Міністерство освіти і науки України

Луцький національний технічний університет

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОЗРОБКА ANDROID-ДОДАТКУ ДЛЯ ТАЙМ-МЕНЕДЖМЕНТУ «TIMESAGE»

DEVELOPMENT OF ANDROID APPLICATION FOR TIME MANAGEMENT «TIMESAGE»

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав: здобувач вищої освіти
Групи ІПЗ-42
Цісар Андрій Анатолійович
_____
(підпис)

Керівник:
к.т.н., доцент
Ящук Андрій Анатолійович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«8» 06 2024 р.
к.т.н., доцент
Гарант освітньої програми:
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2024 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *121 Інженерія програмного забезпечення*
Освітня програма: *«Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ *А. Лінник*
«*2*» *02* 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА ПЕРШОГО (БАКАЛАВРСЬКОГО) РІВНЯ ВИЩОЇ ОСВІТИ

Цісар Андрій Анатолійович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: *Розробка Android-додатку для тайм-менеджменту «TimeSage»*

Керівник роботи: *к.т.н., доцент, Ящук Андрій Анатолійович*
затверджені наказом закладу вищої освіти від «*30» грудня 2023 р.* № *477/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи *«5» червня 2024 р.*
3. Вихідні дані до роботи: *технічне та програмне забезпечення ЕОМ, вимоги до розробки програмного забезпечення, ергономічні вимоги до функціонування програмного засобу.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити):
Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення, опис функціонального наповнення об'єкта проектування, практична реалізація об'єкта проектування.

5. Перелік графічного матеріалу: *21 рисунок.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Яцук А. А.*		
Специфікації вимог до розробленої системи	*Яцук А. А.*		
Розробка об'єкта проєктування	*Яцук А. А.*		
Нормоконтроль	*Повстяна Ю. С.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту		29 %	
Академічна доброчесність	*Яцук А. А.*		

7. Дата видачі завдання *«2» лютого 2024 р.*

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*07.02.2024 р.*	вик.
2	*Аналіз проблеми розробки та впровадження об'єкту проектування*	*27.02.2024 р.*	вик.
3	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*12.03.2024 р.*	вик.
4	*Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних*	*17.04.2024 р.*	вик.
5	*Практична реалізація об'єкта проектування та розробка бази даних*	*18.05.2024 р.*	вик.
6	*Тестування та налагодження об'єкта проектування*	*01.06.2024 р.*	вик.
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*05.06.2024 р.*	вик

Здобувач вищої освіти _____ (підпис) *Цісар А. А.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Яцук А. А.* (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти: *Цісар Андрій Анатолійович*

Тема: *Розробка Android-додатку для тайм-менеджменту «TimeSage»*

Коротка характеристика кваліфікаційної роботи:
Кваліфікаційна робота складається з трьох розділів, в яких проведено аналіз предметної області, здійснена чітка постановка завдань за темою роботи, проведено дослідження методів та засобів для розробки Android-додатку, здійснено практичну реалізацію і проведено дослідження результату ефективності веб-сайт, а також здійснено аналіз отриманих результатів.

Самостійні розробки і пропозицій автора:
Автором було самостійно створено Android-додаток для тайм-менеджменту на мові програмування Kotlin

Практичне значення роботи *полягає в ефективному використанні розробленого Android-додатку для підвищення продуктивності*

Недоліки: *Суттєвих недоліків в роботі не виявлено.*

Загальний висновок: *Кваліфікаційна робота Бакалавра виконана на високому рівні. Робота виконана з розумінням предмету. Матеріал викладено в чіткій та зручній формі. Вважаю, що кваліфікаційна робота відповідає вимогам до випускних робіт і заслуговує оцінки «відмінно», а її авторові, студенту Цісар А.А., може бути присвоєний кваліфікаційний рівень "бакалавр" зі спеціальності «Інженерія програмного забезпечення»*

Рецензент: *Саварин Павло Вікторович, к.п.н. доцент кафедри ЦОТ*

Рецензент кваліфікаційної роботи _____ (підпис)

«7» 06 2024 р.

Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

ВІДГУК
КЕРІВНИКА НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувач вищої освіти *Цісар Андрій Анатолійович*
(прізвище, ім'я, по батькові)
група *ІПЗ-42*

Тема кваліфікаційної роботи:
Розробка Android-додатку для тайм-менеджменту «TimeSage»

Керівник *Яцук Андрій Анатолійович*

Актуальність теми
У сучасному світі ефективне управління часом є критично важливим для підвищення продуктивності та зменшення стресу. Додатки для тайм-менеджменту допомагають користувачам структурувати свій робочий день, створювати та відстежувати завдання, а також аналізувати прогрес. Тому такі додатки стають незамінними інструментами для студентів та підприємців.

Об'єкт дослідження
Розробка Android-додатку та методи для подолання прокрастинації

Характеристика теоретичного рівня, наявності самостійних розробок і практичної значимості роботи:
Автор аналізує різні методи та засоби для розробки функціонального Android-додатку. Визначає найкращі з них і обґрунтовує своє рішення. Зміст роботи відповідає обраній темі.

Зауваження та недоліки: *Суттєвих недоліків в роботі не виявлено.*

Загальний висновок:
Кваліфікаційна робота бакалавра виконана на високому рівні. Робота виконана відповідно до вимог, логічно й послідовно описує хід виконання поставленого завдання. Пояснювальна записка відповідає стандартам до її оформлення. Робота може бути оцінена на високу оцінку

Керівник _____ **(*Яцук А.А.*)**
(підпис) (прізвище, ім'я, по батькові)

«7» червня 2024 р.

АНОТАЦІЯ

Цісар А. А. Розробка Android-додатку для тайм-менеджменту «TimeSage» Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2024.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків і пропозицій, списку використаних джерел.

У першому розділі здійснено аналіз сучасного стану проблеми прокрастинації і сформульовано завдання. В другому розділі визначено вимоги до розроблюваної системи, а також засоби, методи і алгоритми розробки. У третьому розділі розроблено мобільний додаток описані усі використані функції під'єднано додаток до бази даних. У висновках узагальнено інформацію, відображену у попередніх частинах. Результати розробки демонструють можливості розроблення програм для операційної системи Android.

Ключові слова: прокрастинація, firebase, операційна система, компонент.

ABSTRACT

Tsisar A. A. Development of Android Application for Time Management "TimeSage" Manuscript.

Bachelor's qualifying thesis of the OP "Software Engineering". Lutsk National Technical University. Lutsk, 2024.

The bachelor's qualification work consists of an introduction, three sections, conclusions and proposals, a list of used sources.

In the first chapter, an analysis of the current state of the problem of procrastination is carried out and the task is formulated. The second section defines the requirements for the developed system, as well as means, methods and development algorithms. In the third section, the mobile application is developed, all the functions used are described, the application is connected to the database. The conclusions summarize the information presented in the previous parts. The development results demonstrate the possibilities of developing applications for the Android operating system.

Keywords: procrastination, firebase, operating system, component.

ЗМІСТ

ВСТУП

Актуальність теми. Прокрастинація – це відкладення справ на потім [1]. Усі люди відкладають справи, існують багато різних чинників для цього такі як: занадто велика само впевненість, думки про те, що справи можна завершити за декілька годин, або навпаки страх невдачі, що нічого не вийде, незважаючи на ті зусилля, які ви вкладете.

Прокрастинація виступає чинником, який починає гальмувати процес самоосвіти і розвитку, заважає становленню особистості у соціальному середовищі. Також слід зазначити про людей з розладом дефіциту уваги та гіперактивності (РДУГ) – це розлад, який характеризується стійкими труднощами з концентрацією уваги, гіперактивністю та імпульсивністю. Люди з цією хворобою мають більшу схильність до відкладання своїх завдань через недостачу концентрації, їх може відволікти найменший подразник.

Для боротьби з причинами прокрастинації, в першу чергу, слід навчитися керувати власним часом, тобто необхідно навчитися планувати роботу, щоб можна було виконати максимальну кількість завдань за якнайменший термін. Але єдиного способу побороти прокрастинацію не існує, кожна людина повинна знайти свій шлях для вирішення цієї проблеми.

Хорошим способом є розбиття завдання на невеликі підпункти, які виконати простіше, також можна розбити час виконання на інтервали по 25 хвилин з невеликими перервами по 5 хвилин [2].

Мета дослідження: розробка додатку тайм-менеджеру для допомоги подолання прокрастинації.

Для досягнення поставленої мети потрібно виконати наступні завдання:

- провести аналіз існучих аналогів;
- здійснити вибір засобів для розробки мобільного додатку;
- здійснити розробку додатку тайм-менеджера;
- провести тестування додатку.

Об’єкт дослідження: розробка Android-додатку та методи для подолання ознак прокрастинації.

Предмет дослідження: додатки планування дня і помічники в концентрації.

Практична цінність розробки: допомагає ефективніше використовувати свій час.

РОЗДІЛ 1
АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ РОЗРОБКИ МОБІЛЬНИХ ДОДАТКІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Сучасний стан прокрастинації все більше починає впливати на наше життя. З кожним роком все більше людей стикаються з проблемою прокрастинації. Вона з’являється, починаючи від звичайних домашніх справ, завершуючи важливими робочими завданнями [1]. Прокрастинація викликана великою кількістю факторів які можна поділити на дві групи: внутрішні і зовнішні. До першої групи можна віднести:

– бажання виконати все бездоганно, що призводить до страху невдачі;

– низька самооцінка, через яку можуть відкладатися справи;

– через малий інтерес до завдання, або неправильне відчуття його значення призводить до відкладання;

– погані навички саморегуляції, або труднощі з керуванням часом, плануванням і організацією завдань можуть призвести до прокрастинації.

До другої категорії відносять:

– у сучасному світі отримати інформацію стало просто, через це ми почали дуже легко відволікатися;

– відсутність чітких цілей і термінів призводить до зволікань у їх виконанні;

– погано організоване, або просто не зручне робоче місце змушує постійно звертати на себе увагу.

Постійна прокрастинація може мати серйозні наслідки для роботи і життя загалом, такі як:

– зменшення продуктивності;

– постійний стрес через невиконання поставлених завдань, який може призвести до різноманітних проблем зі здоров’ям;

– проблем з комунікацією із рідними, або колегами на роботі.

Проаналізувавши доступні крамниці застосунків, альтернативи можна розділити на 2 категорії, а саме веб-застосунок і додаток, який встановлюється безпосередньо на сам пристрій. Хоча більшість сервісів мають і додаток, і веб-версію.

Google Keep – це безкоштовний сервіс для створення заміток, створений компанією Google. На рисунку 1.1 представлено головну сторінку сервісу.

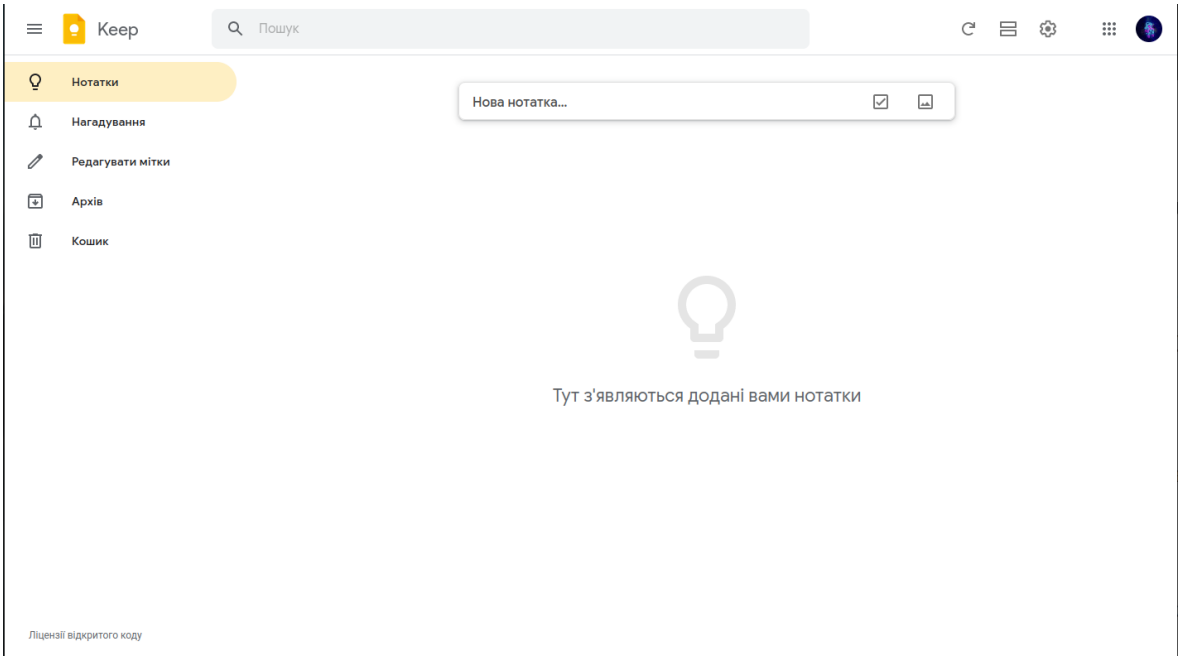

Рисунок 1.1 – Google Keep [3]

Він є насправді досить потужним і одночасно простим в користуванні сервісом планування. Він має наступні можливості:

– створення декількох видів нотаток і списки;

– створення міток для швидкого пошуку;

– Google Keep існує на різних платформах і має миттєву синхронізацію між ними;

– нотатки можна редагувати спільно з іншими людьми і ділитися ними.

Notion – це платформа для ведення заміток, планування дня або цілого тижня. Notion (рис. 1.2) існує на Windows, Android і IOS, також має веб-додаток, що дозволяє використовувати його на всіх можливих платформах.

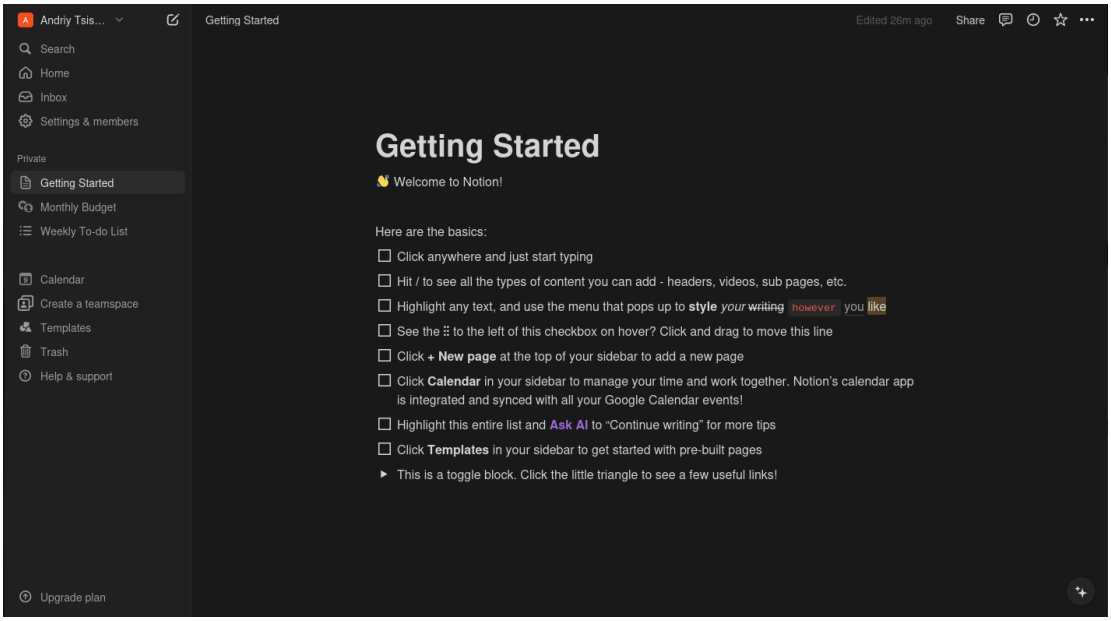

Рисунок 1.2 – Notion [4]

Платформа замінює велику кількість сервісів допомагає краще відстежувати прогрес проєкту і планувати його. Серед основних функцій слід зазначити такі:

– дозволяє створювати нотатки в різних форматах;

– створення щоденника, який дозволяє записувати свій день або тиждень з можливістю додавання фотографій;

– у сервісі існують шаблони, готові до використання;

– сервіс може синхронізуватися з Google Календар, Slack, або Trello.

Forest: Focus for Productivity – це додаток на Android, який допомагає зосередитися на завданні, використовуючи метод Pomodoro, зображення його головної сторінки представлено на рисунку 1.3. Ідея додатку в тому, що коли ви запускаєте таймер, ви садите дерево яке виросте лише після 25 хвилин, якщо зупинити таймер раніше дерево зав'яне.

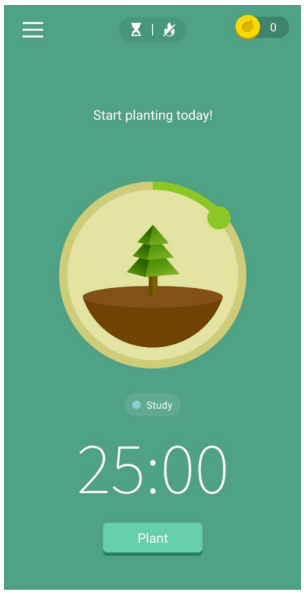

Рисунок 1.3 – Forest: Focus for Productivity [5]

Також можна дивитися кількість днів, які протягом місяця ви використали на зосередження,що в свою чергу спонукає використовувати додаток все більше і не зупиняти таймер завчасно.

Microsoft To Do: Lists & Tasks [6] – це планувальник дня, створений компанією Microsoft, який доступний як на Android, Windows так і у вигляді веб-застосунку. Має простий дизайн, який зображено на рисунку 1.4.

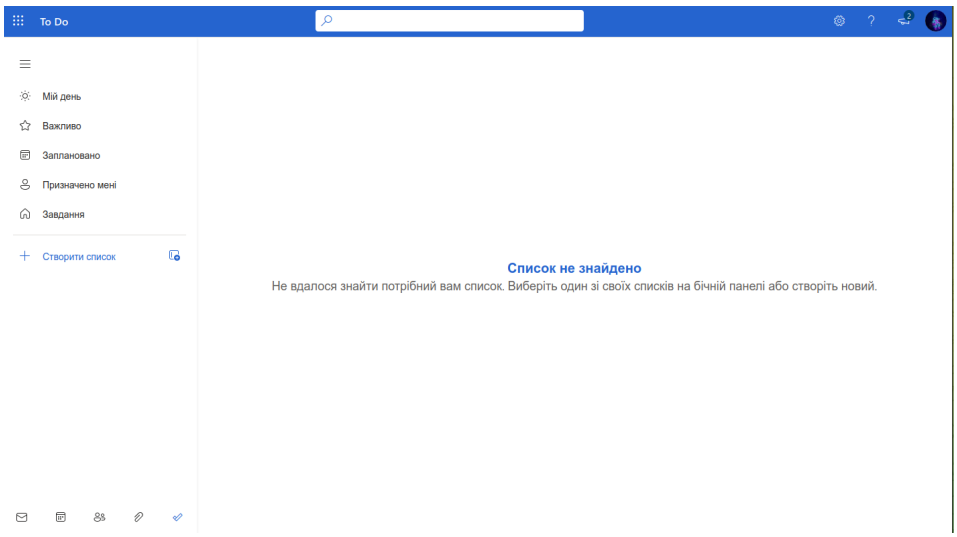

Рисунок 1.4 – Microsoft To Do: Lists & Tasks

За допомогою якого зразу стає зрозуміло, що додаток спеціалізується на To-do листах. Серед основних функцій слід зазначити:

– дозволяє створювати замітки і додавати примітки до них;
– сервіс може легко синхронізуватися між своїми пристроями;
– має можливість створювати списки спільно з іншими людьми;
– дозволяє створювати мітки для швидкого пошуку списків.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Після проведеного аналізу стало зрозуміло, що для кращої організації дня необхідно складати список справ, це допомагає краще концентруватися на завданні, тому що не має необхідності в запам’ятовуванні всього списку завдань, які були поставленні на день. Також важливою частиною є відрахунок залученого часу. Для цього необхідно розробити таймер зворотного відліку для поставлення жорстких дедлайнів, щоб мати можливість отримати кращі результати. Отже, завдання на кваліфікаційну роботу бакалавра наступне:

– провести аналіз існучих аналогів;
– здійснити вибір засобів для розробки мобільного додатку;
– здійснити розробку з реалізацією функцій реєстрації і аутентифікації користувачів; створенню, редагуванню і видаленню заміток; переходами між екранами; створити таймер з можливістю паузи;
– провести тестування додатку.

Висновки до розділу 1

Мною було здійснено аналіз чинників і можливих способів для боротьби з прокрастинацією. Також було здійснено аналіз доступних альтернативних сервісів і додатків. Основною проблемою яких було перевантаження функціями і досить велика плата за користування деякими з них. Хоч усі сервіси дозволяють використання і без внесення плати, але відбувається обмеження

функціоналу, а в деяких випадках через обмеження взагалі пропадає суть для їхнього користування.

В ході мого дослідження було виявлено, що для покращення продуктивності достатньо створювати список лише на один день. Також відчутне збільшення продуктивності відбувалося при використанні таймеру з методом pomodoro [2], суть якого полягає у виділені 25 хвилин на одне завдання з перервами між ними на 5 хвилин.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

Розроблювана мною система складається з Android додатку, написаного на мові програмування Kotlin. Відштовхуючись від мови програмування, було прийнято рішення використовувати редактор коду Android Studio з усіма встановленими доповненнями для розробки програм під операційну систему Android. У додатку дані користувача про реєстрацію і нотатки будуть зберігаються у базі даних, у ролі якої використовується потужна хмарна база даних під назвою Firebase, розроблена компанією «Firebase Inc».

Для ефективної розробки було прийнято рішення створити uml-діаграму прецедентів. На рисунку 2.1 зображена UseCase діаграма взаємодії користувача із системою і її можливостями. На діаграмі показано, що користувач може зареєструватися у системі. Після чого він отримує можливість аутентифікуватися.

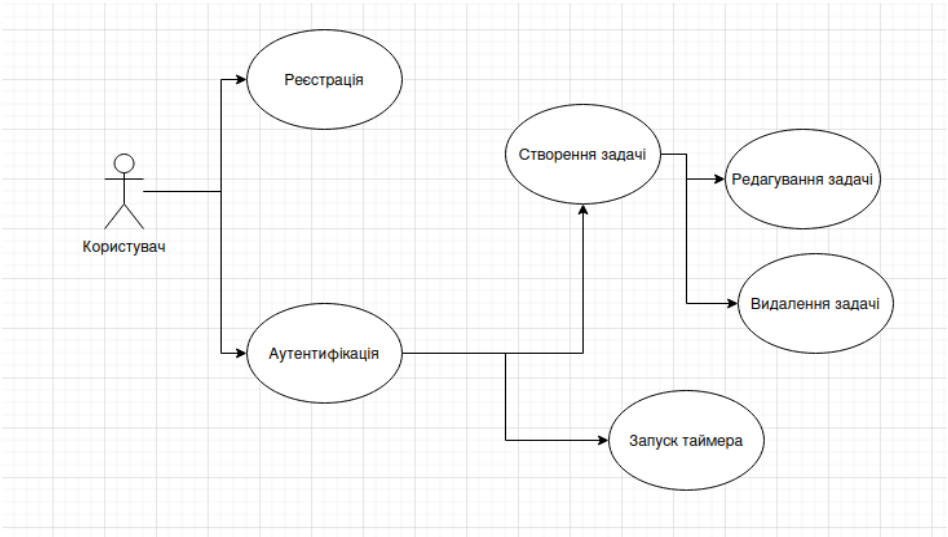

Рисунок 2.1 – UseCase діаграма взаємодії користувача з системою

Аутентифікація у свою чергу дає можливість користувачу створювати завдання, які він може редагувати або видаляти. Крім того, він отримує отримує можливість запускати таймер. Також було розроблено діаграму класів, яка показує структуру системи і взаємодію між різними її компонентами (рис. 2.2).

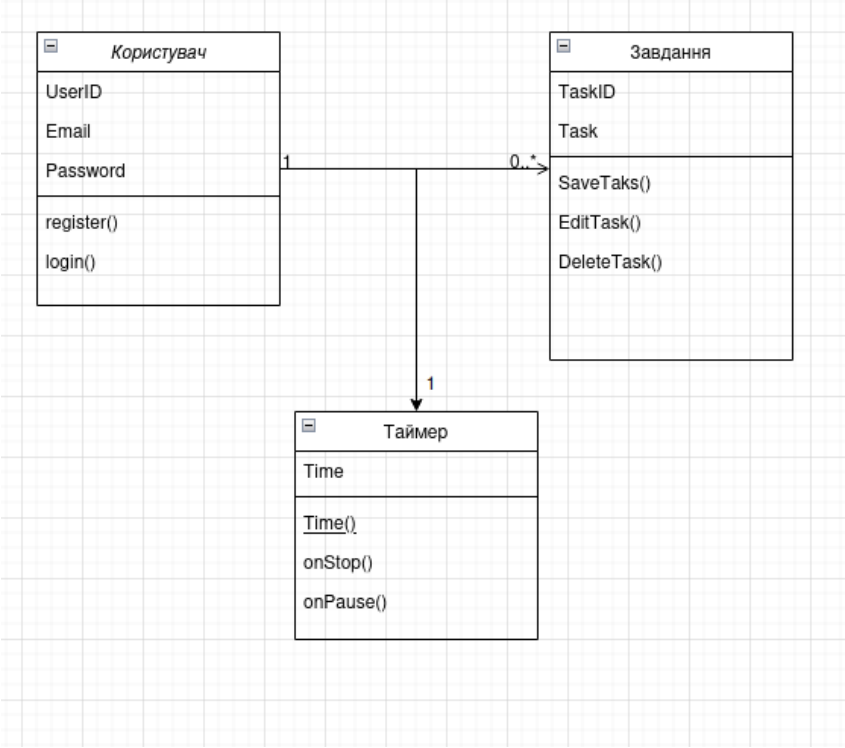

Рисунок 2.2 – Діаграма класів

За допомогою діаграми класів стає зрозуміло, що створювана система складається з трьох основних класів, а саме: користувач, таймер і завдання, клас користувач може створювати декілька завдань, а клас таймер можна створити лише один.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Android Studio – це інтегроване середовище розробки або скорочено IDE, створене компанією Google (рис. 2.3) [7].

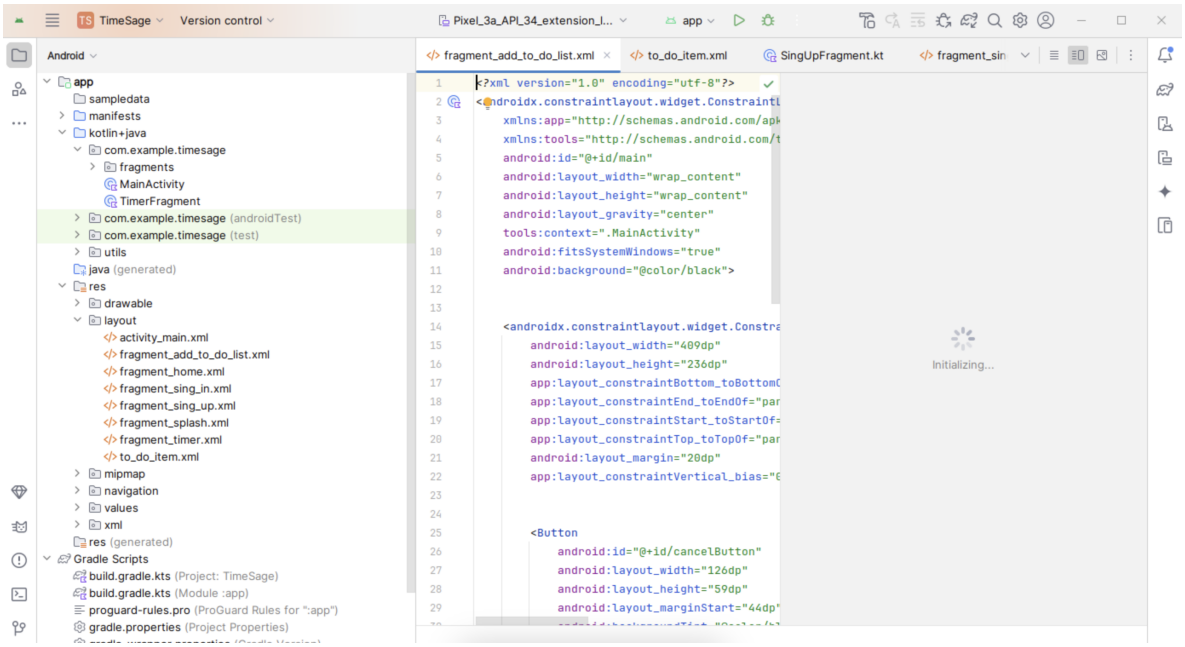

Рисунок 2.3 – Android Studio

Воно є відкритим і безкоштовним. Має простий і приємний інтерфейс, підтримку основних мов програмування таких як Java і Kotlin. Які популярні у першу чергу для створення мобільних додатків.

Відтворення коду може відбуватися через встановлений емулятор або й відразу на мобільному пристрої за допомогою кабелю USB. Також, починаючи з версії Android 11, є можливість використовувати і WIFI для перевірки правильності написання програми, відтворюючи її відразу на мобільному пристрої. Однією з корисних функцій є можливість одразу опублікувати свій додаток в Play Marker. Важливими функціями є наступні:

– можливість створення програми за допомогою макету, який спрощує і пришвидшує його створення у декілька разів особливо для початківців у сфері розробки програмного забезпечення для мобільних пристроїв;

– Android Studio є безкоштовним, що робить його досить привабливим рішенням для розробки;

– у середовища є багато різних бібліотек, доступних для використання;

– у середовища є велика спільнота розробників.

Firebase – це потужна і проста платформа для створення мобільних або веб-додатків (рис. 2.4) [8].

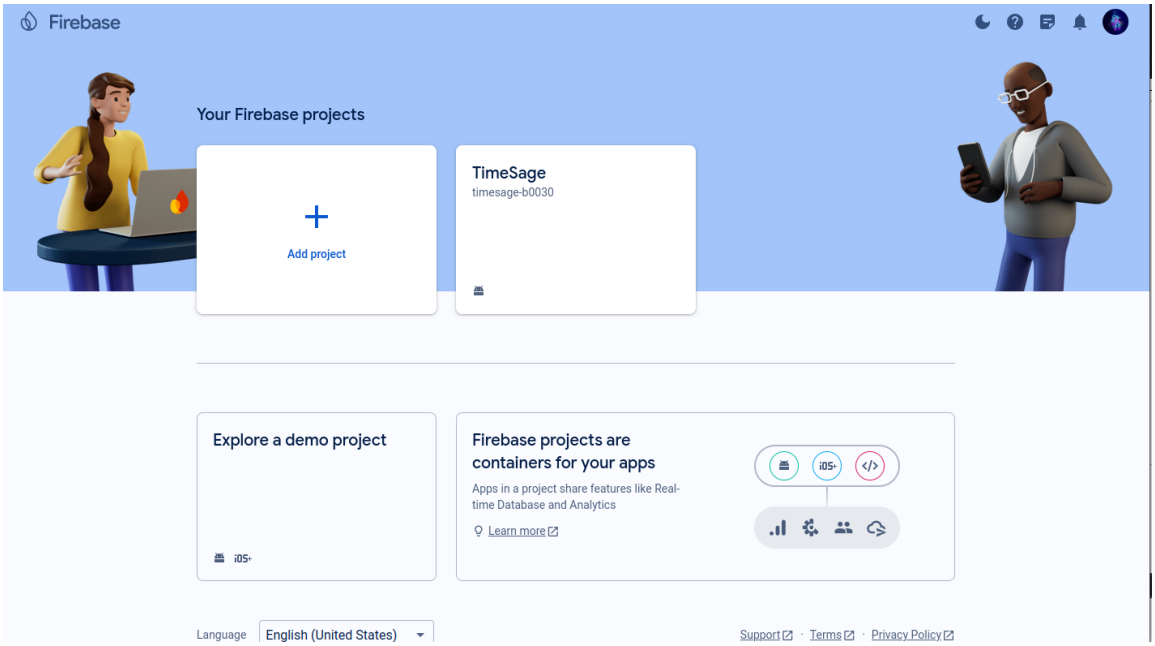

Рисунок 2.4 – Головна сторінка сервісу Firebase

У платформи є усе, що потрібно для успішного створення і підтримування проєктів, а саме:

– хмарне сховище дозволяє безпечно зберігати і керувати користувачеві своїми створеними файлами;

– дозволяє створити безпечну перевірку користувачів за допомогою сервісу Firebase Authentication;

– за допомогою сервісу Firebase Cloud Messaging (FCM) дозволяє створювати повідомлення, які побачить кожен ваш користувач;

– за допомогою сервісу Firebase Analytics дозволяє отримати загальну статистику для контролю за вашим додатком.

Важливим є те, що сервіс майже безкоштовний. Тому, на мою думку, він є найкращим рішенням для створення додатку.

Kotlin – це сучасна об’єктно-орієнтована мова програмування, розроблена компанією JetBrains (рис. 2.5)[9, 10].

Kotlin є мультиплатформенною мовою програмування. Це означає, що його можна використати для написання коду під різні платформи, такі як:

– Android;

– IOS;

– веб–сайти;
– сервери;
– desktop: Windows, MacOS, Linux.

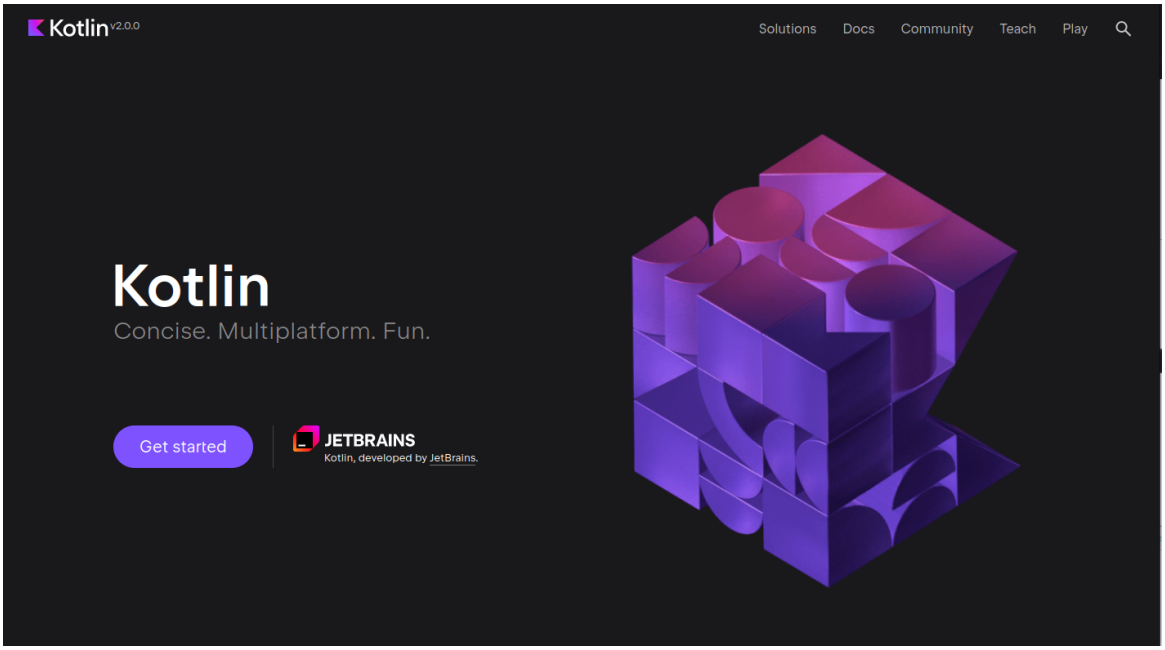

Рисунок 2.5 – Офіційна сторінка мови програмування Kotlin

Однією з переваг Kotlin є його спільнота, яка постійно збільшується. Це зумовлено наступними чинниками:

– Kotlin може бути інтегрований в код на мові програмування Java, це зумовлено тим що код на Kotlin компілюється в байт-код, який в свою чергу, виконується на віртуальній машині Java;

– код на Kotlin має кращий захист порівняно з кодом написаним на інших мовах програмування;

– компанія Google допомагає в розробці і підтримці мови програмування;

– мова програмування допомагає писати код, який за швидкістю не поступається кодові написаним на Java;

– Kotlin сприяє написанню «чистого» коду, завдяки своєму синтаксису це робить його підтримку більш простою для розробників [11].

Саме ці чинники дозволяють Kotlin бути однією з найпопулярніших мов програмування для платформи Android.

Висновки до розділу 2

У другому розділі був проведений аналіз методів розробки для наступного його використання під час створення мобільного додатку для операційної системи Android. Середовищем розробки був обраний Android Studio як найкраще середовище для мобільної розробки.

Мовою програмування було вирішено вибрати Kotlin за його популярність і простоту написання додатків на цій мові програмування. В якості бази даних вирішено вибрати сервіс Firebase за його можливість масштабування і простоту розробки на ньому.

РОЗДІЛ 3
ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

3.1 Практична реалізація об’єкта проектування

З самого початку відкриємо Android Studio і створимо проєкт, вибравши «Empty Views Activity», після натискаємо кнопку, далі в полі Name вводимо назву нашої програми і вибираємо мінімальний SDK, в моєму випадку це Android 7.0 (рис. 3.1) і мову програмування Kotlin.

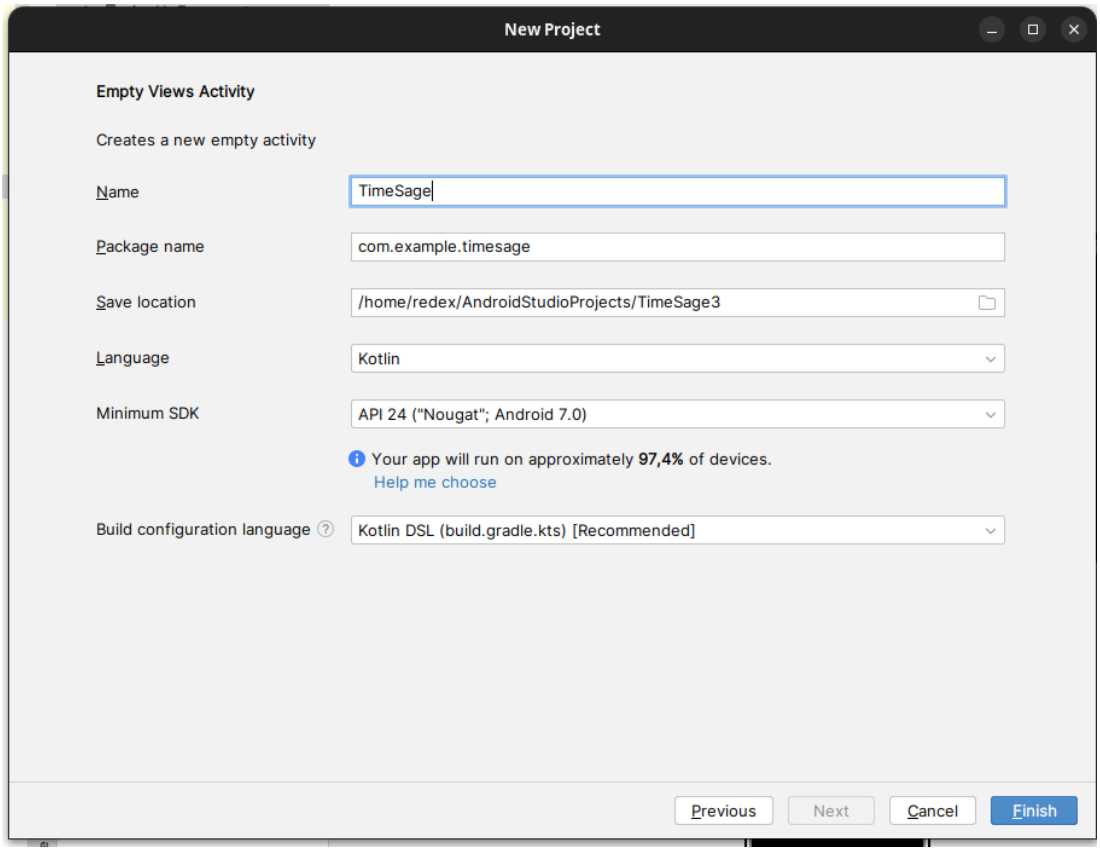

Рисунок 3.1 – Вікно створення нового проєкту

Після завершення створення проєкту будуть відкриті два файли: «MainActivity.kt» і «activity_main.xml». Перший це файл, в якому ми будемо писати код, а другий це файл з дизайном вікна. Перш за все був реалізований повноекранний режим, написавши наступний код, який наведений у лістингу 3.1.

Лістинг 3.1 – Функція увімкнення повноекранного режиму

```
 // Встановлюємо відображення вікна у повноекранному режимі
 window.setFlags(
    WindowManager.LayoutParams.FLAG_FULLSCREEN, // Увімкнення
повноекранного режиму
    WindowManager.LayoutParams.FLAG_FULLSCREEN  // Застосування
повноекранного режиму
     )
```

Кінець лістингу 3.1

Тепер для переміщення між вікнами будемо використовувати navigation component. Для цього перейдемо в директорію «Gradle Scripts» і відкриємо файл «build.gradle.kts» (рис. 3.2), з двох доступних файлів виберемо той, в якому записуються модулі і впишемо необхідні залежності, після чого натиснемо на кнопку «Sync Now».

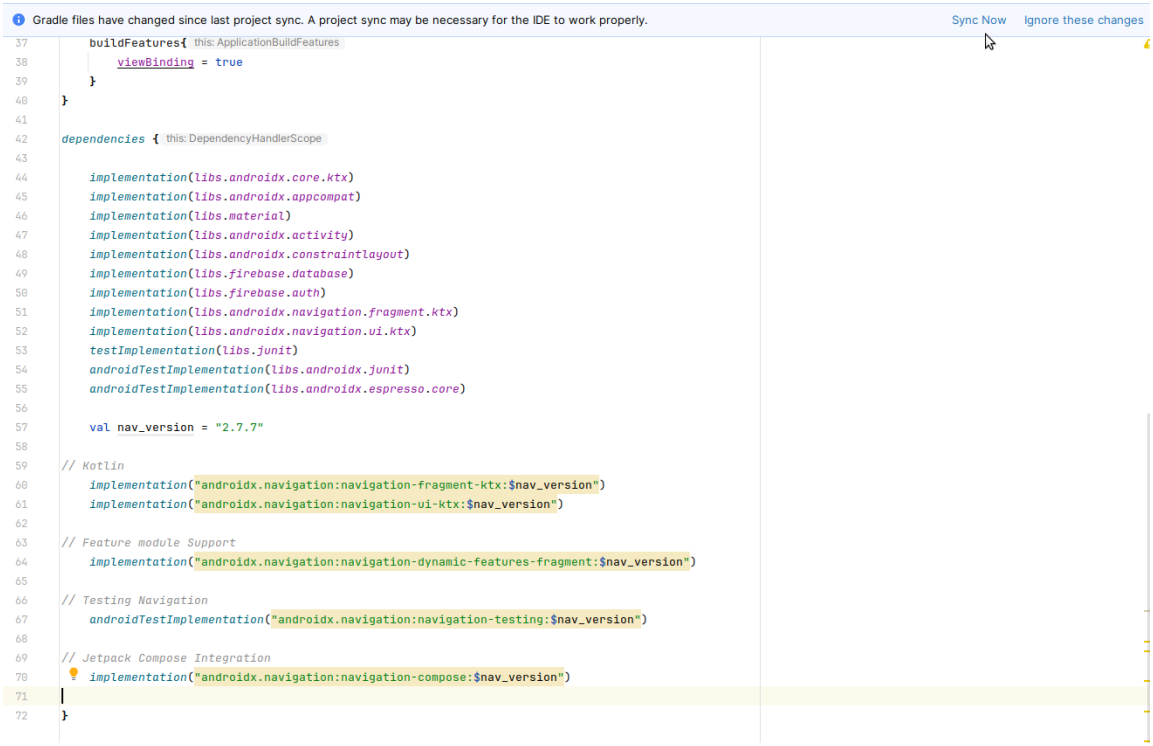

Рисунок 3.2 – Файл «build.gradle»

Наступним кроком буде створення нової директорії. Для цього наводимо курсор на папку res і натискаємо на праву кнопку, із списку вибираємо «New» і «Android Resourse Directory», називаємо його Navigation. Після цього у

створеній папці додаємо «Navigation Resource File» назву можна обрати довільну я напишу nav_graph. Тепер двічі натискаємо на створений нами файл.

Після чого натискаємо кнопку «New Destination» і «Create New Destination» і створимо нові вікна, а саме: екран заставки, аутентифікації, реєстрації, екран зі списком, екран з таймером, створимо між ними зв'язки, просто розмістивши їх у вигляді стрілок між компонентами (рис. 3.3).

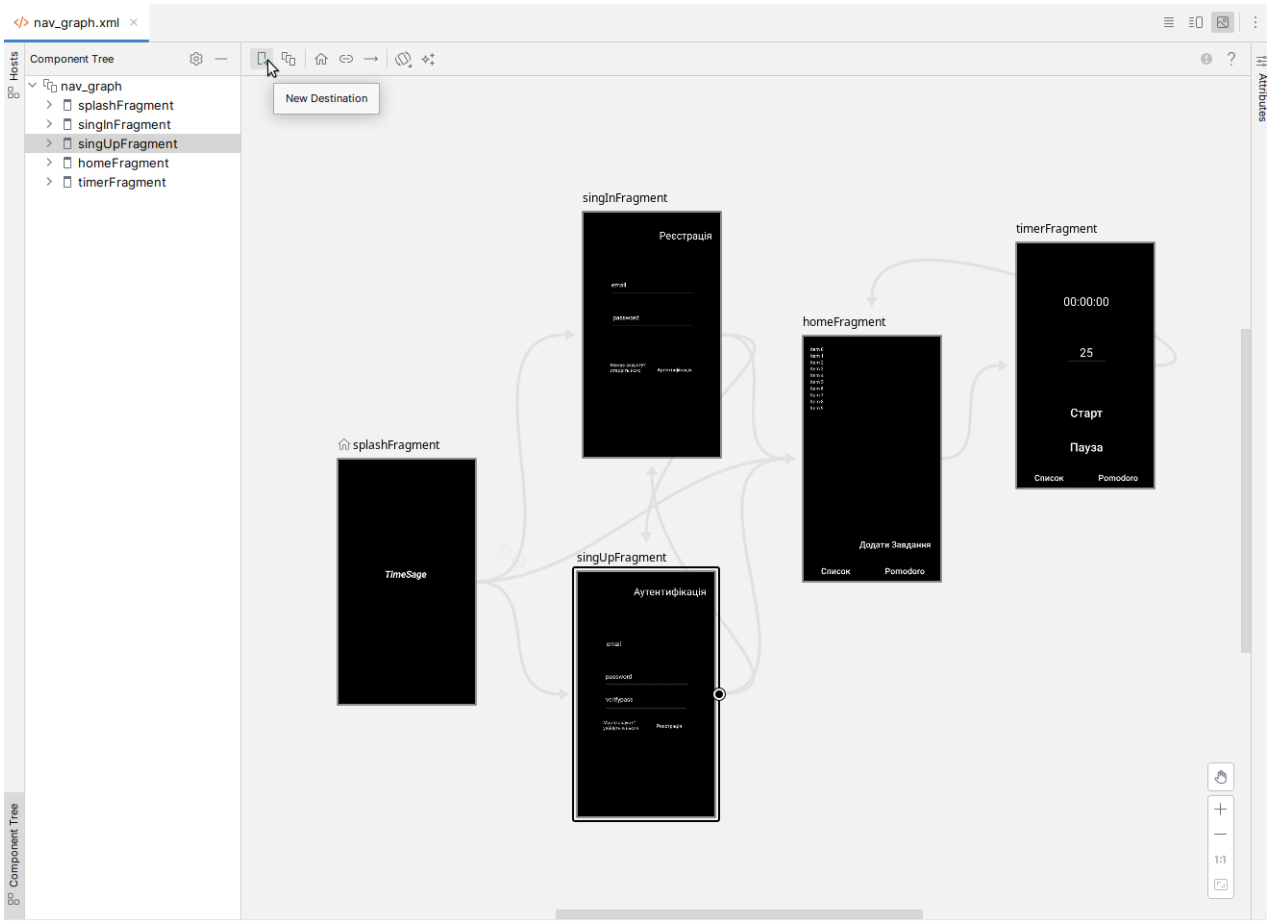

Рисунок 3.3 – Навігаційний компонент

Для кращого розуміння давайте розглянемо наступні принципи його роботи [7]:

– graph компонент для визначення маршруту для переходу між компонентами;

– navigation host контролює переходи між екранами на основі попереднього компоненту;

– navigation controller дозволяє програмно запускати фрагменти і отримувати інформацію про поточну сторінку;

– navArgs передає дані між фрагментами.

Наступним кроком буде під'єднання і налаштування Firebase. Для цього в Android Studio виберімо вкладку «Tools», у випадаючому меню натискаємо на пункт Firebase (рис. 3.4).

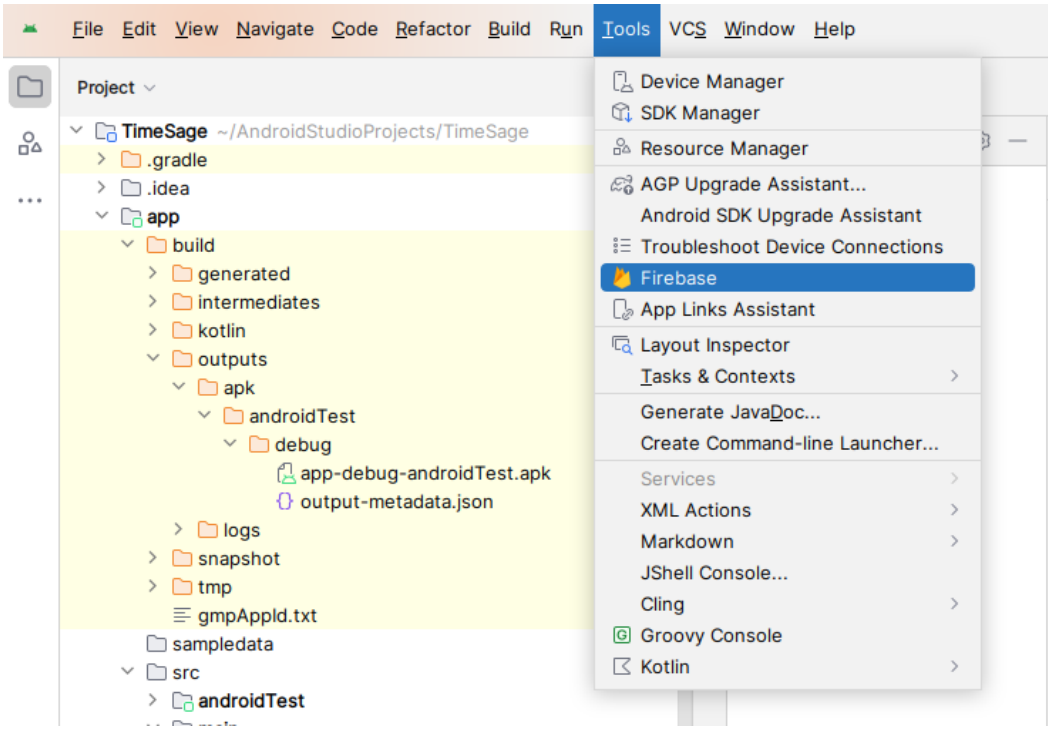

Рисунок 3.4 – Розташування пункту Firebase

Праворуч відкривається меню, в ньому шукаємо вкладку Authentication, після чого натискаємо на кнопку connect to firebase, потім відкриється браузер, в якому реєструємося у системі і натискаємо add project, даємо ім’я нашому проєкту і вибираємо геолокацію нашого сервера, після чого натискаємо завершити.

Наступним кроком потрібно перейти у браузер на офіційний сайт Firebase і увімкнути Authentication, натиснувши кнопку «Get Started», після чого вибрати спосіб входу з запропонованих. Створимо так само базу даних, вибравши найближче зі запропонованих до вас розташування.

На самому початку була змінена стандартна іконка програми, для цього наводимо мишку на папку «res» права кнопка миші пункт «New», далі вибираємо «Image Asset» (рис. 3.5).

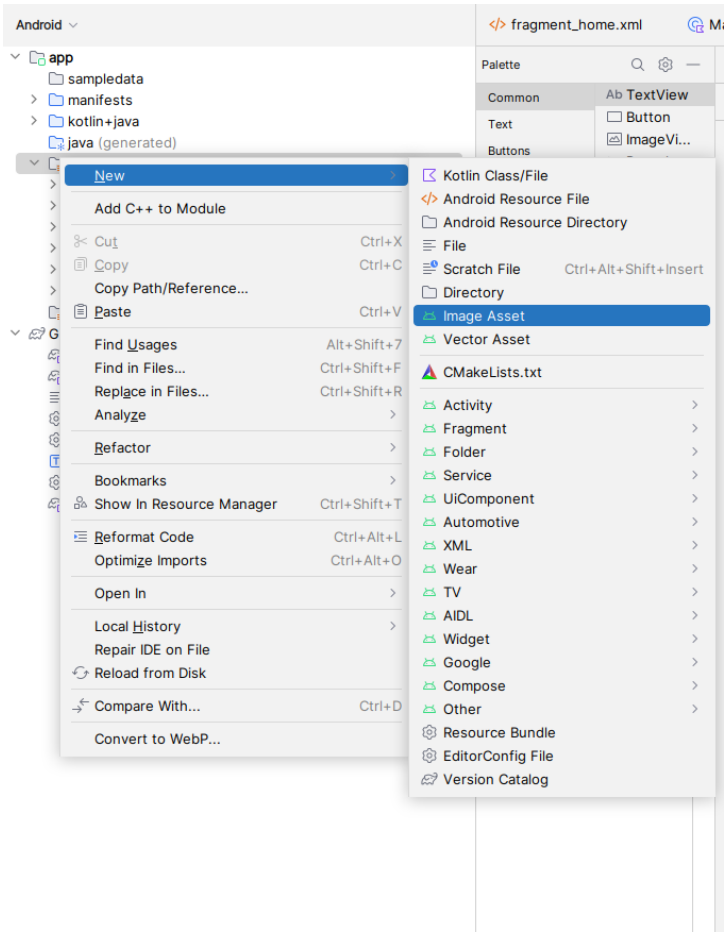

Рисунок 3.5 – Розташування пункту «Image Asset»

У випадаючому вікні у пункті «path» вибираємо розташування нашої іконки і натискаємо далі. Іконка додатку змінилася на задану.

Далі було реалізовано екран- заставку, який складається з простого напису з назвою програми. У файл з кодом імпортуємо компонент навігації і компонент аутентифікації. Створимо обробник, який перевірить чи користувач аутентифікований, після 2 секунд паузи його перенаправить на головну сторінку або на сторінку реєстрації, код обробника представлений у лістингу 3.2

Лістинг 3.2 – Функція перевірки і переадресації з екрану заставки

```
// Отримуємо FirebaseAuth для автентифікації користувачів
auth = FirebaseAuth.getInstance()
// Отримуємо контролер навігації для управління переходами між
фрагментами
navController = Navigation.findNavController(view)
// Створюємо елемент Handler для затримки виконання
Handler(Looper.myLooper()!!).postDelayed(Runnable {
    // Перевіряємо, чи користувач автентифікований
    if (auth.currentUser != null) {
        // Якщо користувач автентифікований, переходимо до
домашнього фрагменту

navController.navigate(R.id.action_splashFragment_to_homeFragment)
    } else {
        // Якщо користувач не автентифікований, переходимо до
фрагменту реєстрації

navController.navigate(R.id.action_splashFragment_to_singUpFragmen
t)
    }
}, 2000) // Затримка виконання на 2 секунди
```

Кінець лістингу 3.2

Наступним кроком є створення сторінки реєстрації і аутентифікації Для цього створимо три поля введення тексту, а саме: рядок для введення email адресу і два рядка для введення паролю і його перевірки.

Для цього було створено метод в який відбувається передача даних які ввів користувач попередньо забравши пробіли. Наступним кроком відбувається перевірка, щоб реєстрація спрацьовувала лише при умові того, що користувач ввів усі необхідні дані. При успішному спрацюванні за допомогою методу «Toast» виводиться повідомлення користувачу що реєстрація є успішною. Приклад реалізації коду наведено у лістингу 3.3.

Лістинг 3.3 – Функція перевірки і реєстрації користувача

```
// Перевіримо, чи всі поля заповнені
if (email.isNotEmpty() && pass.isNotEmpty() &&
verifypass.isNotEmpty()) {
    // Перевіримо, чи паролі співпадають
```

```
    if (pass == verifypass) {
        // Створюємо користувача
        auth.createUserWithEmailAndPassword(email, pass)
            .addOnCompleteListener(OnCompleteListener { task ->
                // Перевіряємо, чи реєстрація була успішною
                if (task.isSuccessful) {
// Відображаємо повідомлення про успішну реєстрацію
       Toast.makeText(context, "Реєстрація успішна",
Toast.LENGTH_SHORT).show()
// Переходимо до домашнього фрагменту

navControl.navigate(R.id.action_singUpFragment_to_homeFragment)
                } else
          {
// Відображаємо повідомлення про помилку, якщо реєстрація не
вдалася
    Toast.makeText(context, task.exception?.message,
Toast.LENGTH_SHORT).show()
        }
           })
  }
}
```

Кінець лістингу 3.3

Було добавлено також текст, при натисканні на який буде відбуватися перехід на сторінку аутентифікації для користувачів, які вже мають аккаунт. Для цього створимо метод в якому при натисканні відбуватиметься перехід на сторінку аутентифікації. Приклад коду наведений у лістингу 3.4.

Лістинг 3.4 – Функція переходу на іншу сторінку при натисканні на текст

```
// Встановлюємо обробник для натискання на TextView
binding.authTextView.setOnClickListener {
    // Переходимо до фрагменту авторизації

navControl.navigate(R.id.action_singUpFragment_to_singInFragment)
}
```

Кінець лістингу 3.4

Подібним способом створимо сторінку аутентифікації, змінивши метод createUserWithEmailAndPassword на signInWithEmailAndPassword і забравши

перевірку правильності введення паролю. Створимо таймер, для його роботи необхідно добавити ще одну залежність в файл Gradle (лістинг 3.5).

Лістинг 3.5 – Залежність необхідна для роботи таймера

```
// Увімкнення функції ViewBinding у проєкті
buildFeatures {
    viewBinding = true
}
```

Кінець лістингу 3.5

На головній сторінці було додано кнопку, яка перенаправить нас на вікно з таймером. Створимо нову функцію і назвемо її countDownTimer, в ньому створимо об’єкт таймера, якому буде передано час на скільки його запускати, і кількість оновлень в мілісекундах. По завершенню відліку замінимо циферблат на слово «завершено». Приклад реалізації наведено у лістингу 3.6.

Переведемо відлік з мілісекунд на звичний для нас формат часу години, хвилини, секунди, використавши прості математичні обчислення.

Лістинг 3.6 – Функція роботи таймера

```
// Функція для запуску таймеру
private fun countDownTimer(time: Long) {
    // Скасовання таймер, якщо він запущений
    timer?.cancel()
    // Створення нового таймеру
    timer = object : CountDownTimer(time, 1000) {
        //функція яка викликається на кожен тік таймера
        override fun onTick(millisUntilFinished: Long) {
            // Вираховування години
            val hour = (millisUntilFinished / 1000) / 3600
            // Вираховування секунди
            val sec = (millisUntilFinished / 1000) % 60
            // Вираховування хвилини
            val min = ((millisUntilFinished / 1000) % 3600) / 60
            // збереження часу який залишився
            timeleft = millisUntilFinished
            // Форматування час у форматі HH:MM:SS
            val timerpomodoro = "%02d:%02d:%02d".format(hour, min,
sec)
            // Оновлення циферблату таймера
            binding.pomodoroTimer.text = timerpomodoro
        }
        override fun onFinish() {
```

```
            // Оновлення тексту таймера на "Завершено"
            binding.pomodoroTimer.text = "Завершено"
        }
    }.start() // Запуск таймера
}
```

Кінець лістингу 3.6

Далі було добавлено кнопку «старт», код функції представлено у лістингу 3.7, для цього в конструкторі макету перенесемо відповідний елемент на наш екран. Реалізуємо, щоб при натисканні на кнопку, числові дані з текстового поля, які користувач буде вводити самостійно або залишати за замовчуванням, а саме 25 хвилин перетворювати в мілісекунди і надсилатися у функцію таймера. Також кнопка «старт» при запуску таймера зникає і вона змінюється на кнопку «стоп» за допомогою модифікатора visibility.

Лістинг 3.7 – Функція роботи таймера

```
binding.btnstart.setOnClickListener {
          val time = binding.timePomodoro.text.
toString().toLong()
            val timemin = time*60*1000
            countDownTimer(timemin)
            binding.btnstop.visibility = View.VISIBLE
            binding.btnstart.visibility = View.GONE
}
```

Кінець лістингу 3.7

Подібним чином створюємо кнопки «стоп», «паузи» і «продовжити». Тепер створимо сам список. Для цього створимо новий клас, в якому будемо під’єднуватися до сервера з базою даних, отримувати ідентифікатор користувача, в ньому створити ідентифікатор завдання, в якому вже будемо записувати сам текст завдання.

При натисканні на кнопку збереження (лістинг 3.8) будемо перевіряти, чи користувач ввів текст своєї замітки, в іншому випадку за допомогою методу «Toast» повідомимо користувача, що він не ввів ніякого тексту і зачекаємо його введення.

Лістинг 3.8 – Функція перевірки чи ввів користувач текст

```
    binding.addButton.setOnClickListener{
            val todoTask = binding.todotext.text.toString()
            if(todoTask.isNotEmpty()){
                listener.onSaveTask(todoTask, binding.todotext)
            }else{
 Toast.makeText(context, "Будь ласка додайне текст",
Toast.LENGTH_SHORT).show()
            }
}
```

Кінець лістингу 3.8

Для реалізації кнопок видалення і редагування завдань створимо новий інтерфейс (лістинг 3.9), в якому будуть функції видалення і редагування тексту, які отримують дані з нашої бази даних, викликати які будемо натисканням на відповідні кнопки на екрані.

Лістинг 3.9 – Інтерфейси видалення і редагування тексту

```
// Інтерфейс для обробки кліків в адаптері
interface ToDoAdapterClick {
    // Метод, який викликається при натисканні на кнопку видалення
завдання
    fun onDeleteTaskClick(ToDoData: ToDoData)
        // Метод, який викликається при натисканні на кнопку
редагування завдання
    fun onEditTaskClick(ToDoData: ToDoData)
}
```

Кінець лістингу 3.9

Для редагування замітки буде використовуватися те саме вікно, що і для її створення. Для цього при натисканні на кнопку будемо отримувати індифікатор тексту і виводити його в текстове поле (лістинг 3.10), також трохи змінимо код створення самої замітки, додавши перевірку при якій, якщо текстове поле при виклику елементу пусте відбувалося створення, а якщо в ньому знаходився якийсь текст, відбувалося редагування.

Лістинг 3.10 – Функція редагування замітки

```
// Метод, який викликається при натисканні на кнопку редагування
override fun onEditTaskClick(ToDoData: ToDoData) {
    // Якщо фрагмент редагування існує видаляємо його
    if (todopop != null)
        childFragmentManager.beginTransaction()
                .remove(todopop!!).commit()
    // Створення нового фрагменту для редагування з новими даними
       todopop  =  addToDoListFragment.newInstance(ToDoData.taskId,
ToDoData.task)
    // Встановлюємо слухача для фрагменту
    todopop!!.setListener(this)
    // Показуємо фрагмент редагування завдання
    todopop!!.show(childFragmentManager, addToDoListFragment.TAG)
}
```

Кінець лістингу 3.10

3.2 Розробка інсталяційного пакета

Для створення інсталяційного пакета було використано середовище розробки Android Studio, тому шо в ньому вже є такий функціонал. Вибираємо вкладку «Build» у меню що з’явилося вибираємо пункт «Build apk(s)» (рис. 3.6).

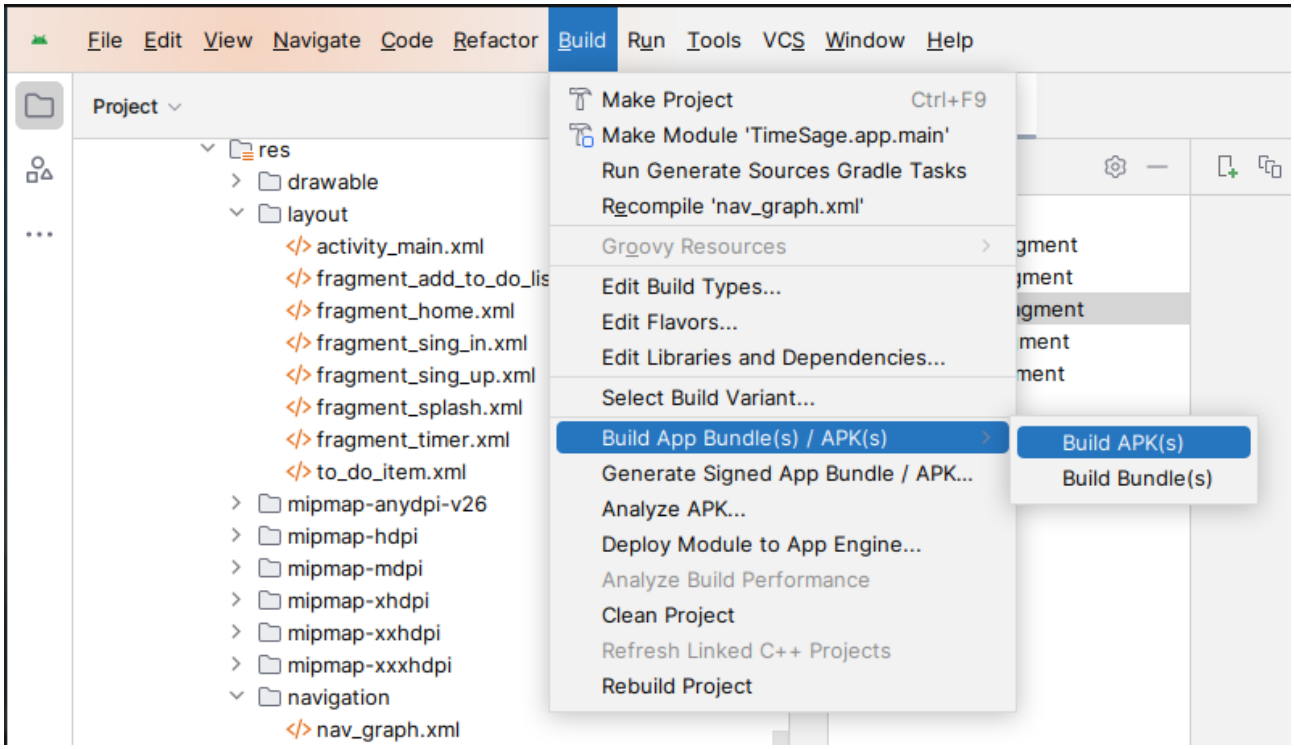

Рисунок 3.6 – Створення apk файлу

Після успішного створення в структурі проєкту з’являється нова директорія (рис. 3.7), в якій зберігається готовий apk-файл.

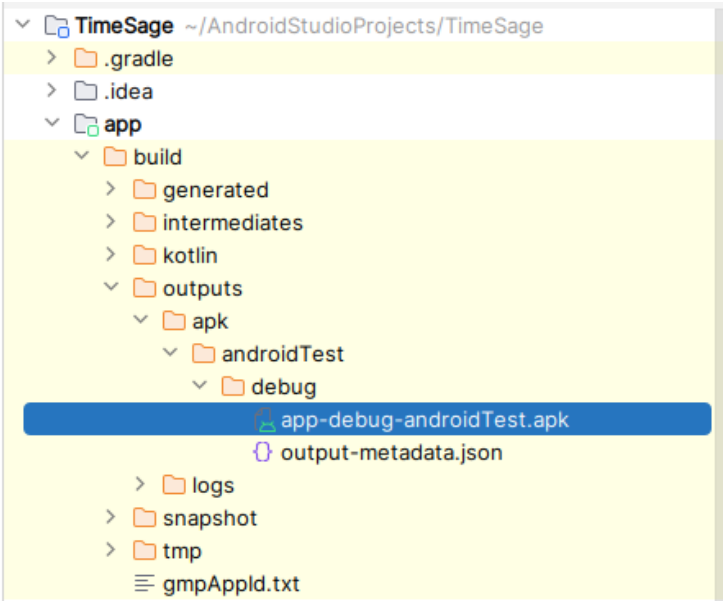

Рисунок 3.7 – Розміщення apk-файлу

3.3 Розробка документації для програмного продукту

Розробка документації є важливим кроком в створенні будь-якого проєкту, вона спрощує взаємодію користувача з технічним продуктом. Документація – це не лише інструкція, як користуватися додатком, вона слугує описом того, як додаток повинен працювати, описувати використані технології і необхідні мінімальні характеристики для своєї роботи. Для роботи додатку необхідно:

– операційна система: не нижче версії Android 7.0;

– оперативна пам’ять: для комфортної роботи необхідно не менше як 2 гігабайти оперативної пам’яті;

– місце на пристрої: для інсталяції програми необхідно 40 мегабайт вільного простору на пристрої.

Використання додатку є інтуїтивно зрозумілим, з самого початку з’являється екран заставки зображений на рисунку 3.8.

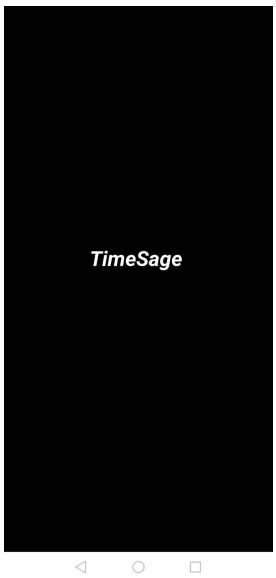

Рисунок 3.8 – Зображення сторінки заставки

Потім, якщо вхід відбувається вперше, відкривається екран, де можна ввести свій емейл і пароль для реєстрації. Приклад вікна зображений на рисунку 3.9.

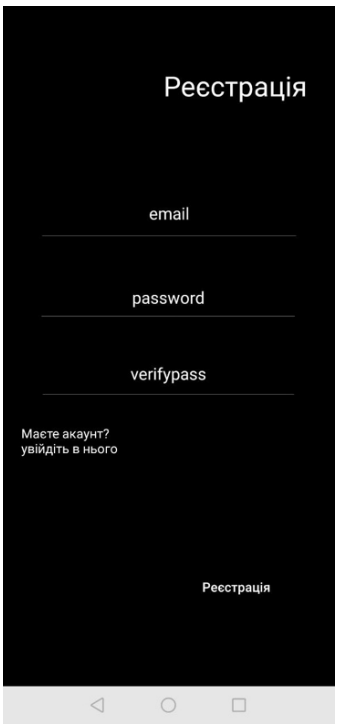

Рисунок 3.9 – Зображення сторінки реєстрації

Якщо користувач має аккаунт, то він може натиснути на текст «Маєте аккаунт? Увійдіть в нього», тоді відкриється вікно з двома полями для введення і кнопкою аутентифікації. Вигляд сторінки зображений на рисунку 3.10.

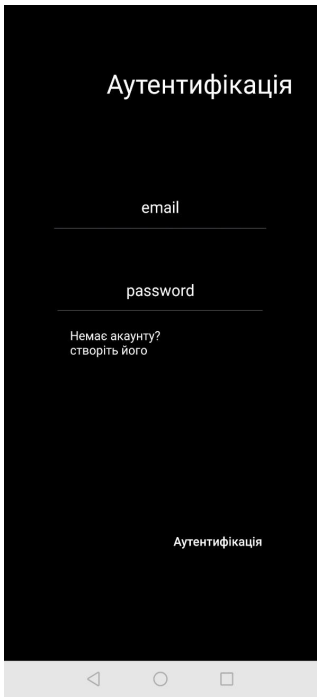

Рисунок 3.10 – Зображення сторінки аутентифікації

Після реєстрації або аутентифікації відкривається головне вікно додатку, на якому можна створити замітку, натиснувши відповідну кнопку, або видалити її, натиснувши на іконку корзини, також є можливість відредагувати замітку, якщо була допущена помилка при її написанні. Також знизу знаходяться 2 кнопки. Кнопка «списки» на головній сторінці є неактивною і кнопка «таймер» яка перенаправляє на відповідну сторінку.

Зображення головної сторінки наведено на рисунку 3.11. Після натискання на таймер відбувається перенаправлення. У вікні є одне поле, для введення числа за замовчуванням стоїть значення 25, і є кнопка «старт», яка запускає таймер.

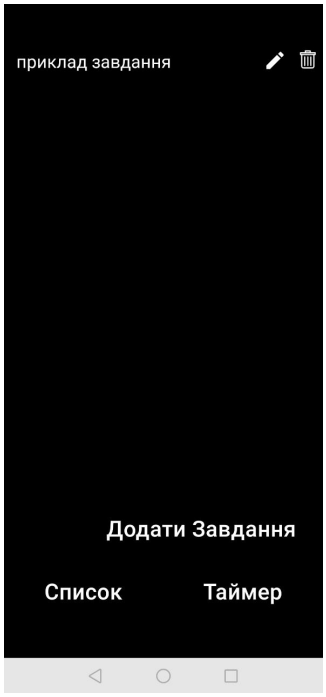

Рисунок 3.11 – Зображення головної сторінки

При запуску таймера кнопка «старт» замінюється на кнопку «стоп» і також з’являється кнопка «пауза», при натисканні на яку таймер призупиняється, і появляється кнопка продовжити. Вигляд таймеру зображений на рисунку 3.12.

Рисунок 3.12 – Зображення сторінки таймера

3.4 Тестування мобільного додатку

Було проведено перевірку основних функцій додатку, а саме: функції реєстрації, аутентифікації, створення, видалення і редагування заміток, роботу таймера.

Реєстрація нового користувача.

Було проведено перевірку можливості реєстрації за наступними кроками:

– перше відкриття додатку;
– введення електронної адреси, пароль;
– натискання кнопки «реєстрація».

В результаті користувач був успішно зареєстрований і перенаправлений на головну сторінку.

Авторизація існуючого користувача.

Було проведено перевірку можливості входу користувача в попередньо створений обліковий запис за допомогою наступних кроків:

– відкриття додатку;
– перехід у розділ аутентифікації;
– введення правильних даних облікового запису;
– натиснення кнопки «аутентифікація».

В результаті користувач успішно авторизувався і був перенаправлений на головну сторінку.

Робота з замітками.

Було проведено перевірку можливості створення редагування і видалення заміток користувачем за наступними кроками:

– натискання на кнопку «додати замітку»;
– введення тексту замітки;
– натискання на кнопку «створення замітки»;
– натискання на іконку «редагування замітки»;
– редагування тексту замітки;
– натискання кнопки «збереження»;

– натискання на іконку «видалення замітки».

Як результат було успішно створено замітку, проведено її редагування і успішне видалення із списку заміток.

Робота з таймером.

Було перевірено можливість роботи з таймером, його зупинку і функцію паузи за наступними кроками:

– перехід на вікно таймера;
– введення часу роботи;
– натискання на кнопку «старт»;
– натискання на кнопку «паузи»;
– натискання на кнопку «продовження»;
– натискання на кнопку «стоп».

Було успішно задано час роботи таймера, його запуск, перевірено можливість паузи і зупинки роботи.

Швидкість відкриття додатка.

Було перевірено швидкість запуску додатка, за результатами якого час запуску не перевищував 5 секунд.

Захист бази даних.

Було перевірено можливість внесення у базу даних SQL-запитів через можливість створення і редагування заміток. За результатами спроби додаток є захищеним від ін’єкцій і зберігає їх як звичайний текст.

Висновки до розділу 3

У даному розділі було реалізовано процес написання коду для створення мобільного додатка. Описано основні функції і методи, які були використані при реалізації програмного забезпечення. Описано спосіб підключення сторонніх сервісів для прискорення розробки. Також показано, як можна створити інсталяційний пакет в Android Studio.

Були описані мінімальні технічні характеристики для роботи додатка на мобільному пристрої і написана інструкція для користування ним. Також було проведено тестування усіх доступних користувачеві функцій. Зроблено спробу зробити SQL-ін’єкцію, і був підтверджений захист додатка від цього методу злому.

ВИСНОВКИ

Під час створення кваліфікаційної роботи було розроблене програмне забезпечення для операційної системи Android, а саме додаток «TimeSage».

На першому етапі даної роботи було проведено аналіз існуючих аналогів мобільного застосунку. Також був проведений аналіз проблеми прокрастинації і поставлені цілі для розробки додатка.

На наступному етапі був проведений аналіз засобів для розробки програмного забезпечення, а саме:

– середовище розробки;
– мову програмування;
– базу даних.

В результаті аналізу було здійснено вибір засобів для розробки мобільного додатка, а саме Android Studio, мова програмування Kotlin, база даних Firebase.

Здійснено розробку додатка тайм-менеджера, а саме: створено графічну частину додатку, приєднано його до хмарної бази даних, також реалізовано перехід між екранами за допомогою навігаційного компоненту. Створено таймер зворотного відліку з можливістю задання будь-якого проміжку часу для відліку, також реалізовані кнопки запуску таймеру його паузи і продовження.

Створено екран-заставку, яка з’являється при кожному запуску додатку, додано можливість реєстрації і аутентифікації для збереження нотаток в хмарній базі даних і його відображення на різних мобільних пристроях. Також додано можливість редагування і видалення заміток.

Також було проведено тестування функцій додатку, таких як реєстрація, авторизація, створення і видалення заміток, можливість редагування, роботу таймера і безпеку додатку від SQL-ін’єкції. Усі пункти тестування додаток пройшов успішно.

Функціонал програми на даному етапі свого існування є досить простий порівняно з аналогами, тому його можна удосконалювати, надавши можливість

створювати замітки на інших платформах наприклад Windows або Linux.

Переваги мобільного додатку в порівняні з аналогами:

– простий дизайн: максимально простий дизайн допомагає інтуїтивно ним користуватися;

– відсутність відволікань: завдяки повноекранному режиму зменшуються подразники на які можна відволіктися;

– синхронізація: можливість створювати замітки на різних пристроях на операційній системі Android;

– можливість поставити таймер на паузу;

– можливість задати будь-який проміжок часу для відліку для таймера.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Прокрастинація: причини, наслідки і шляхи подолання. URL: http://surl.li/ukkrs (дата звернення: 11.03.2024).

2. Що таке метод Pomodoro? – Dropbox. URL: https://experience.dropbox.com/uk-ua/resources/pomodoro-technique (дата звернення: 11.05.2024).

3. Google Keep. URL: https://keep.google.com/ (дата звернення: 20.02.2024).

4. Your connected workspace for wiki, docs & projects | Notion. URL: https://www.notion.so/ (дата звернення: 21.02.2024).

5. Forest. URL: https://www.forestapp.cc/ (дата звернення: 01.02.2024).

6. To Do List and Task Management App | Microsoft To Do. URL: https://www.microsoft.com/en-us/microsoft-365/microsoft-to-do-list-app (дата звернення: 02.03.2024).

7. Download Android Studio & App Tools - Android Developers. https://developer.android.com/studio (дата звернення: 25.03.2024).

8. Firebase | Google's Mobile and Web App Development Platform. URL: https://firebase.google.com/ (дата звернення: 01.03.2024).

9. Kotlin Programming Language. URL: https://kotlinlang.org/ (дата звернення: 01.02.2024).

10. Мова програмування Kotlin – сфери застосування та з чого почати вивчення – Lemon.School. URL: https://lemon.school/blog/mova-programuvannya-kotlin (дата звернення: 01.05.2024).

11. Огляд можливостей kotlin та java для розробки android додатку. URL: http://ir.lib.vntu.edu.ua/bitstream/handle/123456789/27263/7164.pdf?sequence=3 (дата звернення: 01.02.2024).

12. Greenhalgh D., Bailey A., Skeen J. Kotlin Programming: The Big Nerd Ranch Guide. 2021. URL: http://surl.li/ukrla (дата звернення: 02.03.2024).

13. Едблад П.. Самодисципліна. Як боротися з прокрастинацією, досягати мети і отримувати задоволення від життя 2023. URL: http://surl.li/ukkqr (дата звернення: 20.03.2024).