МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Інститут післядипломної освіти

Кваліфікаційна робота бакалавра

на тему: Розробка мобільного Android-застосунку

Виконав студент групи **Кн5**
спеціальності 122 Комп'ютерні науки
Засоба Юлія Михайлівна

Керівник **ст. викладач**
Штефан Наталія Зінов'ївна

Консультант **доктор філософії**
Бучинська Ірина Вікторівна

Рецензент **к.т.н., доцент**
Сергієнко Андрій Володимирович

Одеса 2023

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

ОС – операційна система

IDE – Integrated Development Environment

SDK – Software Development Kit

Android – операційна система

Android Studio – середа розробки

Eclipse – інтегрована середа розробки

Figma – сервіс для проектування

Java – мова програмування

IntelliJ idea – середа розробки

Use-Case – варіант використання у мові UML

Xamarin – платформа для розробки ПЗ з відкритим кодом

ВСТУП

З бурхливим розвитком інформаційних технологій, що дозволяють створювати різні мобільні пристрої, ринок програмних продуктів отримує потужний стимул до розвитку та вдосконалення. Мобільні пристрої завжди відрізнялися від «комп'ютерів» тим, що вони завжди увімкнені (їх не потрібно вмикати, щоб почати або закінчити роботу), завжди з нами (не тільки під рукою, а й особисті пристрої) і в курсі (будучи підключеним і повним датчиків).

Сьогодні мобільні телефони еволюціонували від простих пристроїв зв'язку до гаджетів, які можуть робити що завгодно, будь то покупки в Інтернеті, замовлення їжі, користування послугами таксі чи навіть пошук простих маршрутів. Мобільні додатки є основою багатьох компаній у всьому світі. Тому вкрай важливо розробляти якісні програми.

Існує багато мобільних додатків на будь-які випадки, наприклад, пропонують різні функції, щоб допомогти людям заснути та зберегти сон. З'являються нові засоби покращити свій сон за допомогою інтерактивних та керованих програм сну. Сон таємничий. Але програми для його відстеження обіцяють допомогти зрозуміти, коли людина перетинає поріг між пробудженням і сном, і що відбувається між ними.

Програми для відстеження сну аналізують звуки, рухи та поведінку людини під час сну, щоб надати йому чіткий знімок тривалості та якості сну. Ці програми також можуть допомогти визначити, скільки часу користувач проводи у режимі швидкого сну (REM) і скільки разів його турбують протягом ночі.

Найкращі програми для відстеження сну використовують такі дані, як звук, частота серцевих скорочень, час сну чи пробудження, щоб надати точне уявлення про ніч. Багато програм використовують дані з переносних пристроїв, таких як Apple Watch. Деякі програми дозволяють експортувати

дані для можливості поділитися ними зі своїм медичним працівником, щоб отримати допомогу з проблемами сну.

Метою роботи є проектування та розробка мобільного застосунку для поліпшення сну людини, який наддасть функції розумного будильника, доступ до довідника корисних порад та інше.

1 АНАЛІТИЧНА ЧАСТИНА

1.1 Опис предметної області

Хороші звички сну позитивно впливають на продуктивність праці та результативність. Покращуючи здатність аналізувати інформацію, зосереджуватися на важливому та приймати кращі рішення, ризик нещасних випадків і помилок знижується.

Сон відіграє важливу роль у хорошому психічному здоров'ї. Поліпшення сну – чудовий спосіб підвищити стійкість і покращить спосіб боротьби зі стресовими факторами на роботі та вдома.

Достатній сон має важливе значення для фізичного здоров'я, оскільки він дозволяє організму оптимізувати працездатність і відновлення. Це може допомогти зміцнити імунну систему, сприяти загоєнню та відновленню тканин організму та зменшити ризик розвитку ряду проблем зі здоров'ям у довгостроковій перспективі.

1.2 Огляд аналогів

На цьому етапі слід провести аналітичний огляд існуючих мобільних додатків: найкращі програми для сну за якістю, універсальністю та простотою використання. Все це допоможе при постановці задач до дипломного проекту.

«Sleep Cycle»

Цей додаток працює на двох ОС. «Sleep Cycle» надає усі необхідні інструменти, щоб зрозуміти, покращити та насолоджуватися сном:

1) можна дізнатися, що впливає на якість сну та як вона покращується з часом, за допомогою детальної щоденної статистики;
2) будильник, який визначає ідеальний час, щоб розбудити людину;
3) є можливість прослуховувати звукозаписи свого хропіння, розмови під час сну та інших звуків. визначати, чи відчуває людина нерегулярне дихання під час сну.

4) слідкувати за своїми звичками сну за допомогою розширеного відстеження сну, індивідуальних порад і постійного навчання.

Розумний будильник будить поступово та вчасно під час найлегшої фази сну. Таке відчуття, ніби прокидаєшся без будильника. Щоб прокидатися ще м’якше, слід інтегрувати «Sleep Cycle» у розумний годинник для м’якої вібрації.

Рисунок 1 – Скрін додатку «Sleep Cycle»

Цей додаток допомагає засинати легше та продовжувати спати під заспокійливу музику, розслаблюючу атмосферу та медитації під керівництвом. «Sleep Cycle» забезпечує комфорт, щоб зменшити тривожність, заспокоїти дихання та розслабити ваше тіло та розум.

Програму «Sleep Cycle» можна завантажити безкоштовно, але вона має можливість покупки в програмі. Отже, хоча він може бути не на 100% безкоштовним, але в додатку є стільки чудових безкоштовних функцій, які

можна використовувати, тому людям, можливо, навіть не доведеться входити – покупка програми.

Ця програма спрямована на те, щоб відстежувати чийсь сон, а потім використовувати ці дані, щоб допомогти розбудити людину на найлегшій стадії сну, щоб ця людина прокидалася бадьорою та відпочилою, а найголовніше – не дрімучою [1].

Наступний додаток – «SleepScore» (рис. 2), доступно для iPhone або Android.

Рисунок 2 – Скрін додатку «SleepScore»

«SleepScore» відомий своєю інноваційною високоякісною технологією сну, і вони не розчарують у цьому додатку для моніторингу сну. Він працює на основі дуже авторитетної технології, яка використовує мікрофон у смартфоні та використовує можливості динаміка для відстеження та вимірювання частоти дихання та рухів тіла.

Після збору даних «SleepScore» розбиває нічний сон на 32 різні параметри, а потім дає персоналізовані рекомендації щодо сну на основі аналізу.

Подібно до «Sleep Cycle», програму «SleepScore» можна безкоштовно завантажити, а також є набір безкоштовних і преміум-функцій, які можна придбати за місячною та річною підпискою з автоматичним оновленням [2].

Безкоштовні функції:

- вимірюйте та оптимізуйте освітлення та звук у спальні, щоб створити ідеальний «заповідник для сну» та отримати місцеву зовнішню температуру;
- заспокійливі звуки та музика для сну;
- поради, як допомогти людині знову заснути вночі;
- розумний будильник, розроблений, щоб обережно розбудити людину в ідеальний час циклу сну;
- «оцінка сну» кожного дня на основі різних стадій сну, розрахована від 0 до 100;
- детальний огляд 4 стадій сну людини щоранку, включаючи легкий, глибокий, швидкий сон і пробудження;
- SleepScore «checkup» постійно контролює сон і надає сповіщення, якщо виявлено більш серйозні проблеми зі сном, щоб поділитися з лікарем;
- рекомендації щодо продуктів від наших експертів зі сну на основі унікальних моделей сну.

Преміум функції:

- виклики для досягнення цілей, як-от менше прокидатися або мати більше енергії;
- порівняння сну з ідеальною ніччю для людей однакового віку та статі
- розширені алгоритми sleepscore повідомляють, чи слід людині працювати над іншою метою;

- історія сну людини протягом життя та кореляції сну, а не лише останні сім ночей;
- вичерпний звіт про сон, яким можна поділитися з лікарем;
- щотижневі основні моменти, що показують найкращі ночі сну кожного тижня;
- корисні графіки трендів на основі даних про спосіб життя та режим сну людини для прийняття розумніших рішень у денний час.

Мобільний додаток «Calm» (рис. 3) доступний для iPhone або Android, а також для iPhone, iPad і Apple TV.

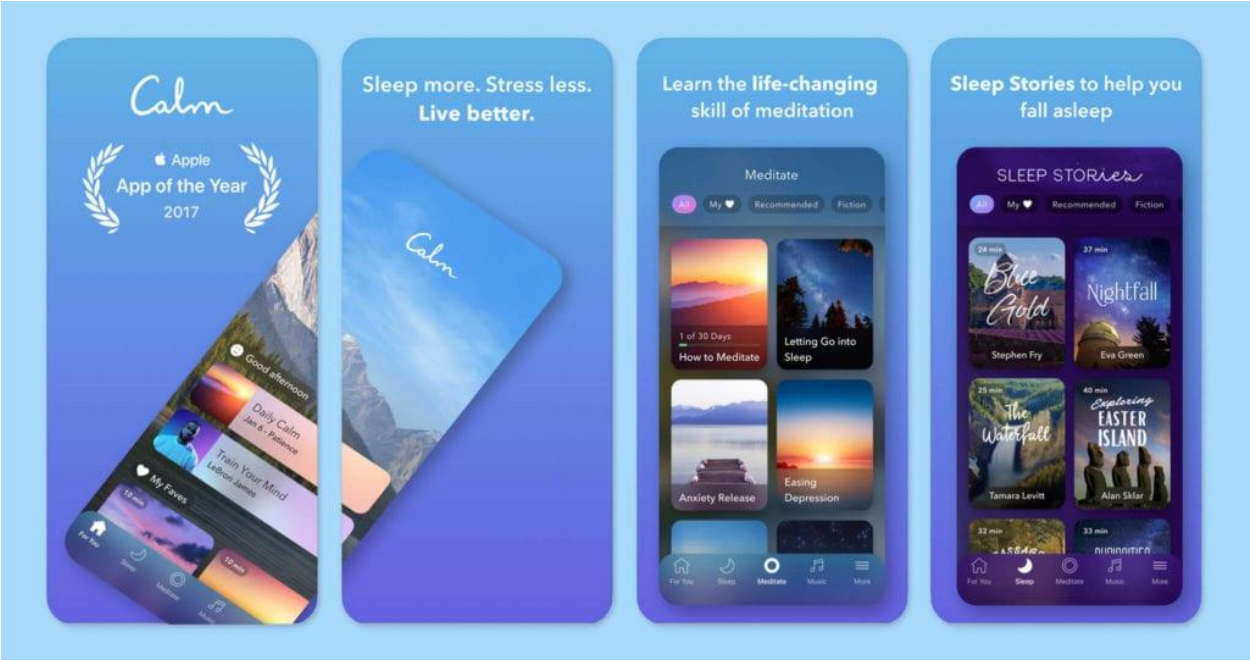

Рисунок 3 – Скрін мобільного додатку «Calm»

«Calm» отримала нагороду Apple як найкраще в 2018 році. «Calm» – це програма для уважності. Тут є щось для всіх рівнів, новачки не злякаються.

«Calm» пропонує медитації для початківців, середнього та просунутих груп, а також історії сну та керовані «рухи сну». Усі ці функції, а також бібліотека звуків для сну гарантовано розслаблять будь-кого в спокійний сон.

«Calm» пропонує безкоштовний пробний період, але згодом людям доведеться платити за доступ у вигляді щомісячних або щорічних платежів з автоматичним оновленням.

Особливості додатку:

- він містить сотні програм для користувачів середнього та досвідченого рівня;
- сеанси медитації під керівництвом доступні тривалістю 3, 5, 10, 15, 20 або 25 хвилин, щоб люди могли вибрати ідеальну тривалість відповідно до свого напруженого графіка;
- у «calm» є понад 100 ексклюзивних історій про сон для дорослих і дітей, у яких представлені такі відомі таланти, як Стівен Фрай, Метью Макконахі, Леона Льюїс і Джером Флінн;
- існує також бібліотека звуків сну, включаючи списки відтворення музики для сну та звуки всіх типів;
- оригінальний daily calm кожен день: 10-хвилинна програма, що додається щодня, щоб допомогти людині полегшити день або розслабитися перед сном;
- спокійний майстер-клас: аудіокласи за участю всесвітньо відомого експерта;
- спокійне тіло: уважне розтягування та рух, щоб розслабити тіло протягом дня [3].

«Sleep» доступно лише для Android. «Sleep» (рис. 4) – це найкраща програма для сну для користувачів Android, оскільки вона призначена саме для них. По суті, це розумний будильник, який також має функцію відстеження сну.

Це допомагає м’яко розбудити вранці в «оптимальний момент для приємних ранків», тобто коли людина не дуже втомлена.

Оскільки в Android є бібліотека функцій, тож давайте розберемо їх:

- відстеження сну за допомогою сонара, тому його можна розмістити на тумбочці, а не в ліжку;

- «розумний будильник», щоб пробудити людину від найлегшого сну
- різноманітність варіантів звуку, включаючи колискові та звуки природи;
- ніжна природа звукові сигнали (птахи, море, шторм...) і списки відтворення;
- колискові зі звуками природи (кити, шторм, море, співи..);
- інтеграція spotify і play music або онлайн-радіобудильники чи колискові;
- перевірка пробудження captcha (математика, підрахунок овець, струшування телефону, qr-код у ванній кімнаті або тег nfc), яку користувач повинен виконати, щоб вимкнути будильник;
- запис розмови під час сну, виявлення хропіння та захист від хропіння.

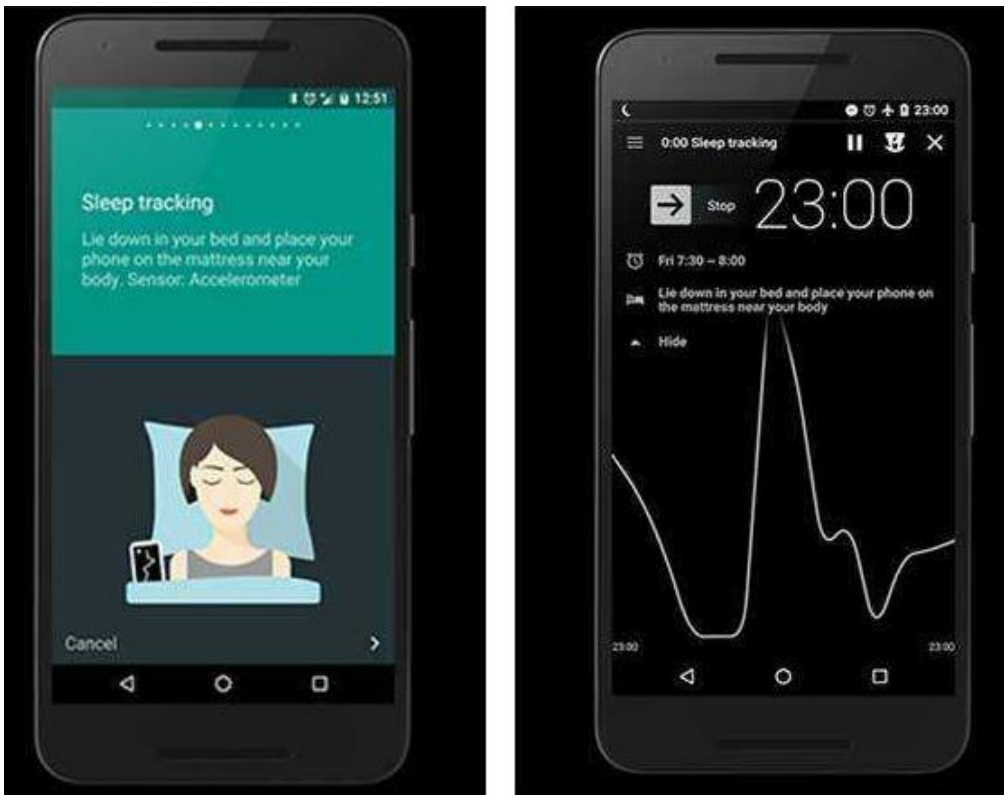

Рисунок 4 – Скрін додатку «Sleep»

Подібно до інших трекерів сну, згаданих у цьому списку, «Sleep» також надає користувачеві «оцінку сну» щоночі. За словами розробників програми, ця оцінка сну – це саме якість сну користувача, яка аналізується за тривалістю, дефіцитом, відсотком глибокого сну, хропінням, ефективністю та нерегулярністю.

«Sleep» надає безкоштовну пробну версію, а потім багато чого можна придбати.

1.3 Вибір інструментальних засобів розробки

Для розробки під ОС Android є три основні IDE: Eclipse (на сьогоднішній день використовується все рідше), IntelliJ idea, Android Studio.

Eclipse IDE це програма для розробки та компіляції додатків на основі мови Java. При використанні в поєднанні з безкоштовним плагіном називається Інструменти Android для розробки (ADT), дозволяє розробляти програми для Android та тестувати їх досить швидко. В якості мови програмування використовується Java.

Дане IDE дуже просте в роботі, тому воно підходить для початкового рівня розробки. ADT розширює можливості Eclipse і дозволяє швидко створювати нові проекти для пристроїв на основі Android.

Поряд з усім іншим створюється користувальницький інтерфейс додатка, додаються компоненти, засновані на Android Framework API, проводиться налагодження додатків за допомогою інструментів Android SDK. Цей комплект засобів розробки (Software Development Kit, SDK) дозволяє створювати додатки для певного пакету програм, ПО базових засобів розробки, апаратної платформи, комп'ютерної системи, ігрових консолей, операційних систем та інших платформ. З його допомогою можна розробляти легкі додатки без програмування [4].

IntelliJ IDEA (рис. 5) – більш серйозний інструмент розробки. Надає такі корисні речі як закриття дужок після умови, груповий перезапис методів та автоматичне створення шаблонних класів (Interface, Singleton). Також тут є можливість зміни теми оформлення. Однак, варто відмітити, що даний проект не є відкритим, однак оновлення виходять дуже часто.

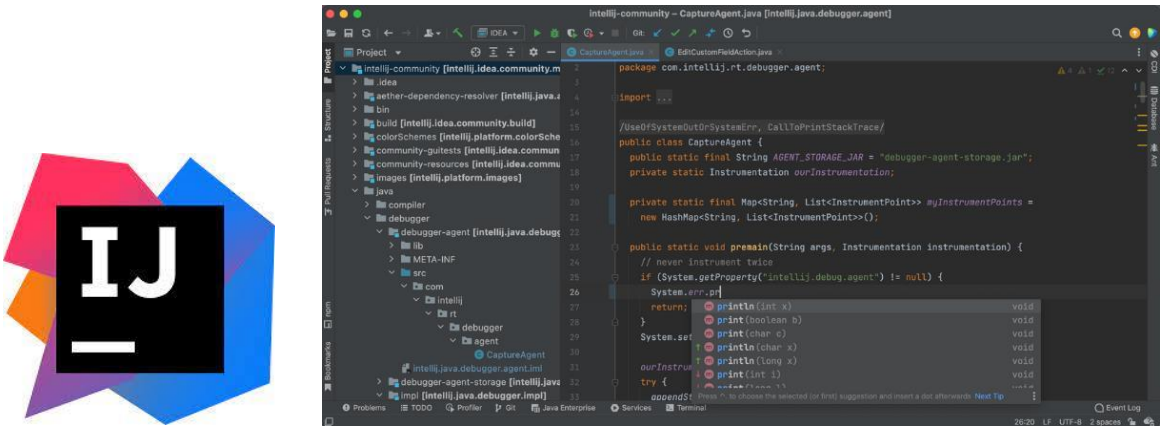

Рисунок 5 – Середа розробки IntelliJ IDEA

Перша версія IntelliJ IDEA з'явилася у січні 2001 року й швидко здобула популярність, як перша Java IDE із широким набором інтегрованих інструментів для рефакторингу, що дозволяла програмістам швидко реорганізовувати сирцевий код програм. Дизайн середовища орієнтовано на продуктивність праці програмістів, дозволяючи їм сконцентруватися на розробці функціональності, тоді як IntelliJ IDEA бере на себе виконання рутинних операцій.

Починаючи з шостої версії продукту IntelliJ IDEA надає інтегрований інструментарій для розробки графічного користувацького інтерфейсу. З версії 9.0 є безкоштовний варіант Community Edition з відкритими кодами. Сирцеві коди відкритої версії IntelliJ IDEA Community Edition поширюються рамках ліцензії Apache 2.0. Бінарні пакунки підготовлені для Linux, Mac OS X і Windows.

До складу IntelliJ IDEA включені напрацювання, створені в результаті спільної роботи з компанією Google, яка використовувала IntelliJ IDEA як базис для своєї нового відкритого середовища розробки Android Studio. Завдяки співпраці істотно розширені штатні можливості IntelliJ IDEA з розробки застосунків для платформи Android.

Android Studio (рис. 6) прийшло на зміну плагіну ADT для платформи Eclipse. Середовище побудоване на базі вихідного коду продукту IntelliJ IDEA Community Edition, що розвивається компанією JetBrains. Android Studio розвивається в рамках відкритої моделі розробки та поширюється під ліцензією Apache 2.0.

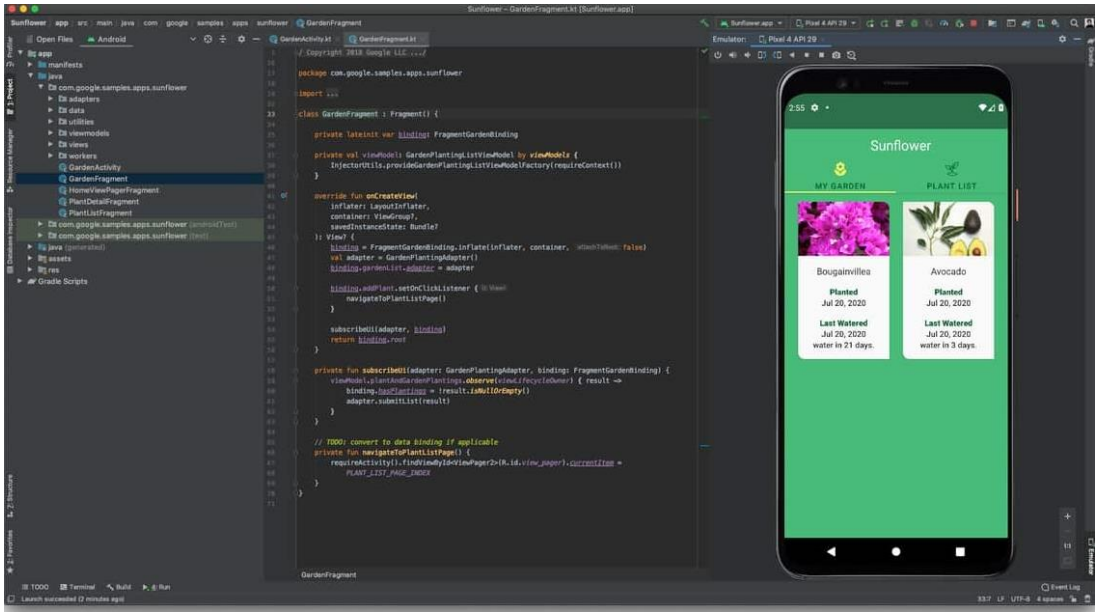

Рисунок 6 – Середа розробки Android Studio

Середовище надає засоби для розробки застосунків не тільки для смартфонів і планшетів, але і для носимих пристроїв на базі Android Wear, телевізорів (Android TV), окулярів Google Glass і автомобільних інформаційнорозважальних систем (Android Auto). Для застосунків, спочатку розроблених з використанням Eclipse і ADT Plugin, підготовлений інструмент для автоматичного імпорту існуючого проекту в Android Studio.

Середовище розробки адаптоване для виконання типових завдань, що вирішуються в процесі розробки застосунків для платформи Android. У тому числі у середовище включені засоби для спрощення тестування програм на сумісність з різними версіями платформи та інструменти для проектування застосунків, що працюють на пристроях з екранами різної роздільності (планшети, смартфони, ноутбуки, годинники, окуляри тощо). Крім можливостей, присутніх в IntelliJ IDEA, в Android Studio реалізовано кілька додаткових функцій, таких як нова уніфікована підсистема складання, тестування і розгортання застосунків, заснована на складальному інструментарії Gradle і підтримуюча використання засобів безперервної інтеграції.

Для прискорення розробки застосунків представлена колекція типових елементів інтерфейсу і візуальний редактор для їхнього компонування, що надає зручний попередній перегляд різних станів інтерфейсу застосунку (наприклад, можна подивитися як інтерфейс буде виглядати для різних версій Android і для різних розмірів екрану) [5].

Для створення нестандартних інтерфейсів присутній майстер створення власних елементів оформлення, що підтримує використання шаблонів. У середовище вбудовані функції завантаження типових прикладів коду з GitHub. До складу також включені пристосовані під особливості платформи Android розширені інструменти рефакторингу, перевірки сумісності з минулими випусками, виявлення проблем з продуктивністю, моніторингу споживання пам'яті та оцінки зручності використання.

У редактор доданий режим швидкого внесення правок. Система підсвічування, статичного аналізу та виявлення помилок розширена підтримкою Android API.

Порівнюючи засоби розробки під мобільні платформи було б неправильно не пригадати ще один. Xamarin (рис. 7) – це один з найстаріших крос-платформних фреймворків такого типу.

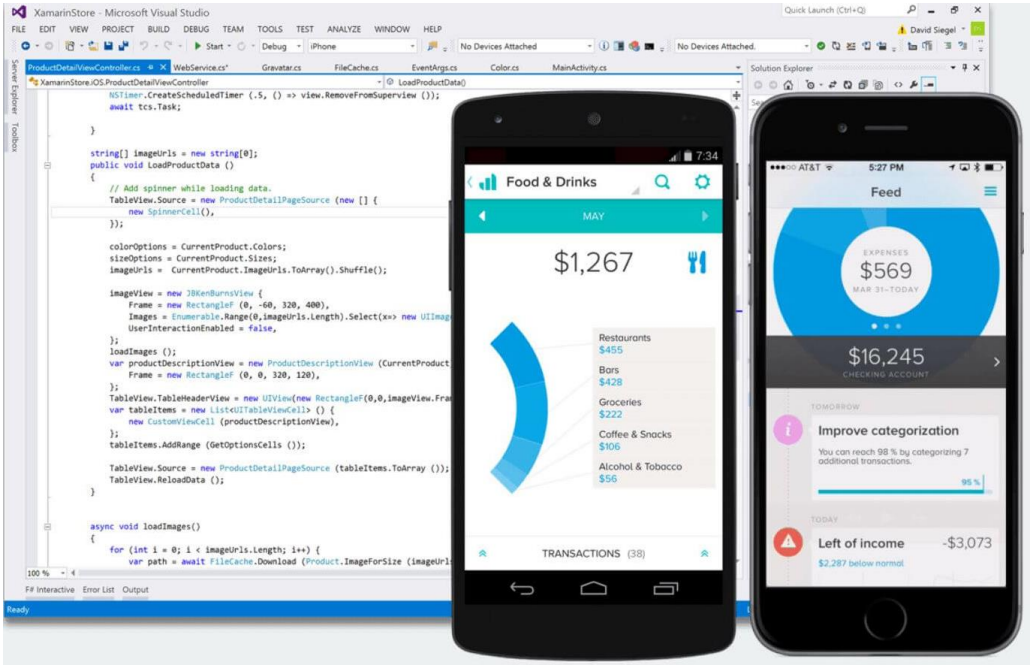

Рисунок 7 – Скріншот розробки додатку у Visual Studio та Xamarin

Проект був придбаний компанією Microsoft у 2016 році та став частиною їх IDE Visual Studio, що забезпечило платформі стабільний подальший розвиток. Це одна з ключових причин, чому великі компанії, такі як, наприклад, Pinterest, покладаються на Xamarin. Він пропонує єдину мову – C#, бібліотеку класів і runtime, що працює на трьох мобільних платформах: iOS, Android та Windows Phone (рідна мова Windows Phone і є C#). Швидкодія та продуктивність продукту на виході достатні навіть для вимогливих ігор [5].

Для реалізації дипломного проекту було прийнято рішення використовувати Android Studio як інструментальний засіб.

1.4 Постановка завдання

Завершуючим етапом аналітичного розділу є постановка завдань до проекту розробки на основі аналізу існуючих аналогів та інструментальних засобів розробки для таких проектів.

Постановка завдання:

1. Визначити функціональні вимоги до проекту.
2. Спроектувати структуру додатку та розробити макети сторінко відповідно до вимог. Для цього використовувати сервіс Figma.
3. Реалізувати основний функціонал додатку: будильник, програвання обраних треків та інше. Використовувати середу розробки Android Studio та мову програмування Java.
4. Після тестування проекту описати його подальший розвиток.

2 ПРОЕКТУВАННЯ ДОДАТКУ

Створення дизайну мобільних програм починається зі збору інформації та вимог. Наступний крок – створення архітектури та навігації. На цьому етапі створення дизайну мобільних додатків основне завдання – це продумати інформаційну та навігаційну архітектуру, тобто. з яких елементів складатиметься продукт, які функції повинні бути закладені і як користувач взаємодіятиме з ними. Варто уточнити складності реалізації та інші технічні нюанси, пов'язані з дизайном та функціоналом мобільного додатку.

Також цьому етапі пишуться користувальницькі сценарії та історії від першого дотику до здійснення цільового дії, тобто, відбувається моделювання програми.

Сценарії використання (use cases) визначають необхідний функціонал та екрани для майбутньої програми. Потім малюють блок-схеми для візуалізації цих історій та сценаріїв та створюють малодеталізовані макети графічного інтерфейсу мобільного додатка.

2.1 Вимоги до проекту розробки

Вимоги до функціоналу:

- можливість встановлювати будильник на обраний час та день. Врахувати функцію повторювання обраних налаштувань для кожного дня;
- панель налаштувань: як і у розглянутих аналогів слід спроектувати розділи «Персоналізації», «Підключення та спільний доступ» для майбутньої можливості зчитувати дані зі смарт-годинника, «Налаштування дісплею», «Сповіщення», «Звуки та вібрація» тощо;
- з урахуванням того, що мобільний додаток проектується під можливості подальшого підключення до смарт-браслету, слід додати збір інформації

щодо стану сну та його тривалості, а результати виводити у вигляді графіку;

- програвання треків колискових пісень з налаштуванням рівня висоти звуку;
- встановлення цілі щодо сну за допомогою нагадувань (ведення журналу, корисні звички, медитація);
- поради для забезпечення корисного сну та відпочинку людини.

Вимоги до дизайну: для того, щоб на виході вийшов якісний продукт, який буде відповідати первісній ідеї і задуму, дуже важливо: правильно виконати проектування, і записати всі елементи інтерфейсу мобільного додатка, промалювати прототипи окремих екранів, попрацювати логіку різних елементів меню і кнопок, описати кількість екранів, їх функції, і продумати поекрано діаграму переходів між ними (мокап /прототип).

2.2 Проектування макетів

Після затвердження ідеї та обговорення архітектури мобільного додатка, створюються мокапи (макети) – низько деталізований прототип, щоб визначити пріоритет та розташування елементів інтерфейсу на екрані, передбачити для них зручне для доступу місцезнаходження. Для створення прототипів буде використано сервіс Figma. За результатами опису функціональних вимог прийнято рішення розробити мобільний застосунок, який буде містити 9 активіті (ектанів).

Слід пам’ятати, що макети дизайну варто створювати під звичайну роздільну здатність з їх щільністю пікселів. Значок і графіку малювати у векторному і використовувати svg [6].

Заставка – це перший екран мобільного додатку, коли користувач запускає програму. Ці екрани грають вирішальну роль, оскільки це шанс залучити та зацікавити користувачів програми. Якщо початковий досвід

користувача буде приємним, він із задоволенням подивиться інші частини додатку. Інакше є шанси втратити нового клієнта [7].

Зазвичай заставка є логотипом, розташованим по центру екрану (рис. 8).

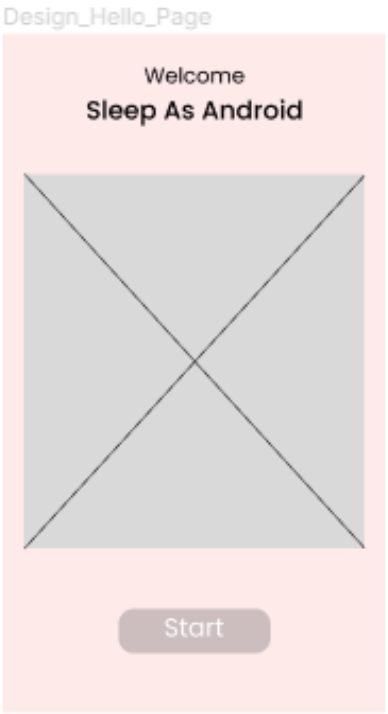

Рисунок 8 – Прототип стартового актівіті

Наступне вікно – «Будильник», тут користувач може обрати час на день сигналу. Кнопка підтвердження налаштувань повинна буди розміщено по центру унизу сторінки. Крім того, починаючи з другого актівіті слід розмістити зверху праворуч кнопку переходу до меню додатку.

Є декілька правил при розробці макетів: слідкувати за розмірами текстових блоків та модулів, а також контролювати значення відступів як від границь самого актітіві, так і від розташованих на ньому інших елементів. Ідеально, коли значення кратні 8ми чи 10ти. Для свого проекту обрано значення «8». Так буде легше описувати файл маніфесту під час розробки та програмування.

Крім того, слід давати всім об'єктам осмислені назви, так буде зрозуміло який об’єкт за що відповідає та тим самим чином давати ім’я об’єктам під час розробки у Android Studio.

Здається, якщо текстовий блок набагато більше, ніж потрібно, то нічого страшного не станеться – там же все одно немає тексту. Необхідно, щоб у тексту завжди було увімкнене налаштування автоматичного вирівнювання по висоті.

Приклад макету другої сторінки представлено на рисунку 9. Сервіс Figma дозволяє слідкувати за відступами між елементами на границями вікна.

Рисунок 9 – Макет активіті «Будильник» та приклад роботи з відступами між його елементами

У нижній частині екрану буде розміщено три кнопки для швидкого переходу до основних активіті: «Dashboard» та «Graphs». Зверху праворуч – меню, яке містить перелік всіх активіті додатку.

Наступна активіті – «Dashboard» (панель приладів) (рис. 10). Ця сторінка буде включати перелік функцій щодо «Personalizations», «Connection& Sharing», «Display&Brightness», «Notifications», «Vibrations».

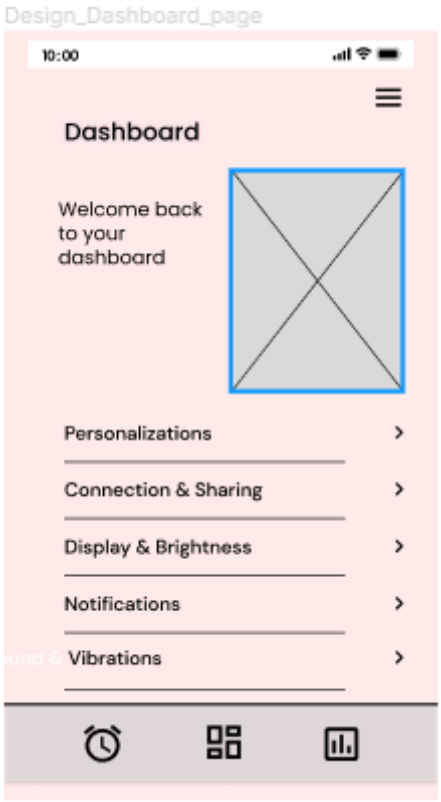

Рисунок 10 – Макет активіті «Панель налаштувань»

Активіті «Graphs» необхідна для слідкування за даними та виводу статистики. Через цю сторінку користувач зможе передивлятись «історію» та «якість» свого відпочинку завдяки взаємодії мобільного додатку та смарт-браслету.

Графік буде відображати відношення кількості годин відпочинку та годин активного життя. Планується у подальшому розвитку проекту додати один з алгоритмів щодо аналізу статистики показників, які використовують оглянуті аналоги.

Так, користувачу буде виводитись повідомлення щодо рекомендацій відпочинку для гарного самопочуття. Макет сторінки представлено на рисунку 11.

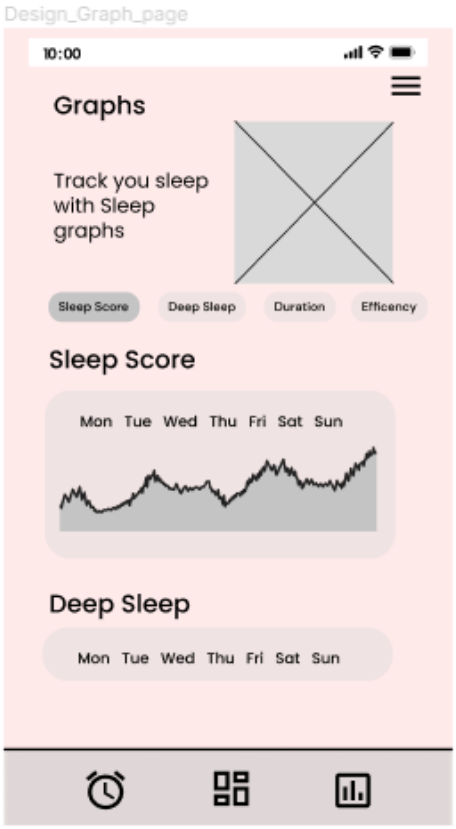

Рисунок 11 – Макет активіті «Graphs»

«Sleep Tracking» ця сторінка активується коли спрацьовує будильник. На екрані повинна бути графічна заставка та кнопка для припинення програвання треку (рис. 12). За замовчуванням, трек буде грати 10 секунд, потім пауза в 10 секунд і знов включиться сигнал доки користувач не натисне кнопку, або пройде ліміт часу. Це допоможе м’яко пробуджувати людину.

Виклик меню – активіті «About» (рис. 13) допомагає користувачу отримати доступ до вибору треків колискових пісень, які будуть розміщені у мобільному додатку, встановлення цілі щодо сну за допомогою сповіщень, ведення журналу дій та перелік рекомендацій щодо корисних звичок, які допоможуть оптимально налагодити сон людини.

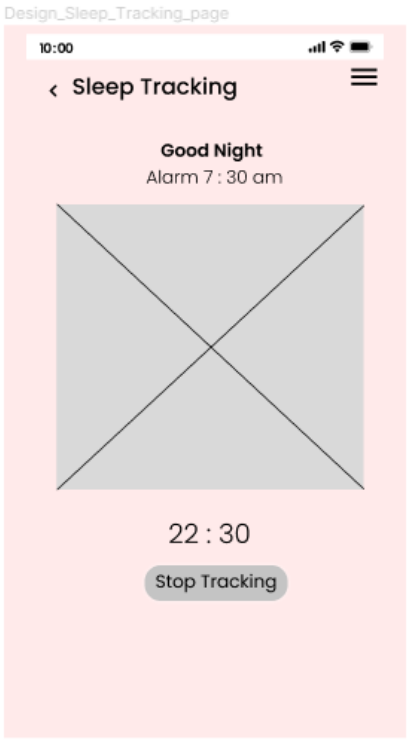

Рисунок 12 – Макет активіті «Sleep Tracking»

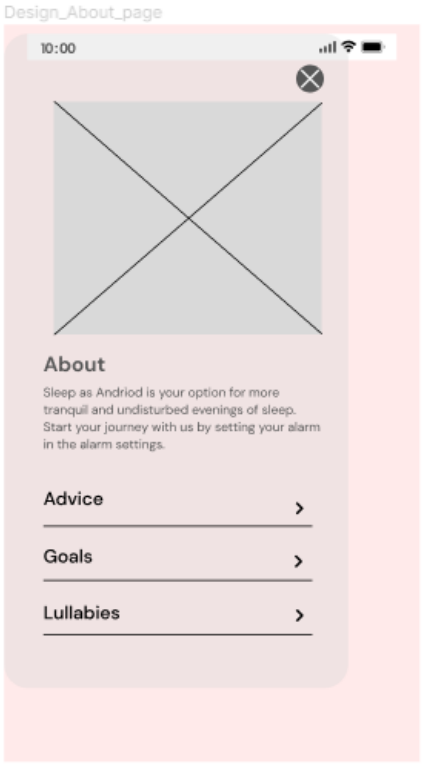

Рисунок 13 – Макет активіті «About»

Макети сторінок «Advice» та «Goals» представлено на рисунку 14.

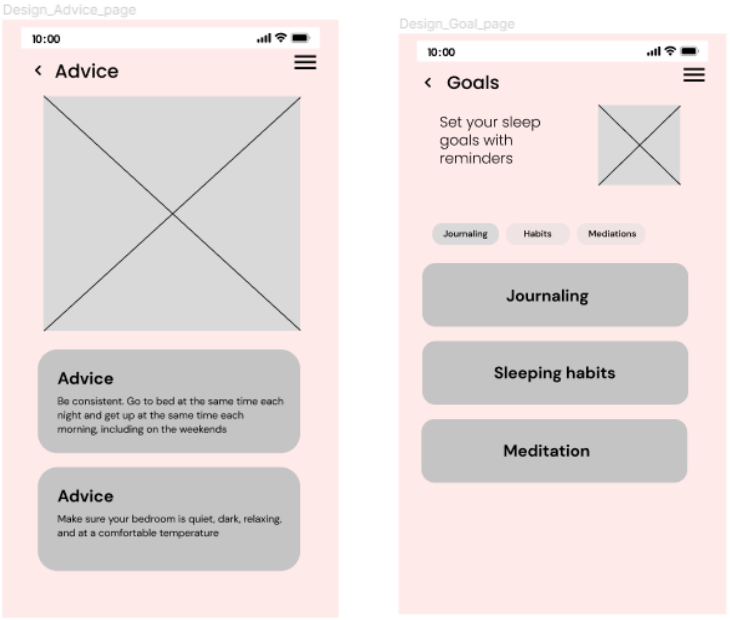

Рисунок 14 – Макети «Advice» та «Goals»

Активіті «Lullabies» буде розміщати елементи доступу до програвання треків, які будуть представлятися користувачу.

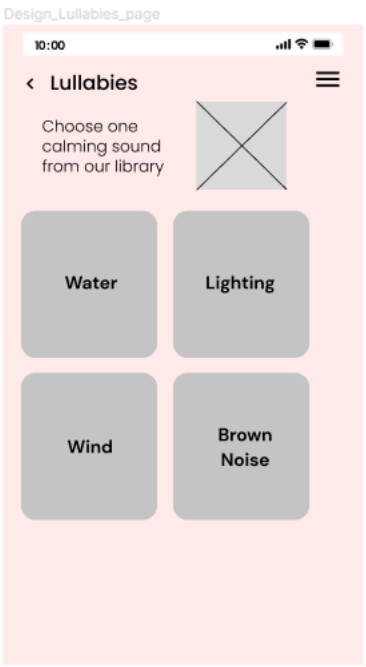

Рисунок 15 – Макет вктівіті «Lullabies»

2.3 Проектування переходів між активіті

Наступний етап в проектуванні – це створення деталізованих, клікабельних прототипів з необхідною динамікою.

Customer Journey Map (Карта подорожей) у Figma показує рух користувача від екрана до екрана, які натискає кнопки. Карта допомагає зрозуміти, як втілити функціонал програми у життя. Такі прототипи стануть фундаментом для розробки додатку у Android Studio.

У сервісі Figma для з'єднання фреймів та налаштування властивостей щодо переходу з однієї активіті до іншої використовується вкладка «Interaction Details». Макети з'єднуються через спеціальні зв'язки, а на вкладці є можливість обрати тип переходу та час (для плавного перелістування сторінок).

Так, та сторінці «About» розміщено меню для переходу на «Advice», «Goals», «Lullabies». Приклад налаштування переходів показано на рисунку 16.

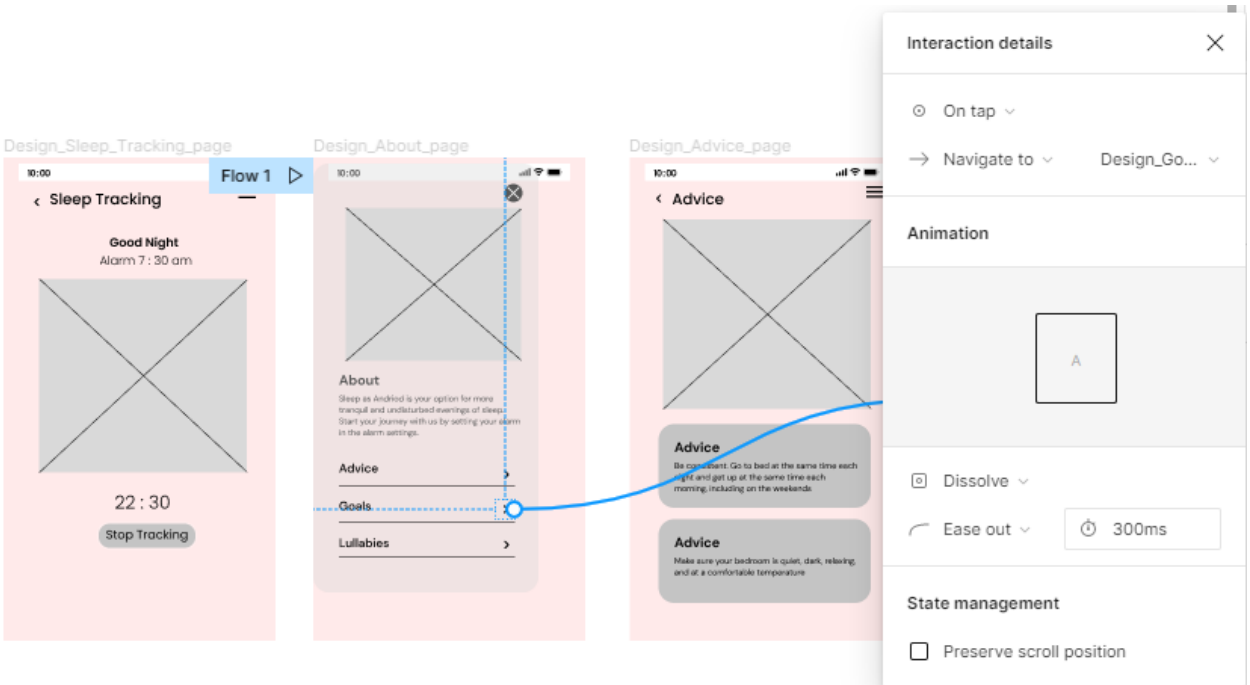

Рисунок 16 – Налаштування переходів між активіті

Взаємозв’язок між всіма актівіті меню «About» представлено на рисунку 17.

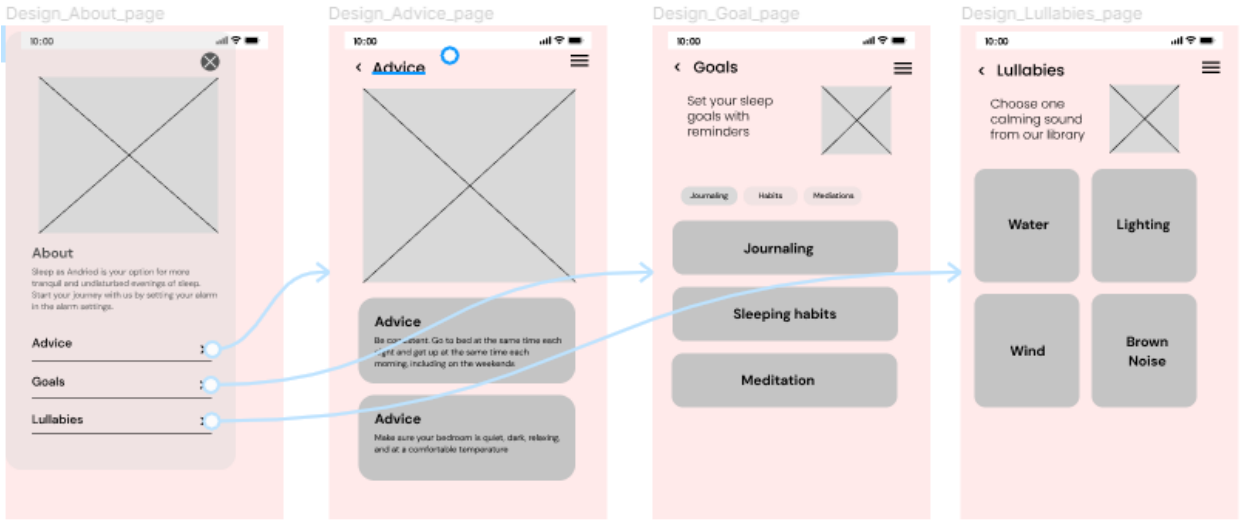

Рисунок 17 – Повний приклад переходів у меню «About»

У футері сторінок «Wake Up», «Dashboard», «Graphs» є кнопки для переходу між цими сторінками (рис. 18). Тому, за правилами проектування макетів додатку, слід встановити відповідні зв’язки – це допоможе уникнути помилок під час програмування у Android Studio.

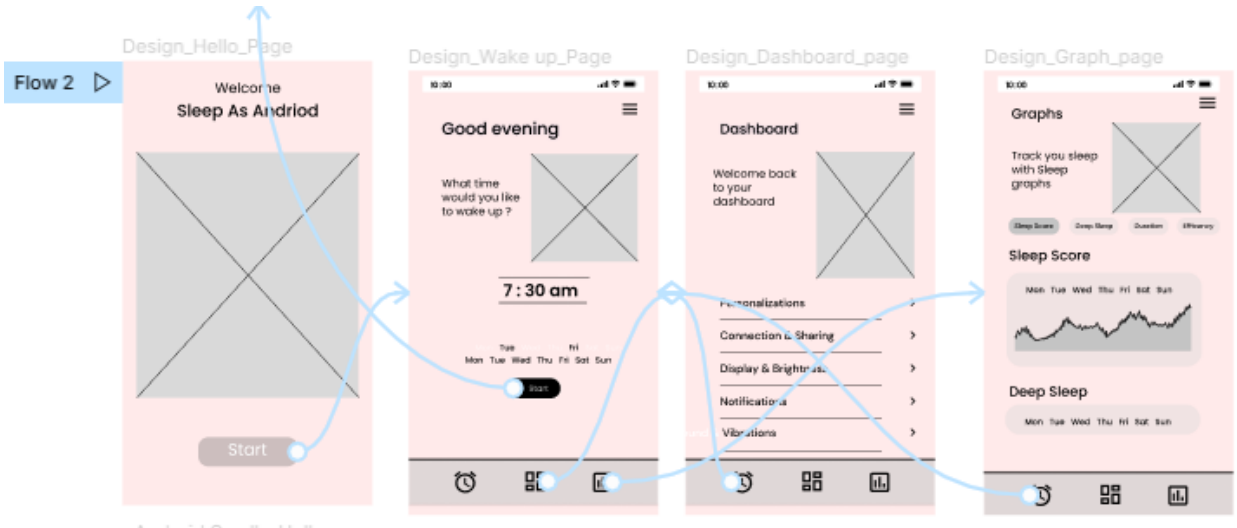

Рисунок 18 – Налаштування переходів по гарячим кнопкам

2.4 Діаграма Use-Case для мобільного додатку

В уніфікованій мові моделювання (UML) діаграма варіантів використання може узагальнити деталі користувачів системи (також відомих як актори) та їх взаємодію з системою. Щоб створити його, слід використовувати набір спеціалізованих символів і з'єднувачів.

Діаграму Use-Case для об'єкту розробки представлено на рисунку 19.

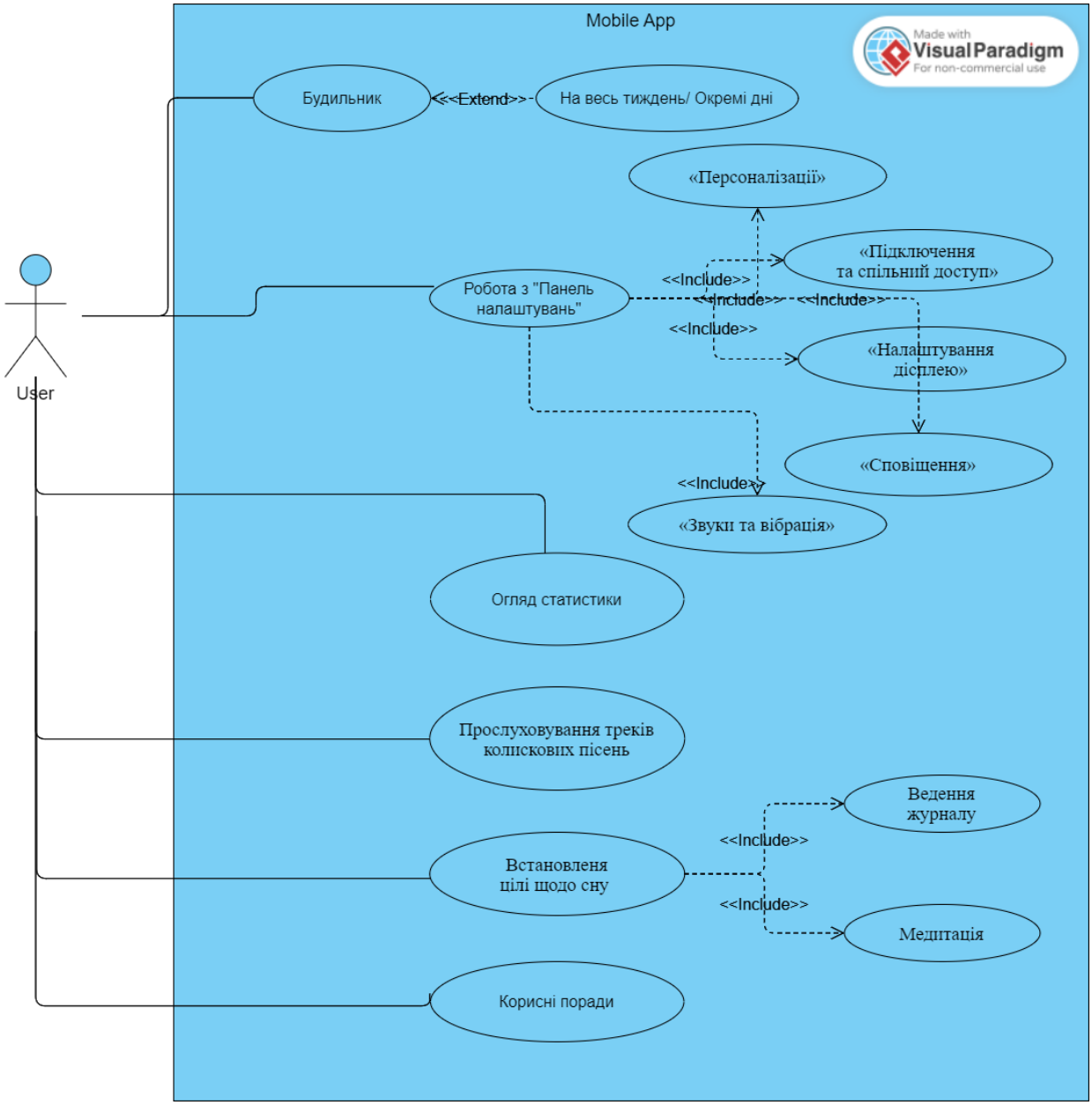

Рисунок 19 – Діаграма Use-Case

Ефективна діаграма варіантів використання може допомогти представити:

- сценарії, у яких система або програма взаємодіє з людьми, організаціями або зовнішніми системами;
- цілі, яких система або програма допомагає цим об’єктам (акторам) досягти;
- область системи.

На цьому етап проектування закінчено. Спроектовані макети, відображенні зв’язки між сторінками та побудовано діаграму варіантів використання. Наступний крок – розробка мобільного додатку у Android Studio та програмування основних його функцій.

3 РЕАЛІЗАЦІЯ

3.1 Опис інтерфейсу

В даний час більшість розробників підтримує стилі дизайнів інтерфейсів, заданими платформи: «Material Design» для Android і «Human Interface Guidelines» для iOS. Незважаючи на переважаючий мінімалізм, вони дозволяють отримати відомі кольори бренду, при необхідності оформити все з використанням брендових елементів. Якщо у замовника є які-то особисті побажання – вони враховуються і переносяться в додаток.

По-перше, було обрано візуальний контент для проекту розробки, а саме графічні елементи для основних активіті (рис. 20):

Рисунок 20 – Графічні елементи для дизайну додатку

Як основний фон використовувався колір «RGB=360476», колір для кнопок «RGB=845EFF», шрифт – «Poppins». Всі активіті формуються відповідно до проектованих макетів. Так Стартова активіті має вигляд (рис. 21).

Рисунок 21 – Скрін стартової активіті

За правилами розробки всі шрифти, кольори та тексти, які використовуються у мобільному додатку, повинні бути прописані у окремих файлах. Це дозволяє легко шукати потрібний елемент та використовувати його у коді через його id. Приклад файлу для кольорів, а також перелік інших файлів директорії « /values» представлено на рисунку 1:

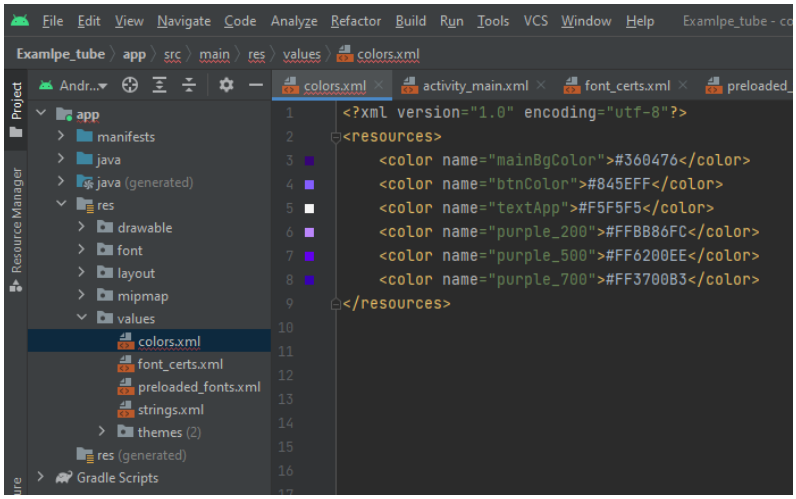

Рисунок 22 – Перелік файлів директорії «values» проекту

Всі властивості слід враховувати у файлі маніфесту «activity_main.xml». Розгляаснемо його більш детально.

```
<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/mainBgColor"
    tools:context=".MainActivity">
```

На сторінці розміщено два елемента для тексту, зображення та кнопка переходу до наступного активіті.

```
 <TextView

    android:id="@+id/textView_Welcome"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:fontFamily="@font/poppins"
    android:textColor="@color/text_app"
    android:textSize="18sp"
    android:text="Welcome">
</TextView>

<TextView
    android:id="@+id/textView_Sleep_with_Adnroid"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:fontFamily="@font/poppins "
    android:textColor="@color/text_app"
    android:textSize="22sp"
    android:text="Sleep as Android">
</TextView>

 <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:contentDescription="@string/sales_icon"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:srcCompat="@drawable/ic_hot_deal"/>
```

```
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Stary"
        android:id="@+id/btnStart">
    </Button>

</LinearLayout>
```

Розглянемо реалізації функціоналу стартового активіті. Для обробки події натискання кнопки у файлі activity_main.xml (файл макета) слід додати дію «android:onClick» для атрибута <Button>:

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_start"
    android:onClick="startApp" />
```

Тут "startApp" – це ім'я методу в активіті, яке викликає система, коли користувач натискає кнопку.

Далі, у класі «MainActivity», який розташований у каталозі проекту src/, додамо наступне:

```
import android.content.Intent;
public void startApp (View view) {
```

Створюємо Intent для запуску активіті під назвою «ClockActivity»:

```
    Intent intent = new Intent(this, ClockActivity.class);
    startActivity(intent);
```

Викликаємо startActivity() и передаємо в нього Intent. Система отримує цей виклик та запускає екземпляр Activity, який вказано в Intent.

Intent це об'єкт, який забезпечує зв'язування окремих компонентів під час виконання (наприклад, двох активіті). Intent представляє "намір щось зробити". Intent не тільки дозволяє почати іншу Activity, але також може

виконувати зв'язок даних у Activity. В результати відкриється наступне актівіті (рис. 23)

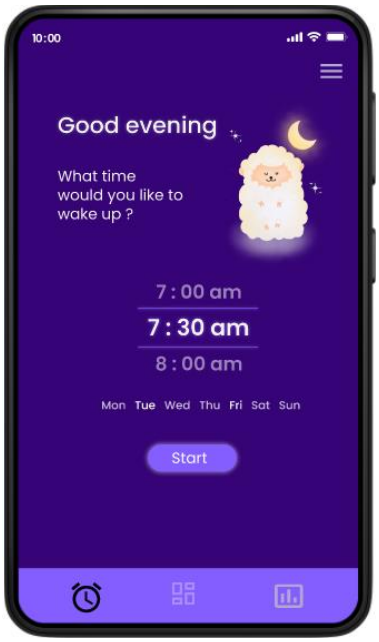

Рисунок 23 – Актівіті для налаштування будильнику

3.2 Опис реалізації основних функцій

Розглянемо реалізацію встановлення будильнику. Для вирішення подібних завдань в Android використовується клас «AlarmManager», який дозволяє виконувати код у потрібний момент часу, навіть якщо ваша програма не запущена.

Клас «AlarmManager» забезпечує доступ до сервісу планування задач Android. Для отримання об'єкта цього класу необхідно викликати метод Context.getSystemService(Context.ALARM_SERVICE).

«AlarmManager» реєструє в системі інтент і коли настає зазначений час, «AlarmManager» запускає цей інтент. Якщо момент виклику програма закрита, вона буде знову запущена.

Для роботи з «AlarmManager» створюється окремий клас, який буде успадкована від базового «BroadcastReciever». Він керуватиме зареєстрованим за допомогою «AlarmManager інтентом». Слід перевизначити метод onReceive(), який буде викликатись після отримання інтенту та отримувати пов'язані з інтентом параметри.

```
import java.text.Format;
import java.text.SimpleDateFormat;
import java.util.Date;

import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.os.PowerManager;
import android.widget.Toast;

publicclass AlarmClock extends BroadcastReceiver{

finalpublic static String ONE_TIME="onetime";
```

Перевизначаємо метод при отриманні сигналу про спрацювання повідомлення:

```
@Override
public void onReceive(Context context, Intent intent){
PowerManager pm=(PowerManager)
context.getSystemService(Context.POWER_SERVICE);
PowerManager.WakeLock wl=
pm.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK,"Wake Up!");
wl.acquire();
Bundle extras= intent.getExtras();
StringBuilder msgStr=new StringBuilder();
```

Далі перевіряємо параметр ONE_TIME на необхідність виводу повідомлення «One Time Alarm!»:

```
f(extras!=null && extras.getBoolean(ONE_TIME, Boolean.FALSE)){
            msgStr.append("One Time Alarm!");
        }
```

Додаємо дату та час спрацьовування сигналу:

```
    Format formatterи = new SimpleDateFormat("hh:mm:ss a");
        msgStr.append(formatter.format(new Date()));
```

Працюємо з прапорцями: скидання повідомлення.

```
Intent notificationIntent = new Intent(context,
MainActivity.class);
notificationIntent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP |
Intent.FLAG_ACTIVITY_SINGLE_TOP);


    PendingIntent int1 = PendingIntent.getActivity(context, 0,
notificationIntent, PendingIntent.FLAG_UPDATE_CURRENT);
```

Створюємо повідомлення, яке спрацює за перевіркою дати:

```
    NotificationManager nm = (NotificationManager)
context.getSystemService(Context.NOTIFICATION_SERVICE);
        NotificationCompat.Builder mBuilder = new
NotificationCompat.Builder(context)
                .setContentTitle("SMS")
                .setContentText("Alarm")
                .setSmallIcon(R.drawable.ic_launcher)
                .setContentIntent(int1)
                .setAutoCancel(true);//слід очистити
повідомлення при натисканні


        mBuilder.build().flags = Notification.FLAG_AUTO_CANCEL;
        nm.notify(1, mBuilder.build());
```

Наступний крок – це встрановити звук на оповіщення:

```
    Uri nt =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION;
        Ringtone r = RingtoneManager.getRingtone(context, nt);
        r.play();

        wl.release(); //разблокування потоку
    }
```

Наступний метод setAlarm() встановлює будильник, що повторюється, за допомогою методу setRepeating(). Цьому методу потрібно чотири параметри: тип будильника, час запуску (встановлюємо поточний момент), інтервал у мілісекундах, інтент, який буде викликатись при спрацюванні будильника.

```
public void SetAlarm(Context context)
{
   AlarmManager
   am=(AlarmManager)context.getSystemService(Context.ALARM_SERVI
   CE;
   Intent intent=new Intent(context,
                    AlarmManagerBroadcastReceiver.class);
   intent.putExtra(ONE_TIME, Boolean.FALSE);//параметр интента

   PendingIntentpi= PendingIntent.getBroadcast(context,0,
                    intent,0);
```

Встановимо інтервал спрацьовування у 5 секунд:

```
am.setRepeating(AlarmManager.RTC_WAKEUP,
                System.currentTimeMillis(),1000*5,pi);
}
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd
                 HH:mm:ss");
SimpleDateFormat anothersdf = new SimpleDateFormat("yyyy-MM-
                 dd");
sdf.setTimeZone(TimeZone.getDefault());
```

Щоб повідомлення не з'являлося, якщо час спрацьовування вже минув, слід перенести його на наступний день:

```
    Date date = new Date();
        String data = anothersdf.format(date) + " " +
                 inputString;
        Date current = date;
        try {
            date = sdf.parse(data);
        } catch (ParseException e) {

        }
```

```
        Calendar c = Calendar.getInstance();
        c.setTime(date);

        if (date.compareTo(current)<0) {
            c.add(Calendar.DATE, 1);
        }
        date = c.getTime();
        am.setRepeating(AlarmManager.RTC_WAKEUP, date.getTime(),
                AlarmManager.INTERVAL_DAY, pi);
    }
```

Метод cancelAlarm() скасовує зареєстрований раніше будильник за допомогою виклику методу cancel(), якому передається як параметр інтент. При збігу цього параметра із зареєстрованим раніше інтентом, буде видалено будильник.

```
public void CancelAlarm(Context context) {
     Intent intent = new Intent(context, AlarmBroadcast.class);
     PendingIntent sender = PendingIntent.getBroadcast(context,
                0, intent, 0);
     AlarmManager alarmManager = (AlarmManager)
                context.getSystemService(Context.ALARM_SERVICE);
     alarmManager.cancel(sender);
    }
```

Метод onetimeTimer() створює будильник, який спрацьовує один раз. Робиться за допомогою методу set(), якому передається три параметри: тип будильника, час запуску, інтент, що викликається.

```
public void setOneTimeTimer(Context context, String inputString,
                            int id)
{
    AlarmManager am = (AlarmManager)
context.getSystemService(Context.ALARM_SERVICE);

    Intent intent = new Intent(context, AlarmBroadcast.class);
    intent.putExtra(ONE_TIME, Boolean.TRUE);
    PendingIntent pi = PendingIntent.getBroadcast(context, id,
                       intent, 0);

   SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd
                          HH:mm:ss");
   SimpleDateFormat anothersdf = new SimpleDateFormat("yyyy-MM-
                          dd");
```

```
sdf.setTimeZone(TimeZone.getDefault());

Date date = new Date();
String data = anothersdf.format(date) + " " + inputString;
   try {
      date = sdf.parse(data);
     } catch (ParseException e) {

     }
  am.set(AlarmManager.RTC_WAKEUP, date.getTime(), pi);
 }}
```

У файл MainActivity.java додамо створення будильнику та його виклик:

```
AlarmBroadcast alarm = new AlarmBroadcast();
alarm.SetAlarm(getApplicationContext(), time, notid);
```

тут: time – це час у форматі Date, notid – унікальний ідентифікатор будильника, потрібно робити кілька будильників.

На актівіті «Clock» унизу є кнопки для навігації по сторінці «Dashboard» та «Graphs» (рис. 24).

Рисунок 24 – Скрини екранів «Dashboard» і «Graphs»

3.3 Опис функціоналу активіті «About»

На активіті «About» розміщено меню для переходу на «Advice», «Goals», «Lullabies» (рис. 25). Меню в додатках представляє клас android.view.Menu, і кожна діяльність асоціюється з об'єктом цього типу. Меню у Android є ресурсом. Однак при створенні нового проекту з Empty Activity за замовчуванням немає жодних ресурсів меню, тому їх потрібно додавати вручну (у каталозі «res» створюємо Android Resource File «about_menu.xml»):

Рисунок 25 – Скрін ектану з меню «About»

```
<?xml version="1.0" encoding="utf-8"?>

<menu
xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/advice_act"
        android:orderInCategory="1"
```

```
            android:title="Advice" />
        <item
            android:id="@+id/goals_act"
            android:orderInCategory="3"
            android:title="Goals" />
        <item
            android:id="@+id/lullabias_act"
            android:orderInCategory="2"
            android:title="Lullabies" />
    </menu>
```

За макетом проектування – меню з трьома елементами, причому файл x xml тільки визначає елементи, але не створює їх. Щоб вивести меню на екран, слід використовувати його у класі «Activity», а саме шляхом перевизначення методу «onCreateOptionsMenu:

```
public class MainActivity extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override

    public boolean onCreateOptionsMenu(Menu menu) {

        getMenuInflater().inflate(R.menu.main_menu, menu);
        return true;
    }
}
```

Метод getMenuInflater як перший параметр приймає ресурс, що представляє декларативний опис меню в xml, і наповнює їм об'єкт menu.

На активіті «Advice» (рис. 26) будуть розміщені поради для користувача. На даний час там відображається інформація по замовчуванню. У подальшому розвитку проекту для активіті «Advice» та «Goals» (рис. 27) буде підвантажуватися інформаційні блоки з БД.

Рисунок 26 – Скрін Актівіті «Advice»

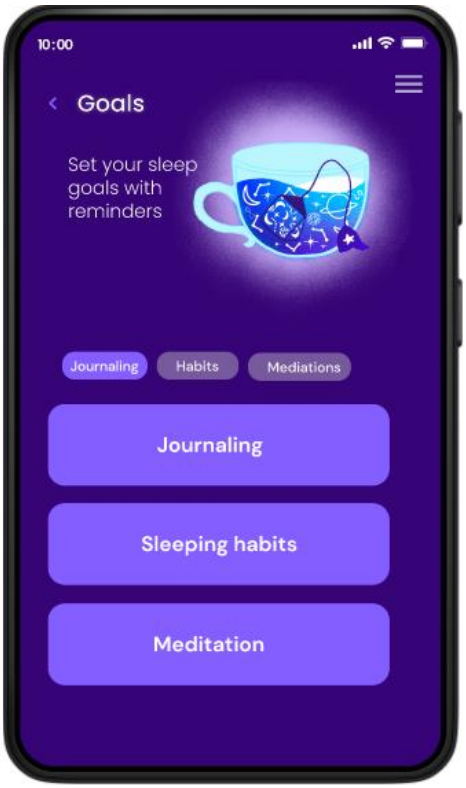

Рисунок 27 – Скрін Актівіті «Advice»

Останні екрани – це «Lullabies» і «Sleep Tracking» (рис. 28).

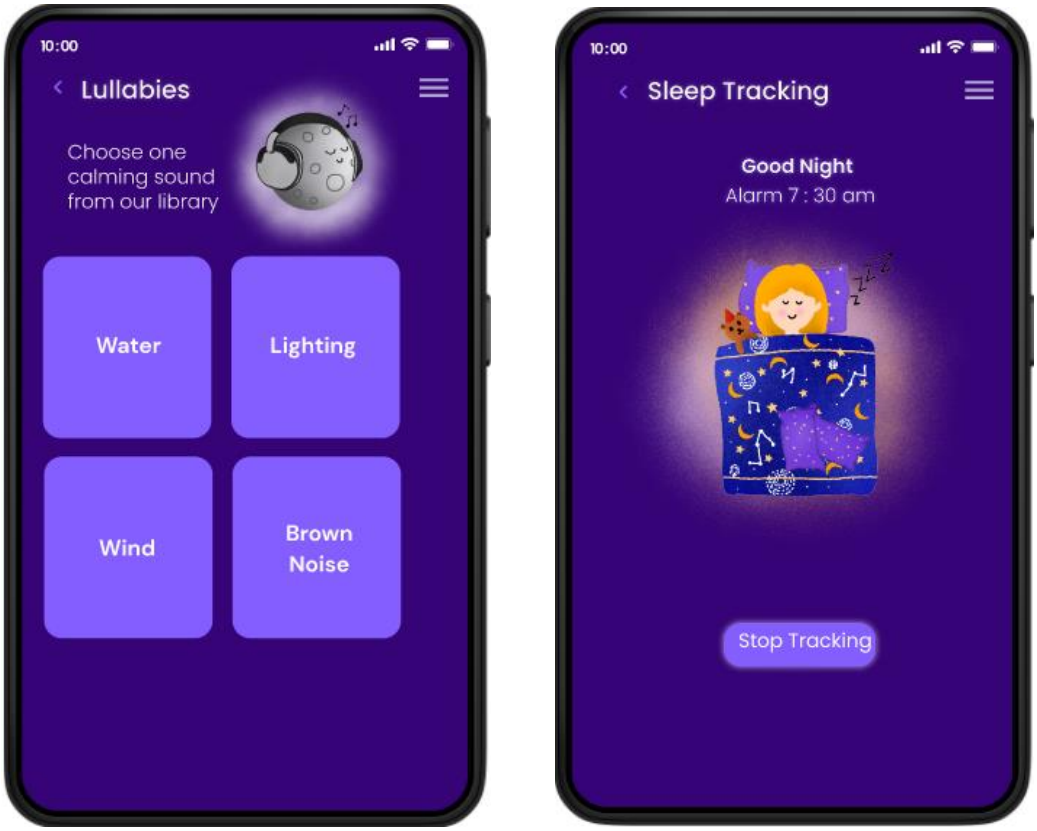

Рисунок 28 – Скріни «Lullabies» і «Sleep Tracking»

На «Lullabies» можна обрати мелодію для програвання на ніч. Після натискання на відповідну кнопку, з’явиться екран «Sleep Tracking» з відповідною кнопкою. За замовчуванням, якщо трек не зупинити під час програвання, він автоматично грає 3 рази та відключається. На даний час у додатку є 4 треки, які зберігаються у папці проекту « res/raw».

Для відтворення аудіо-файлів Android надає клас «MediaPlayer». Щоб відтворювати аудіо, MediaPlayer повинен знати, який ресурс (файл) потрібно виробляти.

Встановити потрібний ресурс для відтворення можна через метод create() об'єкта MediaPlayer: тут передається id ресурсу, що представляє аудіо-файл.

Після встановлення ресурсу викликається метод prepare(). Цей метод готує аудіо-файл до відтворення, витягаючи з нього перші секунди.

Для керування відтворенням у класі MediaPlayer визначено такі методи:

- start(): запускає аудіо;
- pause(): призупиняє відтворення;
- stop(): повністю зупиняє відтворення.

У класі «MainActivity» слід додати наступне:

```
public class MainActivity extends AppCompatActivity {

    MediaPlayer mPlayer;
    Button playButton, stopButton;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mPlayer= MediaPlayer.create(this, R.raw.music);
        mPlayer.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                stopPlay();
            }
        });

playButton = findViewById(R.id.playButton);
        stopButton = findViewById(R.id.stopButton);
        stopButton.setEnabled(false);
    }

    private void stopPlay(){
        mPlayer.stop();
        pauseButton.setEnabled(false);
        stopButton.setEnabled(false);
        try {
            mPlayer.prepare();
            mPlayer.seekTo(0);
            playButton.setEnabled(true);
        }


 catch (Throwable t) {
            Toast.makeText(this, t.getMessage(),
Toast.LENGTH_SHORT).show();
        }
```

```
    }
    public void play(View view){
        mPlayer.start();
        playButton.setEnabled(false);
        stopButton.setEnabled(true);
    }
    public void stop(View view){
        stopPlay();  }
    @Override
    public void onDestroy() {
        super.onDestroy();
        if (mPlayer.isPlaying()) {
            stopPlay();
        }
    }
}
```

4 ТЕСТУВАННЯ ДОДАТКУ

Оскільки смартфони поступово стають обов’язковим, розробники знаходять гарну кар’єру в розробці мобільних додатків. З часом ринки заповнюються мільйонами програм. Але правда полягає в тому, що лише деякі можуть залишити свій слід і керувати мобільним світом.

За такого високого рівня компетенції ви маєте бути впевнені, що, окрім пропозиції чогось інноваційного та цікавого для ваших клієнтів, ваша програма також має бути вільною від будь-яких збоїв. Тому тестування мобільних додатків стає дуже важливим [8].

Вибухове зростання використання мобільних пристроїв і розробка мобільних додатків робить тестування ключовою вимогою для успішної та швидкої доставки високоякісних мобільних додатків. Тестування мобільних додатків – це процес, який має пройти кожен додаток, розроблений для портативних пристроїв, щоб забезпечити певний рівень якості [9].

4.1 Особливості мобільного додатку

Зрозуміло, що мобільний додаток сильно відрізняється від настільного. Тому слід враховувати це при плануванні процесу тестування. Основні відмінності між мобільними та настільними програмами:

1. Мобільний пристрій – це система, яка не має потужної начинки. Отже, він не може працювати як персональний комп'ютер.
2. Тестування мобільного додатку здійснюється на телефонах (Apple, Samsung, Nokia тощо), а настільний додаток тестується на центральному процесорі.
3. Різноманітність екранів мобільних пристроїв, їх розширення та кольори. Розмір екрану мобільного телефону менше, ніж у настільного комп’ютера.

4. Здійснення та прийом дзвінків є основним завданням телефону, тому програма не повинна заважати цій основній функції.
5. ОС мобільних телефонів швидко застаріває. Крім того, існує обмеження на оновлення ОС.
6. Мобільні пристрої використовують мережеві з'єднання (3G, 4G, Wi-Fi), настільні комп'ютери використовують широкосмугове з'єднання або Wi-Fi.
7. Інструменти, які підходять для тестування настільних програм, не зовсім підходять для тестування мобільних програм.

Життєвий цикл розробки мобільних додатків, як правило, набагато коротший, ніж інші, наприклад додатки для настільних ПК. Отже, успішність цих програм значною мірою залежить від тестування мобільних програм.

Мобільні програми тестуються на основі різних параметрів, таких як функціональність, зручність використання, послідовність, продуктивність, безпека та інтерфейс користувача. Це допомагає підвищити загальну ефективність додатків на всіх фронтах, а також підвищити коефіцієнт надійності серед користувачів, які їх використовують [8].

Тестування мобільних додатків допомагає:

- підвищення рівня задоволеності користувачів;
- піднімає імідж бренду;
- досягніть хорошої рентабельності інвестицій;
- досягнення повнофункціональної програми.

4.2 Етапи тестування мобільного додатку

Наскрізне тестування є життєво важливим для безперебійної роботи мобільних додатків. Ефективне наскрізне тестування мобільних додатків складається з наступних кроків:

1. Опис процесу – опис усіх тестів, які необхідно виконати для мобільного додатка, є першим кроком, якого слід виконати. Слід

розробити план тестування з усіма варіантами використання, які необхідно протестувати, пояснюючи тести та очікуваний результат для спринту.

2. Вибір типу тесту для ручного або автоматичного тестування – рішення про те, чи буде тест ручним чи автоматизованим, є наступним кроком.

Тести можна автоматизувати:

- якщо сценарій використання потрібно запускати часто;
- якщо тест має передбачуваний результат;
- якщо тести потрібно писати для різних пристроїв, операційних систем і розмірів екрану;
- для модульного тестування.

Тому на цьому етапі спеціалісти з контролю якості мають визначити сферу ручного тестування проти автоматизованого тестування, щоб оптимізувати тестування та витрати.

3. Підготовка тестів для різних функцій користувача – визначення випадків є першим кроком для написання тестів для мобільних програм, коли вирішите, який тип тестування використовувати.

Тут можна застосувати два підходи:

- тестування на основі бізнес-сценарію: система оцінюється з точки зору бізнесу;
- тестування на основі вимог: оцінюється продуктивність окремих функцій програми.

Визначення тестових випадків також залежить від типу тестування, яке ви хочете виконати. Ручне тестування не вимагає початкових вкладень. Тому рекомендується почати тестовий спринт із дослідницького тестування.

Для дипломного проекту слід провести функціональне тестування – тестування інтерфейсу (відповідність макетів у Figma до вимог) (табл. 1) та основних його функцій (як працюють функції додатку) (табл. 2). Емулятори та симулятори є найстарішими та найпоширенішими інструментами для

тестування мобільних додатків. Це інструменти, які дозволяють вибрати модель мобільного пристрою, імітувати пристрій і запустити його на комп’ютері. Ці функції будуть перевірені через емулятор Android Studio.

Таблиця 1 – Тестування інтерфейсу

Назва	Очікуваний результат	Реальний результат	
Однаковий фон для всіх сторінок	RGB=360476	RGB=360476	+
Колір кнопок у додатку	RGB=845EFF	RGB=845EFF	+
Розроблено всі макети сторінок	Всього 9 макетів	Всього 9 макетів	+
Перевірка покажчика відступім між елементами до границь вікон	Відстань між об’єктами сторінок та границями сторінок кратно 8ми чи 10	Відстань між об’єктами сторінок та границями сторінок кратно 8ми пікселям	+

Таблиця 2 – Тестування функціоналу додатку

Запуск стартового вікна	Запускається активний додаток	Запускається активний додаток	+
Кнопка «Start»	При натискані завантажується активіті «Clock»	При натискані завантажується активіті «Clock»	+

Продовження таблиці 2

Футер сторінки «Clock»	Знаходяться 3 кнопки переходу на інші сторінки	Кнопки активні	+
Меню додатка	При натисканні на іконку меню висвічується вікно з переліком елементів для переходу	Меню активне	+
Будильник	Налаштування на обрану дату – спрацьовує звук сигналу	У відповідний час будильник спрацьовує(рівень звуку відповідає налаштуванню користувача у додатку)	+
Плеєр	Програвання обраного треку	При обиранні треку запускається функція його програвання	+
Сторінка «Корисні поради»	При переході на сторінку відображаються заголовки порад	При переході на сторінку відображаються заголовки порад, які можна розгорнути та передивитись	+

ВИСНОВКИ

Згідно з дослідженнями, більша частина людей часто мають проблеми з засинанням щоночі. Знайти час, щоб відпочити за допомогою розслаблюючої рутини перед сном, важливо для оптимізації якості сну. Медитація та читання можуть бути чудовими інструментами для багатьох, але деякі люди, можливо, віддадуть перевагу використанню програми для сну, яка допомагає в цьому процесі. Тож, розробка мобільного додатку, який буде допомагати людині спостерігати за своїм тілом активного життя та кількості годин сну є актуальним у даний час.

Для виконання роботи на першому етапі визначення вимог до об’єкту розробки були розглянуті популярні аналоги та програмні засоби розробки, після аналізу яких обрано Android Studio і мову програмування Java.

На етапі проектування було спроектовано макети у сервісі Figma, відображенні зв’язки між сторінками та побудовано діаграму варіантів використання. Все це є основою для подальшої розробки.

Наступний крок – розробка мобільного додатку у Android Studio та програмування основних його функцій. Так, реалізовані декілька функцій до основних активіті, а саме: налаштування будильнику, програвання обраного треку. У програмному коді прописані переходи з однієї активіті на інші.

Мобільний додаток протестоване на вимоги до дизайну (відповідність макетам) та на роботу функціональних вимог користувача. Як подальший його розвиток слід вважати додавання до функціоналу додатку БД для зберігання контенту та збору і аналізу інформації щодо стану людини. А також реалізація взаємодії додатку з смарт-браслетом для отримання даних щодо активності користувача та стану його показників. Все це допоможе завершити проект і підготувати його до комерційного просування на ринку серед аналогів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. How Sleep Cycle works. URL: https://www.sleepcycle.com/how-sleep-cycle-works/ (дата звернення 12.04.2023)
2. The most accurate app for improving your sleep. URL: https://www.sleepscore.com/sleepscore-app/(дата звернення 12.04.2023)
3. Calm Review. URL: https://onemindpsyberguide.org/apps/calm/ дата звернення (дата звернення 12.04.2023)
4. Л.С. Глоба, Т. М. Кот. Розробка інформаційних ресурсів та систем. Київ: НТУУ «КПІ», 2014. -318 с.
5. Засоби розробки мобільних додатків. URL: https://stud.com.ua/20598/informatika/zasobi_rozrobki_mobilnih_dodatki (дата звернення 17.04.2023)
6. Порівняльна характеристика актуальних засобів розробки під мобільні платформи. URL: https://core.ac.uk/download/pdf/234094409.pdf (дата звернення 17.04.2023)
7. Розрабка UX/UI-дизайну мобільного додатку. URL: https://appcraft.pro/blog/razrabotka-uxui-dizajna-mobilnogo-prilozheniya/ (дата звернення 23.04.2023)
8. Дизайн мобільних додатків: процес розробки та етапи проектування. URL: https://turumburum.ua/blog/dizayn-mobilnykh-prilozheniy-protsess-razrabotki-i-etapy-proektirovaniya/ (дата звернення 28.04.2023)
9. How to Test a Mobile Application [Step by Step]. URL: https://www.testbytes.net/blog/how-to-test-a-mobile-application/ (дата звернення 18.05.2023)
10. App & Browser Testing Made Easy. URL: https://www.browserstack.com/guide/how-to-test-mobile-applications (дата звернення 25.05.2023)