщо вони доступні;

– забезпечує транзакційні та нетранзакційні механізми зберігання даних;

– сикористовує дуже швидкі дискові таблиці B-дерева (MyISAM) зі стисненням індексів;

– створено для відносно легкого додавання інших механізмів зберігання даних;

– використовує дуже швидку систему розподілу пам'яті на основі потоків;

– виконує дуже швидкі об’єднання за допомогою оптимізованого з’єднання вкладеного циклу;

– реалізує хеш-таблиці в пам’яті, які використовуються як тимчасові таблиці;

– реалізує функції SQL, використовуючи високооптимізовану бібліотеку класів, яка має бути максимально швидкою. Зазвичай після ініціалізації запиту виділення пам’яті взагалі не відбувається;

– надає сервер як окрему програму для використання в мережевому середовищі клієнт/сервер.

Безпека:

– система привілеїв і паролів, яка є дуже гнучкою та безпечною, а також забезпечує перевірку на хості;

– захист паролем шляхом шифрування всього трафіку паролів при підключенні до сервера.

Типи даних:

– багато типів даних: цілі числа зі знаком/без знаку довжиною 1, 2, 3, 4 і 8 байтів, FLOAT, DOUBLE, CHAR, VARCHAR, BINARY, VARBINARY, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP,

YEAR, SET, ENUM , і просторові типи OpenGIS. Див. Розділ 11, Типи даних;

– типи рядків фіксованої та змінної довжини.

Вирази та функції:

– повна підтримка операторів і функцій у списку SELECT і речення WHERE запитів;

– повна підтримка пропозицій SQL GROUP BY і ORDER BY. Підтримка групових функцій (COUNT(), AVG(), STD(), SUM(), MAX(), MIN() і GROUP_CONCAT());

– підтримка LEFT OUTER JOIN і RIGHT OUTER JOIN зі стандартним синтаксисом SQL і ODBC;

– підтримка псевдонімів у таблицях і стовпцях, як того вимагає стандартний SQL;

– підтримка DELETE, INSERT, REPLACE та UPDATE, щоб повернути кількість рядків, які були змінені (зачепили), або повернути кількість рядків, які збігаються, замість цього, установивши прапор під час підключення до сервера;

– підтримка специфічних для MySQL операторів SHOW, які отримують інформацію про бази даних, механізми зберігання даних, таблиці та індекси. Підтримка бази даних INFORMATION_SCHEMA, реалізованої за стандартом SQL;

– оператор EXPLAIN, щоб показати, як оптимізатор вирішує запит.

– незалежність імен функцій від назв таблиць або стовпців. Наприклад, ABS є допустимою назвою стовпця. Єдине обмеження полягає в тому, що для виклику функції не допускаються пробіли між іменем функції та символом «(», що йде за ним. Див. Розділ 9.3, «Ключові слова та зарезервовані слова»;

– можливість посилатися на таблиці з різних баз даних в одному операторі.

Безпека:

– дуже гнучка та безпечна система привілеїв і паролів, що дозволяє перевіряти на хості;

– захист паролем шляхом шифрування всього трафіку паролів при підключенні до сервера.

Масштабованість та обмеження

– підтримка великих баз даних. MySQL Server використовується у базах даних, які містять 50 мільйонів записів. Також є користувачі, які використовують MySQL Server з 200 000 таблиць і близько 5 000 000 000 рядків;

– підтримка до 64 індексів на таблицю. Кожен індекс може складатися з 1 до 16 стовпців або частин стовпців. Максимальна ширина індексу для таблиць InnoDB становить 767 байт або 3072 байти. Див. Розділ 15.22, «Обмеження InnoDB». Максимальна ширина індексу для таблиць MyISAM становить 1000 байт. Див. Розділ 16.2, «Механізм зберігання MyISAM». Індекс може використовувати префікс стовпця для типів стовпців CHAR, VARCHAR, BLOB або TEXT.

Підключення:

1) Клієнти можуть підключатися до MySQL Server за допомогою кількох протоколів:

– Клієнти можуть підключатися за допомогою TCP/IP сокетів на будь-якій платформі.

– У системах Windows клієнти можуть підключатися за допомогою іменованих каналів, якщо сервер запускається з увімкненою системною змінною named_pipe. Сервери Windows також підтримують підключення до спільної пам’яті, якщо запущено з увімкненою системною змінною shared_memory. Клієнти можуть підключатися через спільну пам'ять, використовуючи параметр --protocol=memory.

– У системах Unix клієнти можуть підключатися за допомогою файлів сокетів домену Unix.

2) Клієнтські програми MySQL можуть бути написані багатьма мовами. Клієнтська бібліотека, написана на C, доступна для клієнтів, написаних на C або C++, або для будь-якої мови, яка забезпечує прив'язки C.

3) Доступні API для C, C++, Eiffel, Java, Perl, PHP, Python, Ruby і Tcl, що дозволяє писати клієнтам MySQL багатьма мовами. Див. Розділ 29, Конектори та API.

4) Інтерфейс Connector/ODBC (MyODBC) забезпечує підтримку MySQL для клієнтських програм, які використовують підключення ODBC (Open Database Connectivity). Наприклад, ви можете використовувати MS Access для підключення до сервера MySQL. Клієнти можна запускати в Windows або Unix. Доступний конектор/джерело ODBC. Підтримуються всі функції ODBC 2.5, як і багато інших. Перегляньте посібник розробника MySQL Connector/ODBC.

5) Інтерфейс Connector/J забезпечує підтримку MySQL для програм-клієнтів Java, які використовують з'єднання JDBC. Клієнти можна запускати в Windows або Unix. Доступний коннектор/джерело J. Див. Посібник розробника MySQL Connector/J 5.1.

6) MySQL Connector/NET дозволяє розробникам легко створювати програми .NET, які потребують безпечного високопродуктивного підключення даних до MySQL. Він реалізує необхідні інтерфейси ADO.NET та інтегрується в інструменти ADO.NET. Розробники можуть створювати програми, використовуючи власну мову .NET. MySQL Connector/NET — це повністю керований драйвер ADO.NET, написаний на 100% чистому C#. Перегляньте посібник розробника MySQL Connector/NET.

Локалізація

– сервер може надавати клієнтам повідомлення про помилки багатьма мовами;

– повна підтримка кількох різних наборів символів, включаючи latin1 (cp1252), німецьку, big5, ujis, декілька наборів символів Unicode тощо. Наприклад, скандинавські символи «å», «ä» та «ö» дозволені в назвах таблиць і стовпців;

– усі дані зберігаються у вибраному наборі символів;

– сортування та порівняння здійснюються відповідно до набору символів за замовчуванням та зіставлення. Це можна змінити під час запуску сервера MySQL. MySQL Server підтримує багато різних наборів символів, які можна вказати під час компіляції та виконання;

– часовий пояс сервера можна змінювати динамічно, а окремі клієнти можуть вказати свій власний часовий пояс.

Клієнти та інструменти:

– MySQL включає кілька клієнтських і допоміжних програм. До них відносяться як програми командного рядка, такі як mysqldump і mysqladmin, так і графічні програми, такі як MySQL Workbench;

– MySQL Server має вбудовану підтримку операторів SQL для перевірки, оптимізації та відновлення таблиць. Ці оператори доступні з командного рядка через клієнт mysqlcheck. MySQL також включає myisamchk, дуже швидку утиліту командного рядка для виконання цих операцій над таблицями MyISAM;

– програми MySQL можна викликати за допомогою --help або -? можливість отримати онлайн-допомогу.

Висновки до розділу 2

Для проекту було обрано платформу Android через її актуальність в Україні, так як за проаналізованими статистичними даними було виявлено, що

Android займає майже 82% ринку в Україні. Також Android має найбільш привабливі функції операційної системи, які можна використати під час розробки програми.

Середовищем розробки проекту було обрано Android Studio. Android Studio був спеціально розроблений для android-девелоперів та має усі необхідні функції для розробки проекту. Найголовнішими перевагами є зручна структура проекту та AVD Manager, який дуже допомагає під час розробки.

Обрана мова програмування – Java. Java вже багато років є одною з провідних мов програмування, яка має багато переваг. Незважаючи на стрімкий ріст мови Kotlin, Java поки зберігає інтенсивність використання, та на сьогодні більшість Android-застосунків пишуться саме на цій мові програмування.

Як серверну частину проекту, а саме структуроване зберігання даних, **було обрано базу даних MySQL**. Це база даних, яка рекомендує себе, як одна з найкращих серверних SQL платформ, яка може опрацьовувати великий потік даних і не потребує сильного догляду зі сторони розробника.

Розроблений мобільний додаток розкладу занять для університету – це застосунок для смартфонів на базі Android, розроблюваний в середовищі Android Studio за допомогою мови Java, із серверною базою даних MySQL.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЗОВНІШНІЙ ВИГЛЯД МОБІЛЬНОГО ЗАСТОСУНКУ РОЗКЛАДУ ЗАНЯТЬ В УНІВЕРСИТЕТІ

3.1 Дерево компонентів програми

Дерево компонентів мобільного застосунку створеного у середовищі розробки Android Studio – це структура головної папки проекту, яке наведено на рисунку нижче.

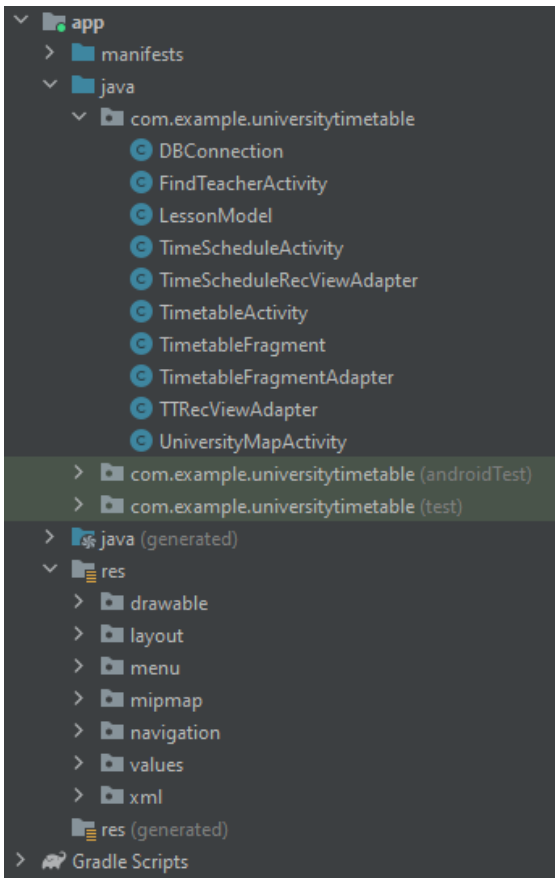

Рисунок 3.1 – Дерево компонентів проекту

У папці app/java/com.example.universitytimetable зберігаються Java класи всього проекту.

Серед них можна побачити класи з суфіксом «Activity» у кінці назви класу. Вони відповідають за створення «активностей» застосунку, які слугують зовнішньою складовою застосунка. Також в цих класах прописаний функціонал компонентів цих самих активностей.

Ще у цій папці є класи з суфіксом «Adapter». Це адаптери фрагментів та листових виглядів. Вони полегшують роботу з цими компонентами.

Також в папці знаходяться класи з даними та функціями, наприклад, клас «DBConnection» відповідає за підключення до бази даних, зберігає сталі дані, такі як посилання на базу даних, ім’я користувача бази даних та його пароль, і має в собі функцію опрацювання цих даних, а саме підключення до бази даних. Ще є клас «LessonModel». Він має підготовленні змінні, а також конструктор, геттери та сеттери для заповнення цих даних у створюваних об’єктах.

У папці app/res/layout (див. рис. 3.2) знаходяться макети активностей з приставкою «activity_», в них знаходиться усе зовнішнє представлення активності, розташування компонентів, їх стилі, кольори та розміри. Фрагменти визначаються приставкою «fragment», це зовнішнє представлення фрагментів активностей, мають такі ж властивості як макети активностей. Також в цій папці знаходяться деталі листів, це зовнішній вигляд компоненту листа. Ще в папці є зовнішні представлення спливаючих вікон, мають такі ж властивості як макети активностей.

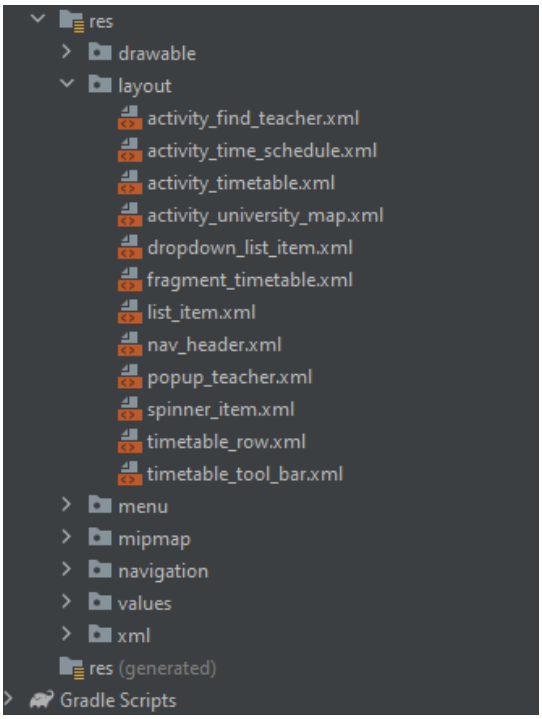

Рисунок 3.2 – Папка «layout»

У папці app/res/drawable (див. вис. 3.3) знаходяться xml файли, серед яких є іконки з приставкою «ic_», вони є зовнішніми складовими деяких компонентів, до яких додається іконка. Це векторні малюнки, які знаходяться у вбудованій бібліотеці з іконками. Також в цій папці знаходяться файли з приставкою «bg_» - це файли стилю деяких кастомізованих компонентів. Ще в папці знаходяться png файли світлин, які використовуються у застосунку, наприклад, chmnu_logo.png – це логотип Чорноморського Національного Університету, а university_map.png – це світлина мапи розташування корпусів університету.

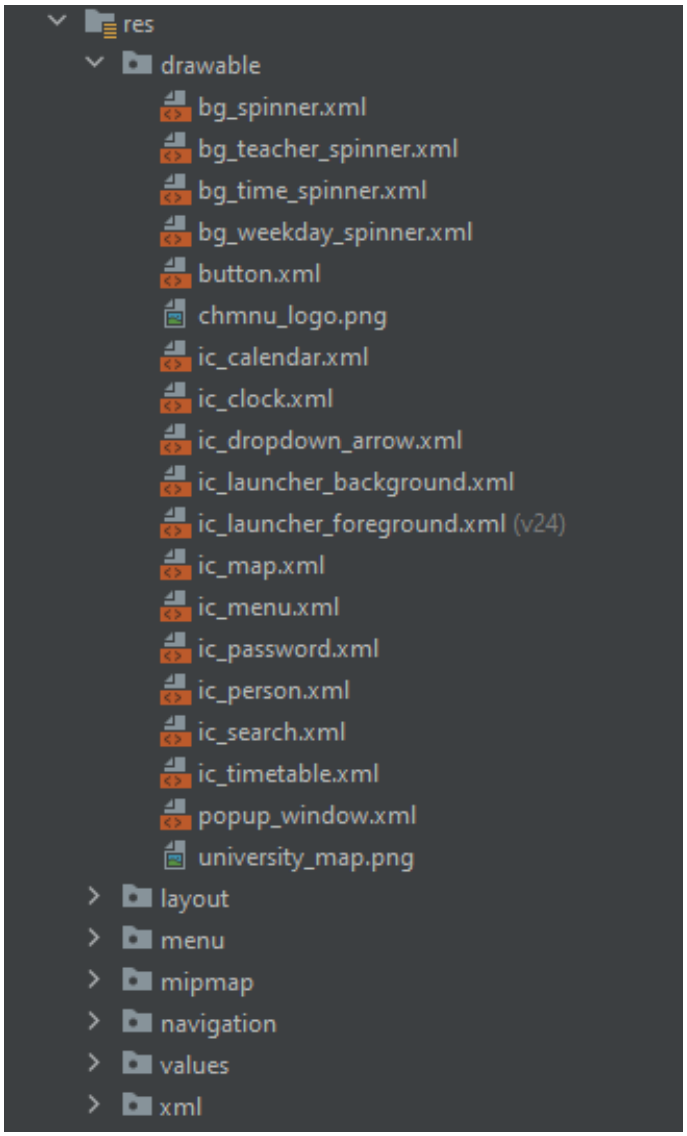

Рисунок 3.3 – Папка «drawable»

У папці app/res/menu (див. рис. 3.4) знаходиться xml файл side_menu.xml, який відповідає за розташування компонентів меню бокової панелі.

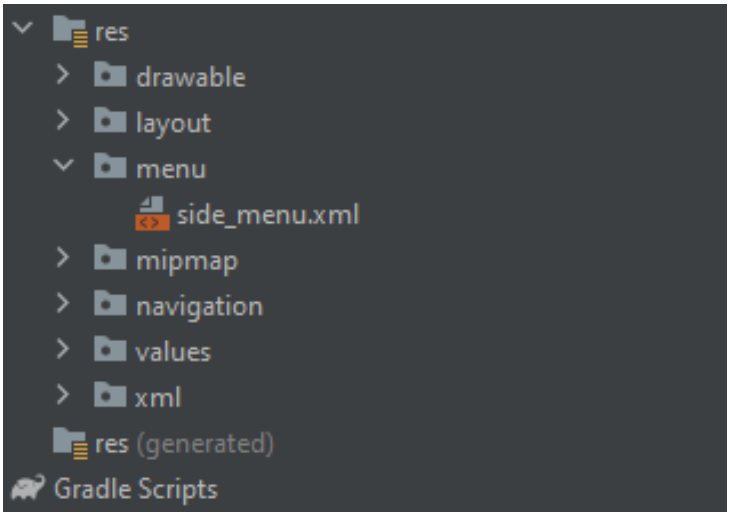

Рисунок 3.4 – Папка «menu»

У папці app/res/mipmap (див. рис. 3.5) знаходяться підпапки з png файлами іконки самого застосунка, яку можна побачити на робочому столі смартфону. Ці файли відповідають за зовнішній вигляд іконки застосунка при різному DPI (Dots per inch) екрану та при різних стилях іконок робочого столу.

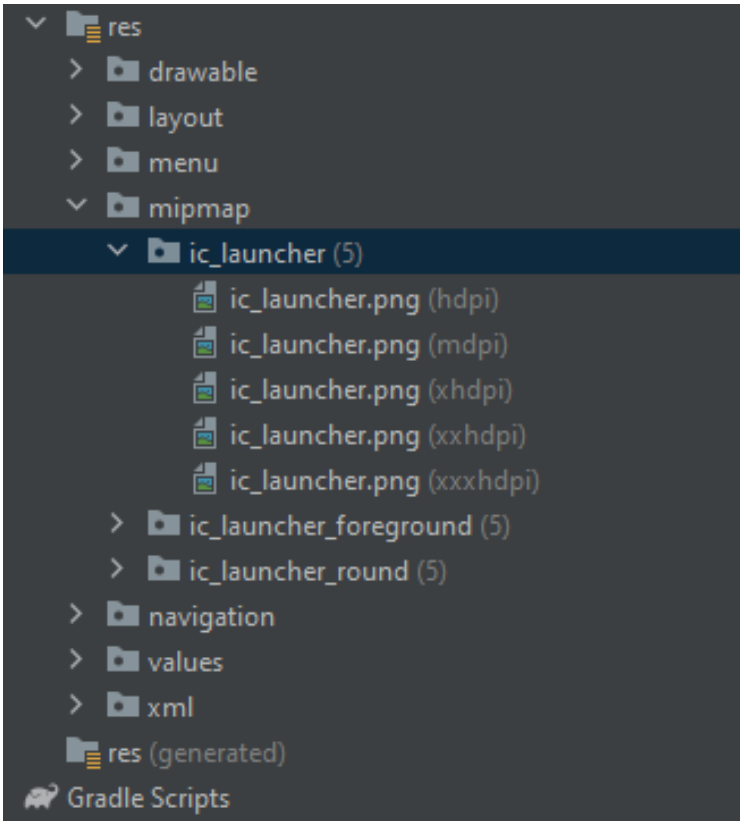

Рисунок 3.5 – Папка «mipmap»

У папці app/res/values (див. рис. 3.6) знаходяться xml файли colors, strings, та styles. Файл colors.xml відповідає за кольори, які використовуються і накладаються на компоненти застосунку. Файл strings.xml вміщує усі строкові представлення застосунку, а також масиви заготовлених даних для листових представлень. Файл styles.xml наповнений стилями, які використовуються у застосунку.

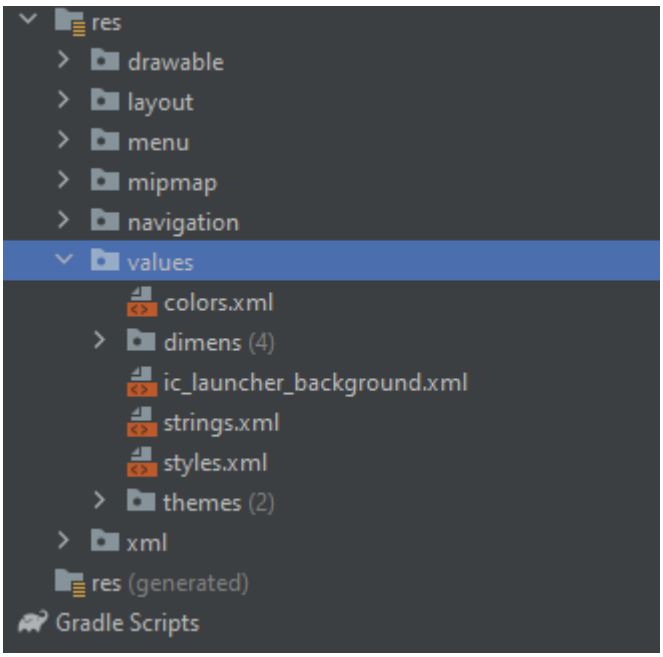

Рисунок 3.6 – Папка «values»

3.2 Користувацький інтерфейс та основний функціонал

При вході до застосунку з’являється головна сторінка – розклад занять (див. рис. 3.7). Зверху сторінки знаходиться панель інструментів. На панелі інструментів знаходяться кнопка бокової панелі та 2 спадні списки – список вибору групи і список вибору тижня. Знизу сторінки знаходиться панель вкладок з днями тижня. По центру знаходиться основна частина сторінки – розклад занять для обраної групи, обраного тижня та обраного дня тижня.

Подивитись розклад для іншого дня тижня можна змахнувши пальцем вліво або вправо по центру екрану, або обравши вкладку з необхідним днем тижня на панелі вкладок знизу.

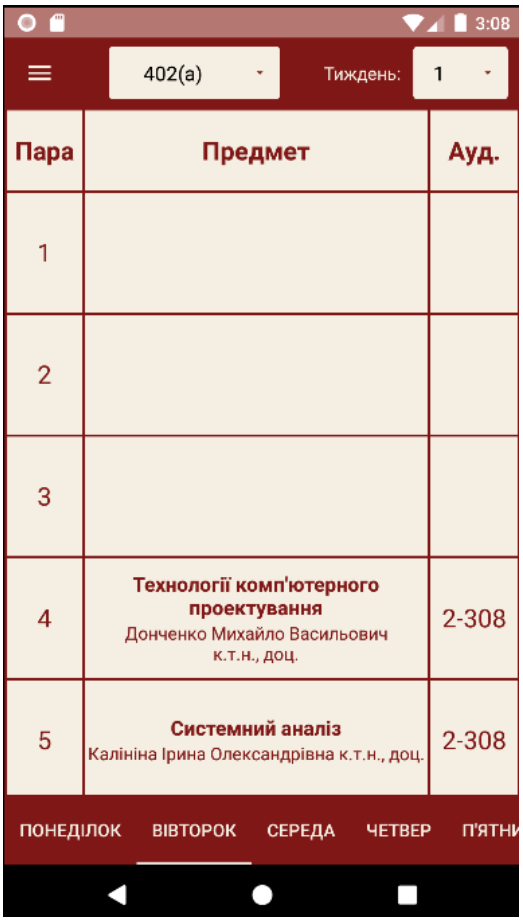

Рисунок 3.7 – Сторінка розкладу занять

При вході у застосунок автоматично обирається вкладка з поточним днем тижня (див. рис. 3.8).

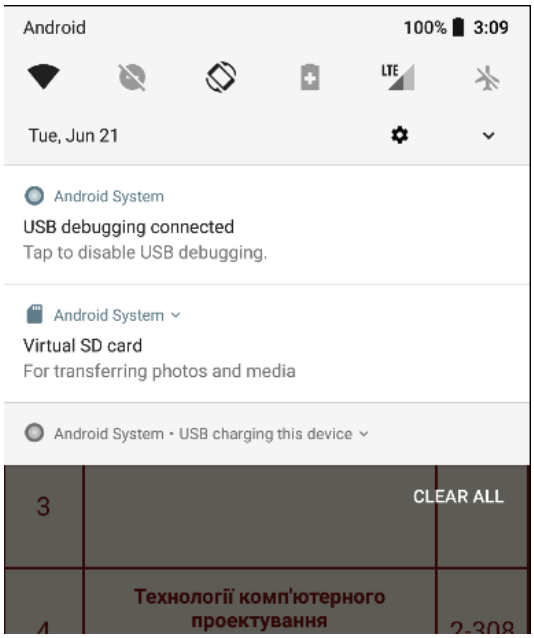

Рисунок 3.8 – Перевірка поточного дня тижня у спадній шторці

При виборі групи та тижня вибір користувача запам’ятовується. На рис. 3.9 було обрано групу 402(б) та тиждень 2, після чого був здійснений вихід із застосунку на робочий стіл (див. рис. 3.10) та звільнення пам’яті.

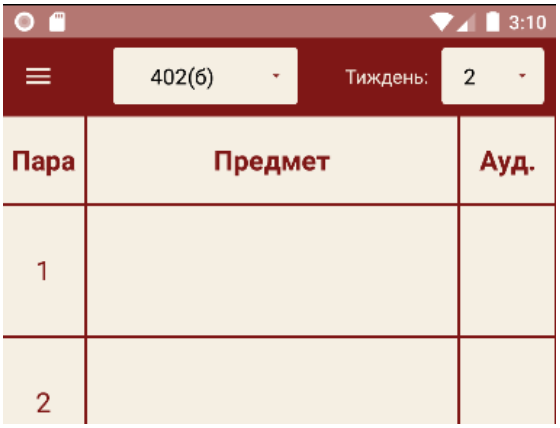

Рисунок 3.9 – Вибір групи 402(б) та тижня 2

На рис. 3.10 також можна побачити зовнішній вигляд іконки застосунку на робочому столі смартфону.

Рисунок 3.10 – Закриття застосунку та вихід на робочий стіл

Після повторного запуску застосунка (див. рис. 3.11) видно, що попередній вибір користувача було збережено та автоматично обрано.

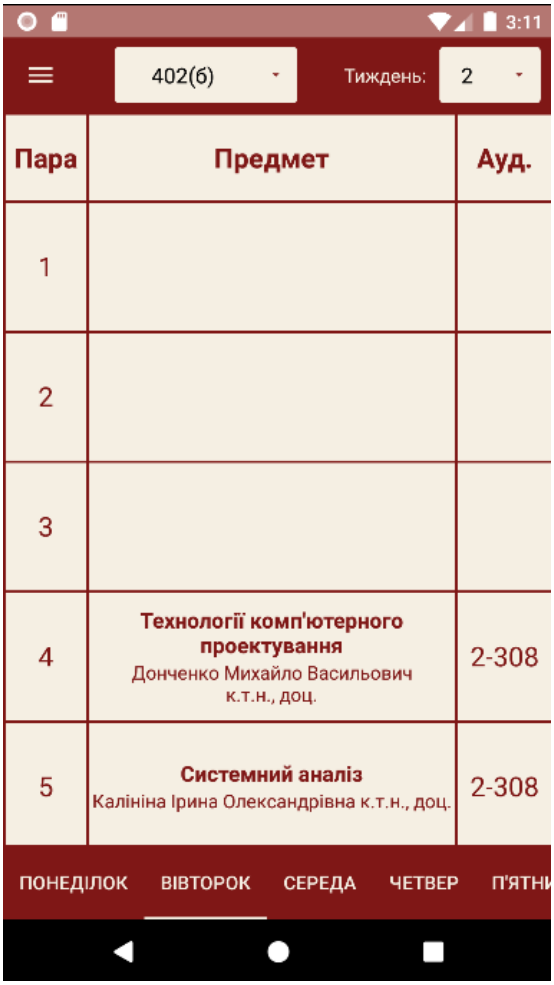

Рисунок 3.11 – Вигляд розкладу занять після повторного запуска застосунку

При натисканні на кнопку бокової панелі на панелі інструментів або змахнувши зліва направо у лівій частині екрану буде відкрита бокова панель застосунку (див. рис. 3.12). У верхній частині панелі знаходиться логотип Чорноморського Національного Університету імені П. Могили. Нижче знаходиться меню із вкладками. Вкладка «Розклад занять» перенаправляє користувача на головну сторінку з розкладом занять.

Рисунок 3.12 – Зовнішній вигляд бокової панелі застосунку

Вкладка «Дзвінки та години роботи» відправляє на сторінку з інформацією про розклад дзвінків, годин роботи бібліотеки та годин роботи станцій друку, зовнішній вигляд якої наведений на рисунку нижче.

Рисунок 3.13 – Зовнішній вигляд сторінки «Дзвінки та години роботи»

Вкладка «Розташування корпусів» відправляє на сторінку з мапою університету (див. рис. 3.14), яка може бути корисна для першокурсників, які не знають розташування корпусів.

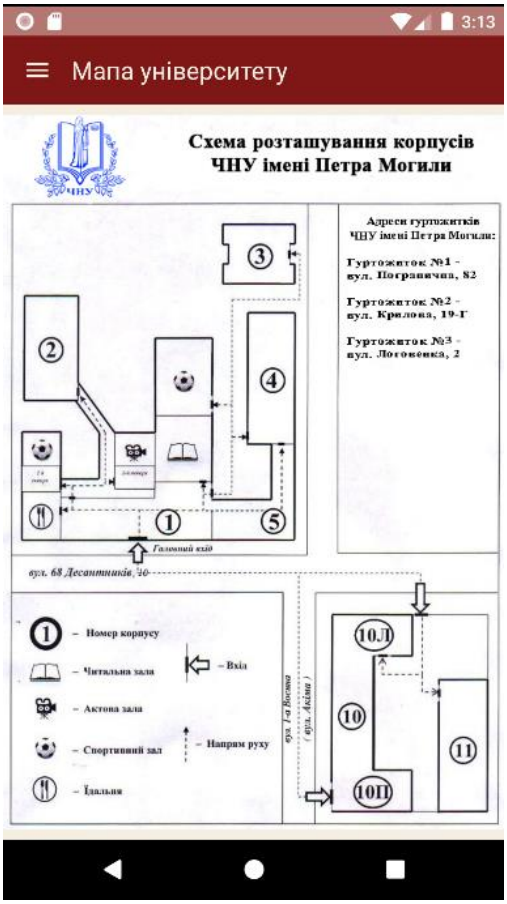

Рисунок 3.14 – Сторінка «Мапа університету»

Вкладка «Знайти викладача» переносить користувача на сторінку з пошуком викладача, зовнішній вигляд якої наведений на рисунку 3.15.

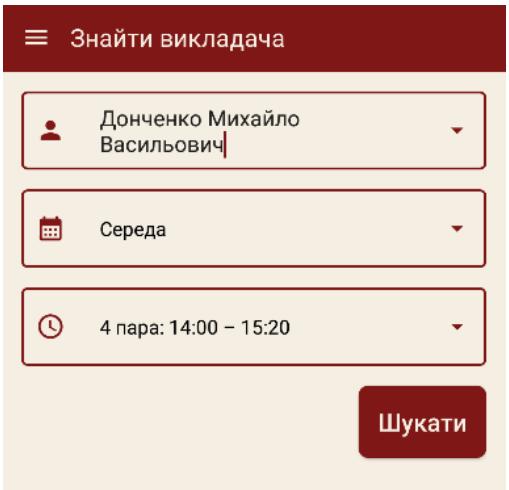

Рисунок 3.15 – Сторінка «Знайти викладача»

На цій сторінці можна обрати шуканого викладача зі спадного списку з викладачами, обрати день тижня і час з відповідних спадних списків. Після вибору усіх даних, при натисканні на кнопку «Шукати» виводиться спливаюче вікно (див. рис. 3.16) з наступною інформацією:

– ім’я викладача;
– факультет і кафедра викладача;
– контактні дані викладача, а саме пошта і телефон;
– місце знаходження – обраний час і місце на 1 та 2 тижнях.

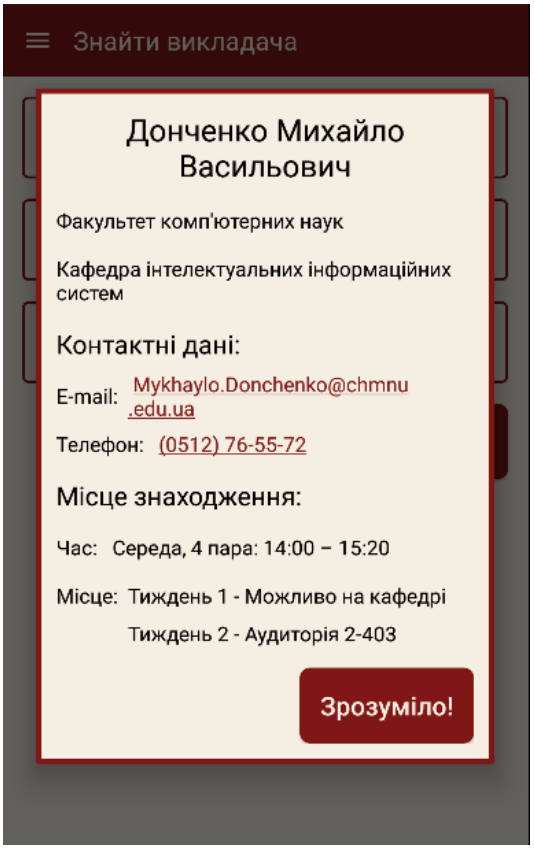

Рисунок 3.16 – Спливаюче вікно з шуканим викладачем

3.3 Опис основних функцій у програмному коді

Функція з’єднання з базою даних

Одна з основних функцій програми – з’єднання з базою даних. З’єднання налаштоване у класі DBConnection (див. рис. 3.17) за допомогою функції connect().

```
public class DBConnection {
    public Connection connect() {
        Connection connection = null;
        String ConnectionURL;
        try{
            StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);
            Class.forName("com.mysql.jdbc.Driver");
            ConnectionURL = "jdbc:mysql://192.168.0.193:3306/university_timetable?useSSL=false";
            connection = DriverManager.getConnection(ConnectionURL, user: "root", password: "YdhUdy");
        } catch (Exception exception){
            if(exception.getMessage() != null){
                Log.e( tag: "Connection to DB Error", exception.getMessage());
                exception.printStackTrace();
            }
        }
        return connection;
    }
}
```

Рисунок 3.17 – Програмний код класу DBConnection

Функція connect() класу DBConnection використовується в різних частинах програми, де треба взяти інформацію з бази даних. Приклад використання цієї функції наведений на рисунку нижче.

```
DBConnection connection = new DBConnection();
try{
    connect = connection.connect();
    st = connect.createStatement();
    rs = st.executeQuery( sql: "SELECT id FROM university_timetable.subgroups;");
    while (rs.next()){
        groupsArrList.add(rs.getString( columnIndex: 1));
    }
} catch (Exception e){
    e.printStackTrace();
}
```

Рисунок 3.18 – Приклад використання функції connect()

Запам’ятовування вибору користувача

Функція збереження вибору користувача реалізована за допомогою класу SharedPreferences. За допомогою цього класу інформація зберігається до кешу програми, а потім, при звертанні до цієї інформації її надає. На рисунку 3.19 наведено приклад використання цього класу у коді. Спочатку спадному списку присвоюється адаптер, після цього створюється рядок selectedWeek і за допомогою класу SharedPreferences в цей рядок присвоюється збережена в кеш інформація, про попередньо обраний тиждень. В свою чергу ця інформація

переходить в кеш при виборі тижня вручну у слухачі вибору onItemSelectedListener, інформація про обраний тиждень переходить в кеш у функції класу SharedPreferences.Editor – putString().

```
// ініціалізація випадаючого списку з тижнями
ArrayAdapter<CharSequence> weekArrayAdapter = ArrayAdapter.createFromResource( context: this, R.array.week, R.layout.spinner_item);
spinner_week.setAdapter(weekArrayAdapter);

sharedPreferences = getPreferences(MODE_PRIVATE);
String selectedWeek = sharedPreferences.getString(SELECTED_WEEK,  defValue: "1");
spinner_week.setSelection(weekArrayAdapter.getPosition(selectedWeek));

spinner_week.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener(){
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
        setupTabLayout();
        selectCurrentDayTab();

        sharedPreferences = getPreferences(MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString(SELECTED_WEEK, spinner_week.getSelectedItem().toString());
        editor.apply();
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {

    }
});
```

Рисунок 3.19 – Приклад використання класу SharedPreferences

Автоматичний вибір дня тижня при запуску програми

Зручна функція автоматичного вибору дня тижня при запуску програми реалізована у функції класу TimetableActivity – selectCurrentDayTab(). Програмний код цієї функції наведений на рисунку 3.20. Ця функція спочатку перевіряє, чи не неділя сьогодні, якщо ні, то за допомогою класу Calendar програма дізнається поточний день і функція select() обирає вкладку з цим днем.

```
private void selectCurrentDayTab(){
    if(Calendar.getInstance().get(Calendar.DAY_OF_WEEK) != Calendar.SUNDAY){
        tabLayout_Timetable.getTabAt( index: Calendar.getInstance().get(Calendar.DAY_OF_WEEK) - 2).select();
    }
}
```

Рисунок 3.20 – Код функції selectCurrentDayTab()

Перегляд розкладу тижня

Зручний перегляд розкладу реалізований за допомогою компонента viewPager. Цей компонент зв’язує фрагмент розкладу з панеллю вкладок з днями тижня. Реалізовано це у функції класу TimetableActivity –

setupTabLayout() (див. рис. 3.21). У функції створюються вкладки з назвами днів тижня за допомогою адаптера TimetableFragmentAdapter і після цього компоненту присвоюється цей адаптер.

```
private void setupTabLayout(){
    ttAdapter = new TimetableFragmentAdapter(getSupportFragmentManager());
    ttAdapter.addFragment( title: "Понеділок");
    ttAdapter.addFragment( title: "Вівторок");
    ttAdapter.addFragment( title: "Середа");
    ttAdapter.addFragment( title: "Четвер");
    ttAdapter.addFragment( title: "П'ятниця");
    ttAdapter.addFragment( title: "Субота");
    viewPager_weekday.setAdapter(ttAdapter);
}
```

Рисунок 3.21 – Код функції setupTabLayout()

Адаптер TimetableFragmentAdapter відповідає за зв’язок між ViewPager та фрагментом. У самому класі, у функції addFragment() (див. рис. 3.22), створюється фрагмент класу TimetableFragment.

```
public void addFragment(String title){
    TimetableFragment fragment = new TimetableFragment();

    Bundle bundle = new Bundle();
    bundle.putString("title", title);
    fragment.setArguments(bundle);

    fragmentsArrayList.add(fragment);
    fragmentTitle.add(title);
}
```

Рисунок 3.22 – Код функції addFragment()

Фрагмент класу TimetableFragment зовнішньо виглядає наступним чином(див. рис. 3.23).

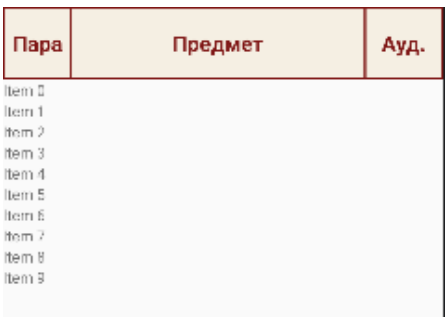

Рисунок 3.23 – Зовнішній вигляд фрагмента розкладу занять

На рисунку вище, можна помітити, що замість розкладу міститься не ініціалізований список, але саме за допомогою цього списку реалізована універсальність фрагменту, і його можна підставити під будь-який день тижня, а розклад буде різний. Список ініціалізується за допомогою ще одного адаптера під назвою TTRecViewAdapter. Цей адаптер ініціалізує список компонентами, які зовнішньо виглядають, як на рисунку нижче.

Рисунок 3.24 – Зовнішній вигляд компонента списку розкладу занять

На рисунку вище видно, що дані стандартні, але при ініціалізації вони замінюються даними з бази даних. Дані з бази даних отримуються у класі TimetableFragment у функції getList() (див. рис. 3.25 та 3.26), записуються у об’єкти класу LessonModel і передаються у TTRecViewAdapter через конструктор у вигляді List<LessonModel>, наповненим цими самими об’єктами.

```
private List<LessonModel> getList() {
    List<LessonModel> lessonModel = new ArrayList<>();

    Spinner spinnerWeek = getActivity().findViewById(R.id.spinner_week);
    Spinner spinnerSubgroup = getActivity().findViewById(R.id.spinner_group);

    Bundle bundle = this.getArguments();

    String week = spinnerWeek.getSelectedItem().toString();
    String weekday = bundle.getString( key: "title");
    String group;
    String subgroup = spinnerSubgroup.getSelectedItem().toString();

    try {
        DBConnection connection = new DBConnection();
        connect = connection.connect();
        st = connect.createStatement();

        // отримуємо групу
        rs = st.executeQuery( sql: "SELECT subgroups.group FROM university_timetable.subgroups " +
                "WHERE subgroups.id = '" + subgroup + "';");
        rs.next();
        group = rs.getString( columnIndex: 1);

        if(weekday.equals("П'ятниця")){
            weekday = "П\\'ятниця";
        }
```

Рисунок 3.25 – Код функції getList()

```
// отримуємо порядковий номер пари, дисципліну, ім'я вчителя та аудиторію, і записуємо у модель
rs = st.executeQuery( sql: "SELECT lessonSequenceNumb, discipline, teacher, classroom\n" +
        "FROM university_timetable.timetable\n" +
        "WHERE (timetable.week = '" + week + "' OR timetable.week = '1/2')\n" +
        "AND timetable.weekday = '" + weekday + "'\n" +
        "AND timetable.group = '" + group + "'\n" +
        "AND (timetable.subgroup = '" + subgroup + "' OR timetable.subgroup IS NULL);");
while (rs.next()){
    lessonModel.add(new LessonModel(rs.getString( columnIndex: 1), rs.getString( columnIndex: 2),
            rs.getString( columnIndex: 3), rs.getString( columnIndex: 4)));
}

// отримуємо наукову ступіть, вчене звання та посаду учителів, і дозаписуємо їх у модель
for(int i = 0; i < lessonModel.size(); i++){
    String teachersFullName = lessonModel.get(i).teacherFullName;
    rs = st.executeQuery( sql: "SELECT degree, academicStatus, position FROM university_timetable.teachers " +
            "WHERE teachers.fullName = '" + teachersFullName + "';");
    rs.next();
    lessonModel.get(i).setTeachersDegree(rs.getString( columnIndex: 1));
    lessonModel.get(i).setTeachersAcademicStatus(rs.getString( columnIndex: 2));
    lessonModel.get(i).setTeachersPosition(rs.getString( columnIndex: 3));
}

// заповнюємо пусті пари у листі
if(lessonModel.isEmpty()){
    for(int i = 1; i <= 7; i++){
        lessonModel.add(new LessonModel( lessonSequenceNumb: "" + i,  discipline: "",  teacherFullName: "",  classroom: ""));
    }
} else{
    for(int i = 1; i <= 7; i++){
        boolean containsEl = false;
        for(int j = 0; j < lessonModel.size(); j++){
            if(lessonModel.get(j).getLessonSequenceNumb().equals("" + i)){
                containsEl = true;
                break;
            }
        }
        if(!containsEl){
            lessonModel.add(new LessonModel( lessonSequenceNumb: "" + i,  discipline: "",  teacherFullName: "",  classroom: ""));
        }
    }
}
```

Рисунок 3.26 – Продовження коду функції getList()

На рисунку 3.27 видно, як лист з об’єктами LessonModel у вигляді функції getList() передаються у конструктор адаптера TTRecViewAdapter

```
public void setRecyclerView(){
    recyclerView_timetable.setHasFixedSize(true);
    recyclerView_timetable.setLayoutManager(new LinearLayoutManager(this.getContext()));
    ttRecViewAdapter = new TTRecViewAdapter(this.getContext(), getList());
    recyclerView_timetable.setAdapter(ttRecViewAdapter);
}
```

Рисунок 3.27 – Код функції setRecyclerView()

Після передачі листа з об’єктами, у функції onBindViewHolder (див. рис. 3.28) класу TTRecViewAdapter інформація дістається з об’єктів класу LessonModel та записується у компонент макету timetable.row, зовнішній вигляд якого було наведено на рисунку 3.24.

```
@Override
public void onBindViewHolder(@NonNull MyViewHolder myViewHolder, int i) {
    if(lessonList != null && lessonList.size() > 0){
        LessonModel lessonModel = lessonList.get(i);

        String teachersFullName = lessonModel.getTeacherFullName() + " ";
        String teachersDegree = lessonModel.getTeachersDegree().equals("-") ?
                "" : lessonModel.getTeachersDegree() + ", ";
        String teachersAcademicStatus = lessonModel.getTeachersAcademicStatus().equals("-") ?
                "" : lessonModel.getTeachersAcademicStatus() + ", ";
        String teachersPosition = lessonModel.getTeachersPosition().equals("-") ?
                "" : lessonModel.getTeachersPosition();

        if(teachersPosition.isEmpty()){
            if(!teachersAcademicStatus.isEmpty()){
                teachersAcademicStatus = lessonModel.getTeachersAcademicStatus();
            }
        }

        if(teachersAcademicStatus.isEmpty()){
            if(!teachersDegree.isEmpty()){
                teachersDegree = lessonModel.getTeachersDegree();
            }
        }
        String teacher =
                teachersFullName + teachersDegree + teachersAcademicStatus + teachersPosition;

        if(lessonModel.discipline.equals("")){
            myViewHolder.lessonSeqNumb.setText(lessonModel.getLessonSequenceNumb());
            myViewHolder.discipline.setText("");
            myViewHolder.teacher.setText("");
            myViewHolder.classroom.setText("");
        } else{
            myViewHolder.lessonSeqNumb.setText(lessonModel.getLessonSequenceNumb());
            myViewHolder.discipline.setText(lessonModel.getDiscipline());
            myViewHolder.teacher.setText(teacher);
            myViewHolder.classroom.setText(lessonModel.getClassroom());
        }
    }
}
```

Рисунок 3.28 – Код функції onBindViewHolder()

Бокова панель

Бокова панель реалізована за допомогою файлів side_menu.xml, де знаходяться компоненти списку вкладок меню, nav_header.xml, де задана шапка навігаційного меню, та програмного коду, який підтримує роботу навігації та об’єднує ці файли. Цей програмний код знаходиться у кожній активності застосунку і трохи відрізняється лише у функції навігації. Приклад коду функції бокової панелі наведено на рисунку нижче.

```
@Override
public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
    Intent intent;
    switch (menuItem.getItemId()){
        case R.id.timetable:
            drawer.closeDrawer(GravityCompat.START);
            break;
        case R.id.alarmTimetable:
            intent = new Intent( packageContext: this, TimeScheduleActivity.class);
            startActivity(intent);
            break;
        case R.id.findTeacher:
            intent = new Intent( packageContext: this, FindTeacherActivity.class);
            startActivity(intent);
            break;
        case R.id.universityMap:
            intent = new Intent( packageContext: this, UniversityMapActivity.class);
            startActivity(intent);
            break;
    }
    drawer.closeDrawer(GravityCompat.START);
    return true;
}
```

Рисунок 3.29 – Код функції onNavigationItemSelected()

Сторінка «Дзвінки та години роботи»

Реалізація сторінки з розкладом дзвінків та годинами роботи реалізована дуже просто, статична інформація, яка записана у файлі strings.xml, записується у компоненти макету активності за допомогою адаптерів і виводиться для користувача. Код ініціалізації наведено нижче.

```
// наповнення листів розкладу дзвінків та годин роботи
lessonTimeSchedule.setLayoutManager(new LinearLayoutManager( context: this));
TimeScheduleRecViewAdapter lessonTimeScheduleAdapter =
        new TimeScheduleRecViewAdapter( context: this, getResources().getStringArray(R.array.alarmSchedule));
lessonTimeSchedule.setAdapter(lessonTimeScheduleAdapter);

libraryWorkHours.setLayoutManager(new LinearLayoutManager( context: this));
TimeScheduleRecViewAdapter libraryWorkHoursAdapter =
        new TimeScheduleRecViewAdapter( context: this, getResources().getStringArray(R.array.libraryWorkHours));
libraryWorkHours.setAdapter(libraryWorkHoursAdapter);

printingStationWorkHours.setLayoutManager(new LinearLayoutManager( context: this));
TimeScheduleRecViewAdapter printingStationWorkHoursAdapter =
        new TimeScheduleRecViewAdapter( context: this, getResources().getStringArray(R.array.printingStationWorkHours));
printingStationWorkHours.setAdapter(printingStationWorkHoursAdapter);
```

Рисунок 3.30 – Код ініціалізації інформації розкладу дзвінків і годин роботи

Сторінка «Розташування корпусів»

Сторінка з мапою університету теж реалізована дуже просто, картинка з мапою збережена у папці res/drawable і використовується прямо у макеті активності без програмної ініціалізації. Використання картинки у макеті наведено на рисунку нижче.

```
<ImageView
    android:id="@+id/imageView2"
    android:layout_width="wrap_content"
    android:layout_height="0dp"
    android:background="@color/lightBrown"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/toolbar"
    app:srcCompat="@drawable/university_map" />
```

Рисунок 3.31 – Код використання картинки у макеті activity_university_map

Сторінка «Знайти викладача»

На сторінці з пошуком викладача знаходиться 4 компоненти:

– 1 автоматично заповнюваний спадний список з викладачами
– 2 спадні списки з вибором дня тижня та часом
– 1 кнопка пошуку

Перші 3 компоненти ініціалізуються за допомогою адаптерів. Код ініціалізації компонентів наведений на рисунку нижче.

```
// отримуємо вчителів з бази даних і ініціалізуємо випадаючий список
ArrayAdapter<String> teachersAdapter =
        new ArrayAdapter<>( context: this, R.layout.dropdown_list_item, getTeachers());
autoCompleteTeacher.setAdapter(teachersAdapter);

// ініціалізуємо випадаючий список з днями тижня
ArrayAdapter<String> weekdayAdapter =
        new ArrayAdapter<~>( context: this, R.layout.dropdown_list_item, getResources().getStringArray(R.array.weekdays));
spinnerWeekday.setAdapter(weekdayAdapter);

// ініціалізуємо пвипадаючий список з часом
ArrayAdapter<String> timeAdapter =
        new ArrayAdapter<~>( context: this, R.layout.dropdown_list_item, getResources().getStringArray(R.array.alarmSchedule));
spinnerTime.setAdapter(timeAdapter);
```

Рисунок 3.32 – Код ініціалізації компонентів за допомогою адаптерів

При створенні активності автоматично заповнюваний список ініціалізується викладачами за допомогою адаптера який бере список

викладачів з бази даних у функції getTeachers та сортується за допомогою компаратора teacherComparator, код якої наведено на рисунку нижче.

```
public String[] getTeachers(){
    List<String> teachersList = new ArrayList<>();

    DBConnection connection = new DBConnection();
    connect = connection.connect();
    try {
        st = connect.createStatement();
        // отримуємо вчителів
        rs = st.executeQuery( sql "SELECT teachers.fullName FROM university_timetable.teachers;");
        while(rs.next()){
            teachersList.add(rs.getString( columnIndex: 1));
        }
        rs.close();
        st.close();
        connect.close();

        // сортування отриманого листа
        teachersList.sort(teachersComparator);

    } catch (SQLException e) {
        e.printStackTrace();
    }
    // конвертація листа в масив
    String[] teachersArr = new String[teachersList.size()];
    teachersArr = teachersList.toArray(teachersArr);
    return teachersArr;
}

public static Comparator<String> teachersComparator = new Comparator<String>() {
    @Override
    public int compare(String str1, String str2) { return str1.compareTo(str2); }
};
```

Рисунок 3.33 – Код функції getTeachers() та компаратора teacherComparator

Компоненти спадних списків ініціалізуються теж за допомогою адаптерів але зі сталих даних з файлу strings.xml.

При натисканні на кнопку пошуку визивається функція onClick_FindTeacher() (див. рис. 3.34), де з попередніх компонентів береться інформація і починається пошук у базі даних, під час чого знайдена інформація записується у рядки.

Після запису необхідної інформації за допомогою функції createTeacherPopupDialog() (див. рис. 3.35) створюється спливаюче вікно з інформацією про вчителя, його факультет і кафедру, контактна інформація, а

саме пошта і телефон, і місце розташування – шуканий час та місце на першому і другому тижнях.

```
public void onClick_FindTeacher(View view){
    if(autoCompleteTeacher.getText().toString().isEmpty()){
        Toast.makeText( context: this,  text: "Поле порожне!", Toast.LENGTH_LONG).show();
    } else{
        String teacher = autoCompleteTeacher.getText().toString();
        String faculty = "";
        String department = "";
        String email = "";
        String phoneNumb = "";
        String time = spinnerWeekday.getSelectedItem().toString() + ", " + spinnerTime.getSelectedItem().toString();
        String week1location = "";
        String week2location = "";

        try {
            DBConnection connection = new DBConnection();
            connect = connection.connect();
            st = connect.createStatement();
            rs = st.executeQuery( sql: "SELECT teachers.faculty, teachers.department, teachers.email, teachers.phoneNumb " +
                    "FROM university_timetable.teachers WHERE teachers.fullName='"+ teacher + "';");
            if(rs.next()){
                faculty = rs.getString( columnIndex: 1);
                department = rs.getString( columnIndex: 2);
                email = rs.getString( columnIndex: 3);
                phoneNumb = rs.getString( columnIndex: 4);
                rs = st.executeQuery( sql: "SELECT timetable.week, timetable.classroom FROM university_timetable.timetable\n" +
                        "WHERE timetable.teacher='"+ teacher + "'\n" +
                        "AND timetable.weekday='" + spinnerWeekday.getSelectedItem().toString() + "'\n" +
                        "AND timetable.lessonSequenceNumb='" + spinnerTime.getSelectedItem().toString().toCharArray()[0] + "';");
                while(rs.next()){
                    switch(rs.getString( columnIndex: 1)){
                        case "1/2":
                            week1location = "Тиждень 1 - Аудиторія " + rs.getString( columnIndex: 2);
                            week2location = "Тиждень 2 - Аудиторія " + rs.getString( columnIndex: 2);
                            break;
                        case "1":
                            week1location = "Тиждень 1 - Аудиторія " + rs.getString( columnIndex: 2);
                            break;
                        case "2":
                            week2location = "Тиждень 2 - Аудиторія " + rs.getString( columnIndex: 2);
                            break;
                    }
                }

                if(week1location.isEmpty()){
                   week1location = "Тиждень 1 - Можливо на кафедрі";
                }
                if(week2location.isEmpty()){
                   week2location = "Тиждень 2 - Можливо на кафедрі";
                }

                createTeacherPopupDialog();
                teacherFullNameTV.setText(autoCompleteTeacher.getText());
                facultyTV.setText(faculty);
                departmentTV.setText(department);
                emailTV.setText(email);
                phoneNumbTV.setText(phoneNumb);
                timeTV.setText(time);
                week1locationTV.setText(week1location);
                week2locationTV.setText(week2location);
                dialog.show();
            } else {
                Toast.makeText( context: this,  text: "Такого викладача не існує!", Toast.LENGTH_LONG).show();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

Рисунок 3.34 – Код функції onClick_FindTeacher()

```
public void createTeacherPopupDialog(){
    dialogBuilder = new AlertDialog.Builder( context: this);
    final View teacherPopupView = getLayoutInflater().inflate(R.layout.popup_teacher,  root: null);

    teacherFullNameTV = teacherPopupView.findViewById(R.id.teacherFullName);
    facultyTV = teacherPopupView.findViewById(R.id.faculty);
    departmentTV = teacherPopupView.findViewById(R.id.department);
    emailTV = teacherPopupView.findViewById(R.id.email);
    phoneNumbTV = teacherPopupView.findViewById(R.id.phoneNumb);
    timeTV = teacherPopupView.findViewById(R.id.time);
    week1locationTV = teacherPopupView.findViewById(R.id.week1location);
    week2locationTV = teacherPopupView.findViewById(R.id.week2location);
    understandBtn = teacherPopupView.findViewById(R.id.button_understand);

    dialogBuilder.setView(teacherPopupView);
    dialog = dialogBuilder.create();

    understandBtn.setOnClickListener(v -> dialog.dismiss());
}
```

Рисунок 3.35 – Код функції createTeacherPopupDialog()

3.4 База даних проекту

База даних була розроблена на MySQL. Вона містить в собі наступні таблиці:

- академічні ступені;
- вчені звання;
- кафедри;
- факультети;
- групи;
- розклад дзвінків;
- типи пари;
- посади;
- підгрупи;
- викладачі;
- розклад занять;
- тижні;
- дні тижня;

Дерево бази даних виглядає наступним чином:

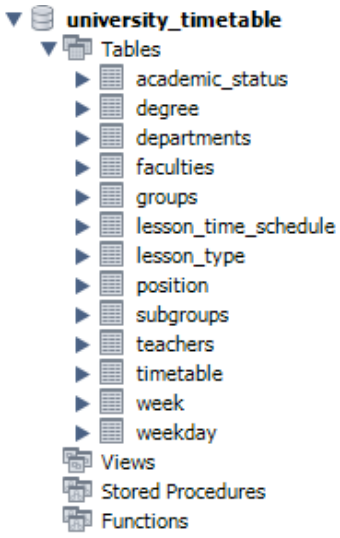

Рисунок 3.36 – Дерево бази даних

Зовнішньо відношення таблиць бази даних виглядає як на малюнку нижче.

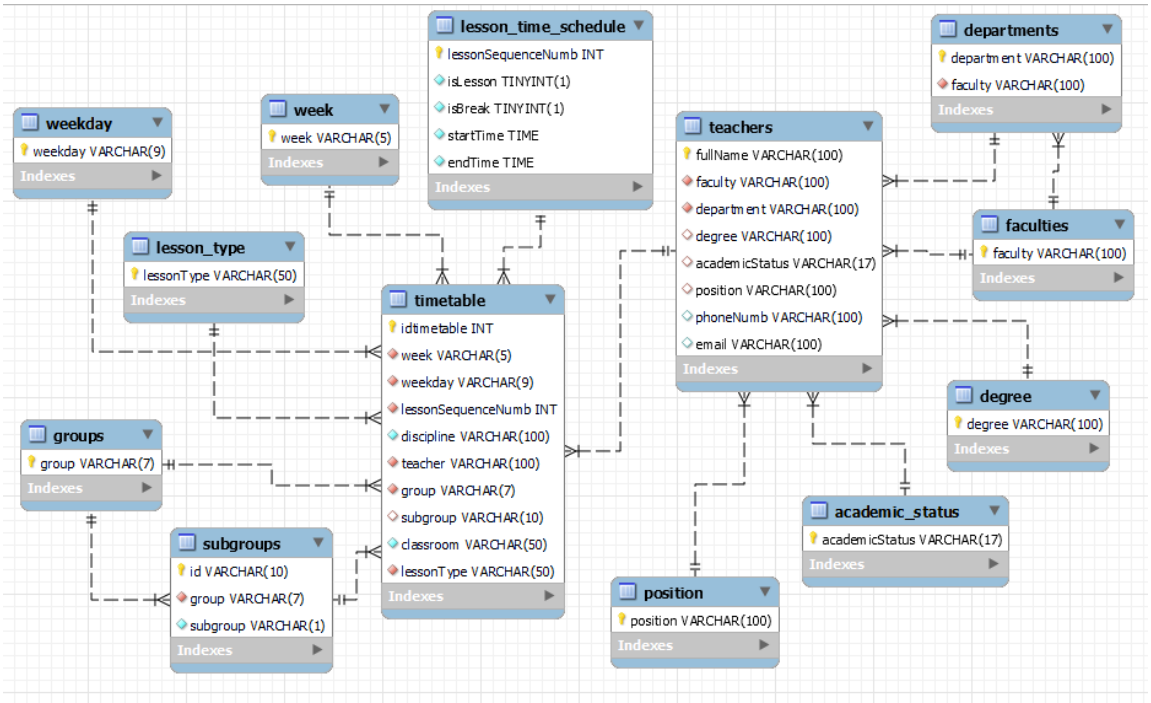

Рисунок 3.37 – Відношення таблиць бази даних

Більшість з таблиць наповнені стандартними даними, і доповнювати або змінювати треба лише таблиці з викладачами, групами, підгрупами та головну таблицю з розкладом занять.

Головна таблиця з розкладом занять із заповненими даними виглядає так:

idtimetable	week	weekday	less	discipline	teacher	group	subgroup	classroom	lessonType
1	1/2	Вівторок	4	Технології комп'ютерного проектування	Донченко Михайло Васильович	402	NULL	2-308	Практичне заняття
2	1/2	Вівторок	5	Системний аналіз	Калініна Ірина Олександрівна	402	NULL	2-308	Практичне заняття
3	1/2	Вівторок	6	Технології розподілених систем та паралельних обчислень	Старченко В'ячеслав Володимирович	402	NULL	2-308	Практичне заняття
4	1	Середа	4	Технології розподілених систем та паралельних обчислень	Старченко В'ячеслав Володимирович	402	NULL	2-403	Лекція
5	2	Середа	4	Технології комп'ютерного проектування	Донченко Михайло Васильович	402	NULL	2-403	Лекція
6	1/2	Середа	5	Системний аналіз	Калініна Ірина Олександрівна	402	NULL	2-403	Лекція
7	1/2	Четвер	4	Іноземна мова	Березовська Олена Анатоліївна	402	402(а)	11-419	Практичне заняття
8	1/2	Четвер	4	Іноземна мова	Діордієва Альона Вікторівна	402	402(б)	11-420	Практичне заняття
9	1/2	Четвер	5	Технології програмування на серверній платформі	Скакодуб Олександр Сергійович	402	NULL	2-306	Практичне заняття
10	1/2	Четвер	6	Проектування інформаційних та програмних систем	Скакодуб Олександр Сергійович	402	NULL	2-306	Практичне заняття
11	1/2	П'ятниця	4	Економіка ІТ-проектів	Лопатін Артем Олегович	402	NULL	2-406	Лекція
12	1/2	П'ятниця	5	Технології програмування на серверній платформі	Скакодуб Олександр Сергійович	402	NULL	2-406	Лекція
13	1	П'ятниця	6	Проектування інформаційних та програмних систем	Скакодуб Олександр Сергійович	402	NULL	2-406	Лекція
14	1	Субота	1	Економіка ІТ-проектів	Джигалюк Дмитро Віталійович	402	NULL	10-302	Практичне заняття
15	1	Субота	2	Економіка ІТ-проектів	Джигалюк Дмитро Віталійович	402	NULL	10-302	Практичне заняття

Рисунок 3.38 – Зовнішній вигляд таблиці розкладу занять з даними

Висновки до розділу 3

Під час розробки проекту було розроблено базу даних, яка містить таблиці з викладачами, академічними ступенями, вченими званнями, посадами, факультетами, кафедрами, групами, підгрупами, днями тижня, тижнями та розкладом занять.

Було створено зручний, інтуїтивний і красивий інтерфейс користувача. Програмно було реалізовано наступні функції застосунку:

– підключення до бази даних;
– сторінка розкладу занять, з можливістю вибору групи і тижня;
– запам’ятовування вибору користувача;
– автоматичний вибір дня тижня;
– бокова навігаційна панель застосунка;
– сторінка розкладу дзвінків та годин роботи структур університету;
– сторінка пошуку викладача;
– спливаюче вікно з виводом повної інформації про викладача та його місцем розташування;
– сторінка з мапою університета.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Спеціальний розділ

ОХОРОНА ПРАЦІ

до кваліфікаційної роботи

на тему:

АВТОМАТИЗОВАНА СИСТЕМА РОЗКЛАДУ ЗАНЯТЬ В УНІВЕРСИТЕТІ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.21810218

Виконав студент 4-го курсу, групи 402
_____ *А.В. Медвідь*

«___» _____ 2022 р.

Консультант канд. техн. наук, доцент
_____*А. О. Алексєєва*

«___» _____ 2022 р.

Миколаїв – 2022

ЗМІСТ

ВСТУП

Сьогодні все більше організацій переходять на дистанційну форму праці, наприклад, через пандемію або війну. Але навіть так будь який вид праці має певні правила поведінки, задля збереження здоров’я. Звичайно є види праці які набагато небезпечніші. Але те, що хтось працює у більш безпечних умовах не звільняє його від дотримання правил охорони праці.

Тривала робота у сидячому положенні тіла за екранними пристроями може у тривалій перспективі мати негативні наслідки, якщо не дотримуватись правил охорони праці.

Перш за все, слід зазначити, що незалежно від типу та характеру роботи, яку виконує працівник, роботодавець завжди повинен дотримуватися нинішніх правил захисту праці та заходів безпеки. Основою для їх дотримання є правильна організація роботодавця робочого середовища та забезпечення відповідних умов для її впровадження. Робота за комп’ютером також регулюється правилами безпеки.

Правила захисту праці та запобіжних заходів досить чітко визначають, які вимоги щодо захисту праці та безпеки повинні дотримуватися роботодавців під час найму працівників. Однак часто підприємці застосовують ці норми лише до виробничих працівників (у галузі робочого одягу та взуття), забуваючи, що офісні працівники також мають власні права. Комп'ютерні працівники особливо вразливі в цьому випадку. Робота за комп’ютером і довгі години до того, як монітор може негативно вплинути на здоров'я працівника, особливо погіршити його бачення, тому в цій ситуації так важливо відповідати нинішнім правилам безпеки та захисту праці.

4 ОХОРОНА ПРАЦІ

4.1 Нормативна база в галузі охорони праці при роботі з екранними пристроями

Екранний пристрій - електронний засіб для відтворення будь-якої графічної або буквено-цифрової інформації (на основі електронно-променевих трубок, рідких кристалів, плазми, проекцій, органічних світлодіодних дисплеїв та інших нових розробок у сфері інформаційних технологій) [11].

Мінімальні вимоги до безпеки та охорони здоров’я при використанні екранних пристроїв усіх типів і моделей установлюють вимоги безпеки та охорони здоров’я працівників при використанні екранних пристроїв, затверджені наказом Мінсоцполітики від 14.02.2018 р. № 207. всім суб'єктам господарювання, незалежно від форм власності, організаційно-правової форми та діяльності, та встановлюють мінімальні вимоги безпеки та охорони праці до робіт, пов'язаних із використанням екранних пристроїв, незалежно від їх типу та моделі [11].

4.2 Загальні вимоги до роботодавців

Нижче наведено частину вимог з документу про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями:

Зокрема, роботодавець зобов'язаний:

– організовувати роботу таким чином, щоб забезпечити безпечні та гігієнічні умови праці;

– забезпечити дотримання положень та правил захисту праці та безпеки на робочому місці, видавати інструкції щодо усунення недоліків у цьому плані та відстеження виконання цих інструкцій;

– реагувати на потреби забезпечення охорони здоров'я та безпеки праці та адаптації заходів, що вживаються для підвищення існуючого

рівня охорони здоров'я та життя працівників, враховуючи зміни умов роботи;

– забезпечити розробку послідовної політики щодо запобігання нещасних випадків у виробничих та професійних захворюваннях, з урахуванням технічних питань, організації праці, умов праці, соціальних відносин та впливу факторів робочого середовища;

– враховувати захист здоров'я підлітків, вагітних або годувальних працівників та інвалідів у рамках профілактичних заходів;

– забезпечити виконання замовлень, заяви, рішень та наказів керівних органів;

– забезпечити виконання рекомендацій інспектора соціальної праці.

4.3 Вимоги до приміщення з використанням екранних пристроїв

Робочі місця працівників, які використовують екранні пристрої, повинні відповідати ергономічним, антропологічним, психофізіологічним вимогам, характеру виконуваної роботи. Вони повинні бути сконструйовані та розраховані таким чином, щоб працівники мали можливість змінювати своє робоче положення та рухи. Освітлення робочого місця повинно створювати відповідний контраст між екраном і навколишнім середовищем, враховуючи вид роботи та відповідаючи ДСанПіН 3.3.2.007-98. Мікроклімат промислових об’єктів та робочих місць працівників із екранним обладнанням необхідно підтримувати на постійному рівні відповідно до ДСН 3.3.6.042-99 «Гігієнічні норми мікроклімату на промислових об’єктах», затверджених МОЗ та Головним державним санітарним лікарем України. від 01.12.1999 № 42 . Робочі місця з екранними пристроями не повинні обмежувати пересування працівників.

Робочі місця з використанням екранних пристроїв повинні відповідати ергономічним, антропологічним, психофізіологічним вимогам і характеру виконуваної роботи. Вони повинні бути розроблені та розраховані таким

чином, щоб працівники мали можливість змінювати свої робочі положення та рухи. Освітлення робочого місця повинно створювати відповідний контраст між екраном і навколишнім середовищем, враховуючи вид роботи та відповідаючи ДСанПіН 3.3.2.007-98. Відповідно до ДСН 3.3.6.042-99 «Стандарт мікроклімату гігієни промислових об'єктів», затвердженого МОЗ та державного головного санітарного лікаря, мікроклімат промислових об'єктів і робочих місць працівників має підтримуватися на постійному рівні України. Від 01.12.1999 № 42. Робочі місця з екранними пристроями не повинні обмежувати пересування працівників.

Крім цього для зменшення впливу негативних факторів на працівника, виробниче приміщення повинно відповідати нормам щодо параметрів мікроклімату, освітлення, шуму та вібрації, рівні електромагнітного та іонізуючого випромінювання.

По-перше, на усіх робочих місцях із електронно-обчислювальними машинами обов'язково має бути забезпечено оптимальні значення параметрів температури відносної вологості й рухливості повітря (ГОСТ 12.1.005-88 [14], СН 4088-86 [17]) (табл. 4.1).

Таблиця 4.1 – Нормовані величини температури, відносної вологості та швидкості руху в робочій зоні виробничих приміщень з ВДТ

Пора року	Категорія робіт	Оптимальна температура повітря, °С, не більше	Оптимальна відносна вологість повітря, %	Оптимальна швидкість руху повітря, м/с
Холодна	Легка – Іа	22-24	40-60	0,1
	Легка – Іб	21-23	40-60	0,1
Тепла	Легка – Іа	23-25	40-60	0,1
	Легка – Іб	22-24	40-60	0,2

До категорії робіт Іа належать, що виконується робота сидячі і не потребує фізичного напруження людини, до категорії робіт Іб належать, що робота виконується сидячі, стоячи або пов'язані з ходінням та потребує деякого фізичного напруження.

По-друге, показники рівню позитивних та негативних іонів в повітрі приміщень з ВДТ мають відповідати санітарно – гігієнічним нормам №2152-80 [14] (табл. 4.2).

Таблиця 4.2 – Рівні іонізації повітря приміщень при роботі на ВДТ

Рівні	Кількість іонів в 1 $см^3$ повітря	
	n+	n-
Мінімально необхідний	400	600
Оптимальний	1500-3000	3000-5000
Максимально допустимий	50000	50000

Вимірювання кількості іонів та його полярності порядку поточного нагляду проводиться 1 разів у квартал. Вимірювання проводяться також у випадках:

– встановлення нових або відремонтованих іонізаторів;
– організації нових робочих місць;
– впровадження нових технологічних процесів, що потенційно можуть змінити іонний режим у зоні дихання персоналу.

Все необхідне обладнання, необхідне для роботи на комп'ютері, повинно бути в межах досяжності працівника, не змушуючи його змінити місце роботи.

4.4 Вимоги до роботи з екранними пристроями

На сьогодні існує ряд вимог, які повинні використовувати працівники на підприємствах, основним місцем роботи яких є місце за персональним комп'ютером. Ці вимоги можна поділити за наступними критеріями:

1) Ергономіка положення:

– робоче місце повинно бути достатньо великим, щоб вільно використовувати всі елементи з ручним управлінням у межах світла;

– відстань між моніторами повинна бути 60 см, а між працівником та задньою частиною іншого сусіднього монітора - 80 см;

– відстань між працівником та монітором повинна бути від 40 см до 75 см;

– для кожного працівника має бути 2 м2 вільної площі та 13 м2 обсягу приміщення, без обмежень на меблі, обладнання тощо;

– екран монітора повинен відповідати певним вимогам: написи на екрані повинні бути чіткими та розбірливими, зображення повинно бути стабільним, екран повинен бути охоплений просвітлюючим шаром або оснащеним відповідним фільтром тощо;

– клавіатура комп'ютера повинна бути не менше 10 см від краю таблиці і повинна бути окремим елементом основного обладнання робочого місця, його поверхня повинна бути матовою, а символи контрастні та вибіркові.

2) Крісло як елемент робочого місця біля екрану монітора:

– має бути стійким з як мінімум 5 опорною базою, на колесах з можливістю обертання на 360° навколо вертикальної осі;

– розміри спинки та сидіння повинні забезпечувати зручне положення тіла та свободу рухів;

– з можливістю регулювання висоти сидіння в діапазоні 40-50 см, рахуючи від підлоги;

– з можливістю регулювання висоти спинки та регулювання нахилу спинки в діапазоні 5° вперед і 30° назад;

– сидіння та спинка мають бути сконструйовані таким чином, щоб відповідати природному вигину хребта та стегна;

– повинні бути обладнані підлокітниками;

– за бажанням працівника, а також у випадках, коли висота крісла унеможливлює розміщення працівника ступнями на підлозі, робоче місце має бути обладнане підставкою для ніг.

3) Розміри та налаштування стола:

– висота столу повинна забезпечувати вільне положення працівника, зберігаючи прямий кут між плечем і передпліччям під час роботи за комп'ютером;

– висота столу, на якому розташований екран монітора, повинна забезпечувати відповідний кут огляду екрана монітора в діапазоні 20-50° вниз (враховуючи від горизонтальної лінії на рівні очей співробітника до лінії, проведеної від очима до центру екрана), верхнім краєм екран монітора не повинен бути над очима працівника;

– коліна працівника, що сидить за столом, не повинні торкатися стільниці, простір під столом має бути вільним;

– поверхня стільниці повинна бути матовою, бажано світлого кольору.

4.5 Вимоги щодо режиму відпочинку та праці на підприємствах з екранними пристроями

Обов'язок роботодавця – організувати роботу так, щоб кожен працівник після кожної години безперервної роботи за комп'ютером мав можливість змінити вид роботи на такий, який не обтяжуватиме зір або буде виконуватися в зміненому положенні тіла. Якщо роботодавець не може забезпечити зміну

роботи, він повинен забезпечити перерву не менше 5 хвилин на кожну годину роботи перед екраном. Така перерва зараховується до робочого часу працівника і не зменшує оплату праці.

При виконанні протягом дня робіт, що належать до різних видів трудової діяльності, за основну роботу з ВДТ ЕОМ і ПЕОМ слід вважати таку, що займає не менше 50% часу впродовж робочої зміни мають передбачатися [12]:

– перерви для відпочинку і вживання їжі (обідні перерви);
– перерви для відпочинку і особистих потреб (згідно з трудовими нормами);
– додаткові перерви, що вводяться для окремих професій з урахуванням особливостей трудової діяльності.

Встановлюються такі внутрішньо змінні режими праці та відпочинку при роботі з ЕОМ при 8-годинній денній робочій зміні в залежності від характеру праці:

– для розробників програм із застосуванням ЕОМ, слід призначати регламентовану перерву для відпочинку тривалістю 15 хвилин через кожну годину роботи за ВДТ;
– для операторів із застосування ЕОМ, слід призначати регламентовані перерви для відпочинку тривалістю 15 хвилин через кожні дві години;
– для операторів комп'ютерного набора слід призначати регламентовані перерви для відпочинку тривалістю 10 хвилин після кожною години роботи за ВДТ.

4.6 Профілактика захворювань органів зору

Напруга очей, пов’язана з цифровими методами, вражає людей різного віку. Проводячи багато годин на день з цифровими пристроями, можна помітити, що зір затуманюється, а очі болять і втомлюються. Також можна виявити, що очі стають сухими, сльозяться або щипатимуть. Це напруження

очей нічим не відрізняється від симптомів, які можуть виникнути під час читання, письма або виконання «близької роботи», наприклад шиття протягом тривалого часу.

Отже, треба робити профілактичні вправи, щоб уникнути захворювань органів зору.

Правила користування для запобігання захворювань органів зору:

1. Моргати. Люди зазвичай моргають близько 15 разів за одну хвилину. Однак дослідження показують, що ми моргаємо приблизно від 5 до 7 разів на хвилину, використовуючи комп'ютери та інші пристрої з цифровим екраном. Моргання — це спосіб ока отримати необхідну вологу на своїй поверхні;
2. Зробити свідоме зусилля моргати якомога частіше. Це запобігає висиханню поверхні очей. Можна розмістити на екрані комп'ютера нотатку, яка нагадує часто моргати;
3. Використовувати штучні сльози, щоб освіжити очі, коли вони сухі. При частому перебуванні в сухому теплому приміщенні, треба подумати про використання зволожувача, щоб додати вологу в повітря;
4. Дотримуватись правила «20-20-20». Робіть регулярні перерви, використовуючи правило «20-20-20»: кожні 20 хвилин переводьте очі, щоб дивитися на об'єкт на відстані принаймні 20 метрів, принаймні 20 секунд.
5. Використовувати комп'ютерні окуляри. При роботі за комп'ютером багато годин поспіль, можна виявити, що використання комп'ютерних окулярів зменшує навантаження на очі. Ці окуляри за рецептом дозволяють фокусувати очі на відстані від екрана комп'ютера (проміжна відстань, яка становить приблизно 40-60 сантиметрів від вашого обличчя). Деякі з цих окулярів мають мультифрактальні лінзи, які допомагають швидко переміщати фокус з близької, середньої та далекої відстані. Треба мати на увазі, що комп'ютерні окуляри для

зниження навантаження на очі – це не ті самі, що «блокують синє світло».

6. Налаштування яскравості та контрастності. Якщо ваш екран світиться яскравіше, ніж оточення, вашим очам доведеться більше працювати, щоб бачити. Налаштуйте яскравість екрана відповідно до рівня освітлення навколо вас. Також спробуйте збільшити контраст на екрані, щоб зменшити навантаження на очі.

7. Зменшити відблиски. Екрани сучасних цифрових пристроїв часто мають багато відблисків. Спробуйте використовувати матовий екранний фільтр, щоб зменшити відблиски. Зверніться до свого комп’ютерного магазину чи магазину мобільних телефонів, щоб дізнатися, що вони можуть надати.

8. Регулювання положення біля комп'ютера. Користуючись комп’ютером, ви повинні сидіти на відстані приблизно 65 сантиметрів (приблизно на відстані витягнутої руки) від екрана. Крім того, розташуйте екран так, щоб ваші очі дивилися трохи вниз, а не прямо вперед чи вгору.

Вправи для профілактики захворювань органів зору

Вправа на зміну фокусування

1. Тримайте один палець на відстані декількох сантиметрів від одного ока;
2. Зосередьте погляд на пальці;
3. Повільно відсуньте палець від обличчя;
4. Зосередьтеся на об'єкті далі, а потім назад на палець;
5. Поверніть палець ближче до ока;
6. Зосередьтеся на об'єкті далі.
7. Повторити три рази.

Вправа рухи очей

1. Закрийте очі;

2. Повільно рухайте очі вгору, потім вниз;

3. Повторіть тричі;

4. Повільно рухайте очима вліво, потім вправо;

5. Повторіть тричі.

Малюнок 8

1. Зосередьтеся на ділянці на підлозі на відстані приблизно 2.5 метри;

2. Перемістіть очі у формі цифри 8;

3. Обведіть уявну цифру 8 протягом 30 секунд, потім змініть напрямок.

Віджимання олівцем

1. Тримайте олівець на відстані витягнутої руки, між очима.

2. Подивіться на олівець і намагайтеся зберегти єдине зображення, повільно рухаючи його до носа.

3. Рухайте олівець до носа, поки олівець не перестане бути єдиним зображенням.

4. Розташуйте олівець у найближчій точці, де це все ще одне зображення.

5. Повторити 20 разів.

Струна Брока

Щоб виконати цю вправу, людині знадобиться довга нитка і кілька кольорових намистин. Цю вправу вони можуть виконувати як сидячи, так і стоячи.

1. Прикріпіть один кінець мотузки до нерухомого предмета, інакше його може утримати інша людина.

2. Тримайте інший кінець шнурка трохи нижче носа.

3. Помістіть одну намистину на нитку.

4. Подивіться прямо на намистину з відкритими обома очима.

Якщо очі працюють правильно, людина повинна побачити намистину і дві нитки у формі Х.

Якщо закрити одне око, одна з струн зникне, а це означає, що око пригнічується. Якщо людина бачить дві намистини і дві нитки, очі не зближуються до намистини.

Висновки до розділу 4

У ході виконання спеціальної частини до дипломної роботи з охорони праці було проаналізовано щодо організації праці працівників на підприємствах і в установах, а також щодо правил роботи з електронно-обчислювальними приладами.

Попит на IT-професії з кожним роком стає все більшим: програмісти, дизайнери, тестувальники, розробники баз даних, менеджери проектів — лише деякі з тих професіоналів, які потребують бізнесу як ніколи. Зростання попиту на таких спеціалістів було очевидним і до глобальної пандемії коронавірусу, але реальне життя фактично призвело до дефіциту людей, які можуть виконувати таку роботу на високому рівні. Люди такого види діяльності проводять багато часу з екранними пристроями у сидячому положенні і тому охорона праці також важлива в IT.

За допомогою проведення цієї роботи були освоєні права та обов’язки робітників та роботодавців. Також було детально розглянуто умови середовища, у яких працівники мають виконувати свої професійні обов’язки.

ВИСНОВКИ

Під час виконання кваліфікаційної бакалаврської роботи було досліджено сферу мобільних застосунків-сервісів, проаналізовано освітню сферу та її складову застосунків, проаналізовано та обрано інструменти для розробки проекту, розроблено мобільний застосунок розкладу занять для університету.

Після дослідження застосунків-сервісів було визначено, що використання мобільних застосунків серед користувачів мобільних пристроїв стає все більш і більш популярним. Застосунки-сервіси продовжують отримувати підтримку користувачів, тому що вони значно полегшують повсякденне життя. На зараз серед застосунків-сервісів велику популярність мають банківські застосунки, застосунки доставки, та застосунки надання різних послуг, починаючи з державних, закінчуючи послуги магазинів та організацій.

Було розглянуто декілька мобільних застосунків розкладу занять. У ході аналізу було зазначено основний функціонал застосунків, а саме перегляд розкладу занять, можливість вибору тижня, перегляд мапи розташування корпусів університету, перегляд інформації викладачів.

Після аналізу інструментів було обрано платформу Android через її актуальність в Україні, так як за проаналізованими статистичними даними було виявлено, що Android займає майже 82% ринку в Україні, а також має найбільш привабливі функції операційної системи, які можна використати під час розробки програми. Середовищем розробки проекту було обрано Android Studio, так як був спеціально розроблений для android-девелоперів та має усі необхідні функції для розробки проекту. Обрана мова програмування – Java. Java вже багато років є одною з провідних мов програмування, яка має багато переваг. Як серверну частину проекту, а саме структуроване зберігання даних, було обрано базу даних MySQL – одна з найкращих серверних SQL платформ,

яка може опрацьовувати великий потік даних і не потребує сильного догляду зі сторони розробника.

Під час розробки проекту було розроблено базу даних, яка містить таблиці з викладачами, академічними ступенями, вченими званнями, посадами, факультетами, кафедрами, групами, підгрупами, днями тижня, тижнями та розкладом занять. Було створено зручний, інтуїтивний і красивий інтерфейс користувача. Програмно було реалізовано функції за поставленим завданням:

- перегляд розкладу занять, з можливістю вибору групи і тижня;
- перегляд розкладу дзвінків та годин роботи структур університету;
- функція пошуку викладача;
- перегляд мапи університета.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Google Play. Дія: вебсайт. URL: https://play.google.com/store/apps/details?id=ua.gov.diia.app&hl=uk&gl=US (дата звернення 23.05.2022)

2. Google Play. Приват24: вебсайт. URL: https://play.google.com/store/apps/details?id=ua.privatbank.ap24&hl=uk&gl=US (дата звернення 23.05.2022)

3. Google Play. Монобанк: вебсайт. URL: https://play.google.com/store/apps/details?id=com.ftband.mono&hl=uk&gl=US (дата звернення 23.05.2022)

4. Google Play. Нова пошта: вебсайт. URL: https://play.google.com/store/apps/details?id=ua.novaposhtaa&hl=uk&gl=US (дата звернення 23.05.2022)

5. Google Play. Glovo: вебсайт. URL: https://play.google.com/store/apps/details?id=com.glovo&hl=uk&gl=US (дата звернення 23.05.2022)

6. Скільки разів на день ви перевіряєте телефон: вебсайт. URL: https://life.nv.ua/blogs/skolko-raz-v-den-vy-proverjaete-telefon-amerikanskij-uchenyj-o-pervyh-priznakah-zavisimosti-ot-gadzheta-192903.html (дата звернення 23.05.2022)

7. Google Play. Розклад Львівської Політехніки: вебсайт. URL: https://play.google.com/store/apps/details?id=com.dp.schedulelp&hl=uk&gl=US (дата звернення 23.05.2022)

8. Google Play. Timetable: вебсайт. URL: https://play.google.com/store/apps/details?id=com.gabrielittner.timetable&hl=uk&gl=US (дата звернення 23.05.2022)

9. НТУ ХПІ. Застосунок розкладу занять: вебсайт. URL: http://schedule.kpi.kharkov.ua/getapps/ (дата звернення 23.05.2022)

10. Mobile Operating System Market Share Ukraine. Stat counter Global stats : вебсайт. URL: https://gs.statcounter.com/os-market-share/mobile/ukraine/#monthly-202104-202204-bar (дата звернення 23.05.2022)

11. Охорона праці на підприємстві: що потрібно знати? Веб сайт: https://te.dsp.gov.ua/ohorona-pratsi-na-pidpryyemstvi-shho-potribno-znaty/\ (дата звернення 27.05.2022)

12. Охорона праці в IT компанії. Веб сайт: https://legalitgroup.com/service/ohorona-praci-v-it-kompanii/ (дата звернення 27.05.2022)

13. Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями. Наказ: https://zakon.rada.gov.ua/laws/show/z0508-18#Text (дата звернення 27.05.2022)

14. Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень. Веб сайт: https://dnaop.com/html/2296/doc (дата звернення 27.05.2022)

15. Санітарні норми ультрафіолетового випромінювання у виробничих приміщеннях. Веб сайт: https://dnaop.com/html/2299/doc%20-%D0%A1%D0%9D_4557-88 (дата звернення 27.05.2022)

16. Природне та штучне освітлення. Веб сайт: https://dnaop.com/html/45036/doc-%20%D0%A1%D0%9D%D0%B8%D0%9F_II-4-79_____ (дата звернення 27.05.2022)

17. Санітарні норми мікроклімату виробничих поміщень. Веб сайт: https://dnaop.com/html/34094/doc%20-%D0%94%D0%A1%D0%9D_3.3.6.042-99 (дата звернення 27.05.2022)