МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра Комп'ютерних систем та мереж

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
комп'ютерних систем та мереж

(Литвиненко О.Є.)

«___» _____ 2021 р.

ДИПЛОМНИЙ ПРОЄКТ

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
"БАКАЛАВР"

Тема: Мобільний додаток для читання книг іноземною мовою з можливістю перекладу невідомих слів

Виконавець: Гуцалюк А.С..

Керівник: Сябрук І.М.

Нормоконтролер: Тупота Є.В.

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютерних систем та мереж
Напрям (спеціальність) 123 "Комп'ютерна інженерія"

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерних систем та мереж

(Литвиненко О.Є.)

«___» _____ 2021 р.

З А В Д А Н Н Я
на виконання дипломного проекту

Гуцалюка Артема Сергійовича

1. Тема проекту (роботи): Мобільний додаток для читання книг іноземною мовою з можливістю перекладу невідомих слів системам.
затверджена наказом ректора від "02" лютого 2021 року № 135/ст.
2.Термін виконання проекту (роботи): з 17.05.2021 до 20.06.2021
3.Вихідні дані до проекту (роботи): розробити програмний продукт за допомогою наступних засобів: Visual Studio 2019, Android Studio та БД MongoDB.
4.Зміст пояснювальної записки (перелік питань, що підлягають розробці):

1. Аналіз предметної області
2. Встановлення вимог до розроблюваного програмного засобу
3. Проектування засобу
4. Прототип засобу

5.Перелік обов'язкового графічного матеріалу:

Презентація *PowerPoint*, діаграма прецендентів, діаграма послідовності виконання, схема архітектури програмного продукту, демонстрація застосунку

6. Календарний план

№ п/п	Етапи виконання дипломного проекту	Термін виконання етапів	Примітка
1	Ознайомлення з постановкою задачі та вивчення літератури Написання 1 розділу, представлення керівнику	11.05.21 – 12.05.21	
2	Попередній друк 1 розділу та допоміжних сторінок (чернетка) - титульної, завдання, графіка, реферат, список скорочень, зміст, вступ, список джерел. 1-ий нормо-контроль.	13.05.21 – 17.05.21	
3	Написання 2 розділу, представлення керівнику	18.05.21 – 22.05.21	
4	Написання 3 розділу, представлення керівнику	22.05.21 – 25.05.21	
5	Написання 4 розділу, представлення керівнику	25.05.21 – 27.05.21	
6	Загальне редагування та друк пояснювальної записки, графічного матеріалу	28.05.21 – 31.05.21	
7	Проходження нормо-контролю, перепліт пояснювальної записки.	31.05.21 – 03.06.21	
8	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації	03.06.21 – 05.06.21	
9	Отримання відгуку керівника, рецензії.	06.06.21 – 08.06-21	
10	Підготовка матеріалів для передачі секретарю ДЕК (ПЗ, ГМ, CD-R з електронними копіями ПЗ, ГМ, презентації, відгук керівника, рецензія, довідка про успішність, 2 папки, 2 конверта)	10.06.21 – 15.06.21	
11	Захист дипломного проекту	14.06.21 – 18.06.21	

7. Дата отримання завдання « 20 » 04 2021 р.

Керівник дипломного проекту Сябрук І.М.
(підпис)

Завдання прийняв до виконання Гуцалюк А.С.
(підпис студента)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту “Мобільний додаток для читання книг іноземною мовою з можливістю перекладу невідомих слів”: 62 с., 14 рис., 1 таблиця, 17 літературних джерел, 1 додаток.

електронний документ, електронна книга, читання з перекладом android клієнт, windows клієнт.

Мета дипломного проекту – проаналізувати основні засоби для читання електронних книг та розробити мобільний додаток для читання книг іноземною мовою з можливістю перекладу невідомих слів.

Об’єкт проектування – програмний засіб для читання книг іноземною мовою з можливістю перекладу невідомих слів.

Предмет проектування – Android додаток для читання книг іноземною мовою з можливістю перекладу невідомих слів.

Метод проектування – об’єктно-орієнтовний.

Прогнози припущення щодо розвитку об’єкта дослідження – створення робочого зразка програмного засобу та використання його в мобільному телефоні.

Результати дипломного проектування рекомендується використовувати при розробці нових програмних засобів, які надають можливість читання книг з можливістю їх перекладу кількома мовами.

інформаційної безпеки власної чи корпоративної інформаційної системи

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

E-book – електронна книга

E-document – електронний документ

UI – Інтерфейс користувача

UML – уніфікована мова моделювання

MVVM – Model - View - ViewModel

API - Application Programming Interface

REST - Representation State Transfer or "transferring view state"

WPF – Windows Presentation Foundation

ВСТУП

Комп’ютерні технології мають значний вплив на процеси сприйняття інформації. Але варто зазначити, що єдиною інтелектуальною технологією оволодіння знаннями, накопиченими людством, все ще залишається читання, незалежно від того, як воно здійснюється: з аркуша, з екрану, на слух тощо. Але не можна не враховувати той факт, що в сучасних соціокультурних умовах читання сама парадигма, вона все більше стає «екраном». Це вимагає пошуку нових способів подання інформації, які сприяли б ефективному сприйняттю змісту книги з екрану. Різноманітність форм електронних комунікацій, швидке збільшення електронного вмісту на книжковому ринку породжують багато проблем щодо ідентифікації, обліку, зберігання, розповсюдження електронної книжкової продукції, відкритого доступу до її вмісту, а також питань авторських прав. В даний час ці проблеми є актуальними і потребують швидкого вирішення.

Виходячи з цього, існує потреба у розробці електронних книг, які відповідають новим реаліям розвитку суспільства та використовують усі переваги інформаційних технологій.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Основні характеристики електронної книги

Поняття Е-книги можна визначити як сукупність, що складається з електронного документа та електронного зчитувача (його ще називають електронною книгою та електронною книгою).

Тим часом термін "електронний документ" з'явився в дослідженнях приблизно в 1970-х роках і тісно пов'язаний з поняттям "машиночитаних документів". Машиночитаний документ розглядався як «документ, придатний для автоматичного зчитування інформації, що міститься там».

Високий попит читачів на електронну книжкову продукцію значною мірою зумовлений незаперечними перевагами електронної книги над друкованою. Основними традиційно вважаються:

1) наявність в електронній книзі пошукової системи, яка дозволяє знайти інформацію, яка вам потрібна, за дуже короткий час, перебуваючи в будь-якій точці світу;

2) доступність та ефективність: електронні технології значно прискорили та спростили процеси книговидання та розповсюдження книг;

3) економічні вигоди: виробництво електронних книг набагато дешевше, ніж друкованих;

4) екологічні переваги: при виробництві електронних книг немає негативного впливу на навколишнє середовище;

5) безпека публікацій;

6) компактність;

7) мобільність;

8) явною перевагою є супровід тексту звуковими та відеоматеріалами, ілюстрації можуть бути анімованими, що полегшує процес пояснення складних процесів, легко адаптує зміст книги до індивідуального сприйняття та приваблює сучасного читача;

9) можливість, іноді єдина, знайти рідкісну книгу, що не продається;

10) немає необхідності передруковувати: зміст електронних книг може постійно оновлюватися авторами;

11) інтерактивність та гіпертекст, які визначають рівень спілкування між автором та читачем, який якісно відрізняється від друкованої книги;

12) комфорт і компактність: сотні текстів книг тощо можуть поміститися в маленькому комп'ютерному пристрої.

Поряд із перевагами електронної книги існує ряд недоліків, серед яких зазвичай вказується:

1) незручність читання з екрану;

2) пристосованість психіки людини до лінійного читання;

3) швидке старіння комп'ютерних технологій;

4) залежність читання від електроенергії тощо.

Але більшість цих факторів походять від технологічних причин і усуваються або будуть усунені в процесі розвитку інформаційних технологій. Найбільш суттєвим з недоліків можна вважати те, що електронний текст гірший за друкований текст, сприймається і поглинається, читання з екрану не доставляє читачеві такого задоволення, як читання друкованої книги. Але ці факти не можна вважати незаперечними.

Наприклад, багато читачів, особливо серед молоді, вважають за краще використовувати електронні книги, «завантажені» на гаджети з Інтернету, і не відчувають незручностей.

Поширенню електронної книги сприяють значні зміни, що відбулися в процесах виробництва та сприйняття інформації. Медіа-технології спричиняють серйозні зміни в психології читання та впливають на вибір засобів комунікації.

Тому переваги електронної книги для багатьох сучасних читачів є визначальними, і в майбутньому кількість її користувачів збільшиться.

З точки зору бібліотеки як власника електронних документів, важливо, щоб користувач міг прочитати «виданий» йому електронний документ (у

форматі, в якому він зберігається в бібліотеці, або у форматі, в якому бібліотекар може перетворити його). Звичайно, спеціалізовані пристрої мають свої переваги перед універсальними, і ми будемо на них звертати увагу, але не обмежуватимемось ними. Більше того, приділіть особливу увагу «читачам» та «перетворювачам», що допоможе максимально розширити коло потенційних користувачів електронного документа.

Зростаючий інтерес до електронних архівів зрозумілий. Переклад документації в електронну форму дозволяє:

• значно спростити та пришвидшити роботу з ним (зокрема, за рахунок майже миттєвого пошуку потрібного документа за будь-якою його деталлю);

• організувати обмін документацією між відділами, підрозділами, філіями компанії із застосуванням сучасних засобів зв'язку;

• вільне офісне приміщення від купи паперів;

• надавати різним працівникам різні права доступу до документації, забезпечувати належний ступінь конфіденційності;

• нарешті, забезпечити інформацію, що міститься в документах, надійним захистом від спотворення або втрати (як у випадку форс-мажорних обставин, так і в результаті незаконних дій).

Все більша кількість організацій та підприємств неминуче вдається до переходу від паперового робочого процесу до електронного. Система електронного документообігу дозволяє швидко отримати доступ до інформації і тим самим оптимізувати роботу працівників компанії. Відомо, що продуктивність праці фахівців компанії, що використовує електронну систему документообігу, зростає приблизно в 20 разів.

Електронні книги також вирішують проблему простору кімнати. Багато електронних книг можна зберігати на комп’ютері. Деякі бібліотеки роблять це, сканують паперові книги та перекладають в електронний формат. Традиційні книги займають багато місця в кімнаті. Для паперових книжок потрібно купувати полички або шафи, а це також доставляє незручності. Книга в електронному форматі не займає місця в просторі квартири. Сучасні

рідкокристалічні монітори не опромінюють і не впливають на зір негативно. На комп'ютері ви можете зробити шрифт легким для читання, і тоді вам не потрібно буде вдивлятися дрібними літерами. Друковані книги часто друкують дрібним шрифтом, і людям із різними вадами зору незручно читати. Раніше однією з переваг паперових книжок було те, що їх можна було брати з собою, куди б ви не пішли. З появою ноутбуків, КПК, електронних блокнотів, питання мобільності книг в їх електронному вигляді слід вважати вирішеним.

Нещодавно виникло запитання: чому електронні пристрої для читання та планшетні комп'ютери так наполегливо просуваються? Відповідь на це запитання досить проста. Ці сучасні технічні пристрої просто пристосовуються до потреб часу. Але друковані книги все ще є невід'ємною частиною життя людей.

Недавні дослідження дослідницького центру Pew показують, що за останні два роки кількість людей, які вважають за краще читати електронні книги, зросла в чотири рази. Це сталося тому, що електронну версію можна читати на різних типах пристроїв, включаючи дешеві (Kindle, Nook) та дорогі планшети (iPad), смартфони та дешевші електронні зчитувачі.

Дослідники Дослідницького центру Pew роблять висновок, що читачі поступово освоюють новий формат книг, і значна кількість людей читає більше, адже тепер книги можна завантажувати будь-коли та де завгодно.

Відсоток тих, хто використовував друковану книгу та її електронну версію за 2019 рік, і хто вважав, що цей формат найкраще підходить для цих цілей (рис. 1.1) [2]:

1) читання з дитиною
2) можливість ділитися книгами
3) читання книги в ліжку
4) можливість широкого вибору книг
5) читання книг під час подорожі
6) можливість швидкого доступу до книги

Таким чином, можна припустити, що тема електронних документів широко поширена і продовжує зростати щодня. Однак читати їх на комп'ютері не завжди зручно, але носити з собою електронну книгу і якось надсилати необхідний документ там складніше. Тим не менше, існує безліч додатків, які перетворюють ваш планшет або смартфон на «читач». Але багатьом з них не вистачає синхронізації між пристроями. Читайте на планшеті, продовжуйте смартфон чи ноутбук з тієї самої сторінки, на якій ви зупинялися на попередньому пристрої. Для цього, звичайно, пристрій повинен мати доступ до Інтернету. Але це потрібно лише для швидкої синхронізації, ви можете читати книги в автономному режимі, вимкнувши всі модулі зв'язку для економії заряду акумулятора та трафіку.

Основна перевага програм для читання електронних матеріалів автоматично випливає з їх призначення - забезпечити легкість читання, в першу чергу завдяки спеціально підібраним шрифтам і можливості швидкого налаштування їх для конкретного користувача, що дозволяє постійно читати великі обсяги тексту, не відчуваючи втома. Але справа не тільки в шрифтах, хоча в принципі процес читання електронних книг мало чим відрізняється від читання паперових - одна і та ж сторінка, одна і та ж сторінка, але лише за допомогою кнопок або миші.

Рис. 1.1. Співвідношення використання електронних книг і паперових в різних життєвих ситуаціях

Багато програм для читання можуть автоматично перегортати текст (це називається автоматичним прокручуванням), тобто зміщувати текст вгору з певною частотою.

Крім того, читаючи електронну книгу в будь-якій з вищезазначених програм, користувачеві не потрібно використовувати закладки або згинати кути біля сторінок, щоб запам'ятати місце закінчення читання - програма запам'ятає номер сторінки та відкриє книгу на це наступного разу, коли він почнеться. "Читачі" програм чудово запам'ятовують усі книги, які нещодавно читав користувач, і швидко відкривають будь-яку з них із так званої бібліотеки. З читабельних книг користувач легко копіює абзаци та швидко знаходить у них потрібні фрагменти, використовуючи знайомий пошук. Крім того, деякі програми в тій чи іншій формі дозволяють вставляти примітки до тексту.

Висновки по розділу

В даному розділі зроблений аналіз предметної області, в результаті чого були встановлені основні характеристики електронних книг, їх переваги та недоліки.

РОЗДІЛ 2

ВСТАНОВЛЕННЯ ВИМОГ ДО РОЗРОБЛЮВАНОГО ПРОГРАМНОГО ЗАСОБУ

2.1 Технічне завдання

Глосарій стандарту IEEE з термінології програмної інженерії визначає вимогу як [3]:

1. Умова або здатність, необхідна користувачеві для вирішення проблеми або досягнення мети.

2. Умова або здатність, яким повинна відповідати або якою повинна володіти система або компонент системи, щоб задовольнити контракт, стандарт, специфікацію або інший офіційно встановлений документ.

3. Задокументоване представлення стану або можливостей, як в 1 або 2.

Вимоги можна класифікувати на функціональні та нефункціональні вимоги.

Функціональні вимоги – це перелік послуг, які повинна виконувати система, і має бути зазначено, як система реагує на певні вхідні дані, як вона поводиться у певних ситуаціях тощо. У деяких випадках вказується, що система не повинна цього робити.

Стандартні форми специфікації функціональних вимог:

• Опис функції або об'єкта.

• Опис вхідних даних та їх джерел.

• Опис вихідних даних із зазначенням місця їх призначення.

• Вказівка на те, що потрібно для виконання функції.

• Якщо це специфікація функції, необхідно описати передумови (передумови), які повинні бути виконані до того, як функція буде

викликається, та опис кінцевої умови (постумови), якій необхідно відповідати після завершення функції.

• Опис побічних ефектів (якщо такі є).

Функціональні вимоги визначають функціональність програмного забезпечення, яке розробники повинні будувати, щоб користувачі могли виконувати свої завдання в рамках бізнес-вимог.

Нефункціональні вимоги - опишіть характеристики системи та її середовища, а не поведінку системи. Тут також можна навести перелік обмежень, що накладаються на дії та функції, що виконуються системою.

Сюди входять обмеження в часі, обмеження процесу розробки системи, стандарти тощо.

Нефункціональні вимоги безпосередньо не пов’язані з функціями, які виконує система. Вони пов'язані з такими властивостями інтеграції системи, як надійність, час відгуку або розмір системи. Крім того, нефункціональні вимоги можуть визначати системні обмеження, такі як пропускна здатність пристроїв вводу-виводу або формати даних, що використовуються в системному інтерфейсі.

Нефункціональні вимоги відображають наступні вимоги користувача: вони базуються на бюджетних обмеженнях, враховують організаційні можливості компанії-розробника, можливість взаємодії розробленої системи з іншим програмним забезпеченням та комп’ютерними системами, а також зовнішні фактори, такі як правила безпеки, законодавство про захист інтелектуальної власності тощо

Нефункціональні вимоги описують цілі та ознаки якості. Атрибути якості - це додатковий опис особливостей товару, що виражається через опис його характеристик, важливих для користувачів або розробників. Ці характеристики включають:

- легкість і зручність використання;
- легкість пересування;
- цілісність;
- ефективність та відмовостійкість;
- зовнішні взаємодії між системою та зовнішнім світом;

• обмеження проектування та впровадження. Обмеження (обмеження) стосуються вибору можливості розвитку зовнішнього вигляду та структури товару.

Таблиця 1.1

Основні функціональні вимоги системи

№	**Опис вимог**
REQ 1	Заявка повинна надавати можливість відкрити текстовий файл
REQ 2	Заявка повинна містити опції в головному меню
REQ 3	Додаток надає можливість користувачеві видалити документ
REQ 4	Додаток надає можливість користувачеві завантажувати документ
REQ 5	Додаток надає можливість користувачеві додавати закладки
REQ 6	Додаток надає можливість користувачеві видалити закладку
REQ 7	Програма повинна надавати можливість синхронізувати інформацію про книги в пристроях, підключених до одного облікового запису
REQ 8	Додаток надає можливість користувачеві створити новий обліковий запис
REQ 9	Додаток повинен надавати можливість користувачеві переключатися між обліковими записами
REQ 10	Додаток надає можливість користувачеві створювати окремі колекції для книг
REQ 11	Заявка повинна надавати можливість керувати колекціями
REQ 11.1	Заявка повинна надавати можливість додавати книги
REQ 11.2	Заявка повинна надавати можливість видаляти книги
REQ 11.3	Заявка повинна надавати можливість редагувати назву
REQ 12	Додаток надає можливість користувачеві виділяти текст
REQ 13	Додаток надає можливість користувачеві перекладати виділений фрагмент (з англійської мови на українську)

REQ 14	Заявка повинна надавати можливість зберегти останню позицію в документі
REQ 15	Додаток надає можливість користувачеві шукати певну книгу
REQ 16	Додаток надає можливість зберігати всі завантажені книги в головному меню

Наведемо основні функціональні вимоги за допомогою діаграми прецендентів, яка на ведена на рис. 2.1.

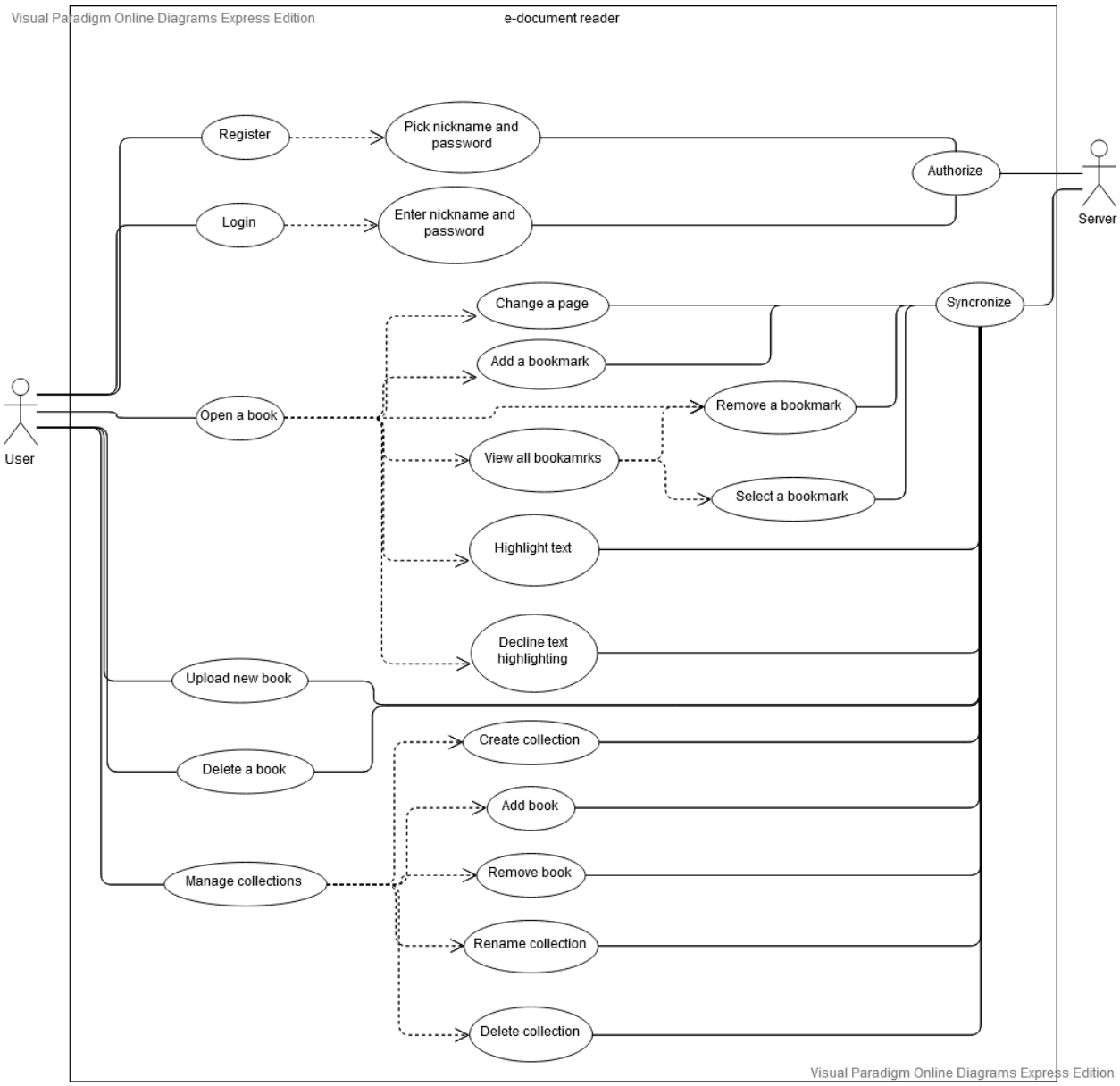

Рис. 2.1. Діаграма прецендентів

Діаграма прецендетнів – це клас діаграм, які описують функції системи, коли вона взаємодіє з кимось (або чимось) із зовнішнього середовища.

Нижче можна знайти 5 специфікації прецендентів відповідної діаграми, пов’язаних із програмою читання книг іноземною мовою з можливістю перекладу невідомих слів, які є типовими сценаріями використання, а також його представлення UML.

Прецендент “Запуск мобільного застосунку”

Назва:	Запуск мобільного застосунку
Ідентифікатор:	UC1
Опис:	
Актор:	Користувач
Перед-умова	Програмний засіб попередньо встановлений на пристрої
Основний потік:	1. Випадок використання починається, коли користувач запускає виконуваний файл 2. Додаток запускається та відкриває головне меню
Пост-умова:	
Альтернативний потік:	
Примітка:	

Прецендент “Завантаження книги”

Назва:	Завантаження книги
Ідентифікатор:	UC2
Опис:	
Актор:	Користувач
Перед-умова	Програмний засіб відкритий
Основний потік:	1. Користувач натискає кнопку «+» у правому куті 2. Відкриється вікно з виділеним файлом для завантаження 3. Користувач вибирає файл і натискає кнопку “OK” 4. Файл успішно завантажений і з’являється в головному меню
Пост-умова:	
Альтернативний потік:	**Випадок А:** 1. Файл не завантажується через деяку невідповідність (невірний формат файлу, заборонений доступ до файлу тощо) 2. З’являється попереджувальне повідомлення
Примітка:	

Прецендент “Робота з книгою”

Назва:	Робота з книгою
Ідентифікатор:	UC3
Опис:	

Актор:	Користувач
Перед-умова	Додаток уже відкритий, і книга завантажена
Основний потік:	1. Користувач відкриває книгу в головному вікні 2. Перегортає кілька сторінок 3. Встановлює закладку 4. Перегортає ще кілька сторінок 5. Відкриває всі закладки та вибирає в них закладку 6. Бачить, що ця сторінка перевернута із закладки 7. Користувач видаляє закладку 8. Користувач вибирає одне речення та виділяє його 10. Користувач натискає на речення і затримує на 2 сек 11. Користувач бачить переклад виділеного фрагменту 9. Користувач бачить переклад і відхиляє виділення речень
Пост-умова:	
Альтернативний потік:	
Примітка:	

Прецендент “Робота з колекціями”

Назва:	Робота з колекціями
Ідентифікатор:	UC4
Опис:	
Актор:	Користувач
Перед-умова	Додаток уже відкрито та завантажено 3 книги
Основний потік:	1. Користувач створює колекцію та називає її "Тестова колекція" 2. Користувач переходить до “Тестової колекції” 3. Бачить, що колекція порожня 4. Користувач натискає кнопку «+» у правому куті кнопки 5. Відкривається вікно зі списком завантажених книг 6. Користувач вибирає дві книги та натискає кнопку “Додати” 7. Дві книги, додані до колекції 8. Користувач вибирає одну книгу та вилучає її з колекції 9. Одна книга була вилучена з колекції 10. Користувач переходить до налаштувань колекції та змінює свою назву на “Звичайні книги” 11. Користувач повертається до сторінки колекції та бачить заголовок “Звичайні книги” замість “Тестова колекція” 12. Користувач переходить до налаштувань колекції та видаляє його 13. З'явиться запит із підтвердженням 14. Користувач натискає “ОК”

	15. Колекція видаляється, всі книги залишаються незмінними
Пост-умова:	
Альтернативний потік:	Випадок А: 1. Користувач натискає кнопку “Скасувати” 2. Бачить вікно із закритими книгами, а колекція залишається порожньою
Примітка:	

Прецендент “Використання різних пристроїв”

Назва:	Використання різних пристроїв
Ідентифікатор:	UC5
Опис:	
Актор:	Користувач, система
Перед-умова	Програма вже попередньо встановлена на телефоні та на ПК
Основний потік:	1. Користувач відкриває програму на ПК 2. Вхід користувачів до облікового запису 3. Користувач завантажує книгу 4. Користувач відкриває програму на телефоні 5. Вхід користувачів до облікового запису 6. Система синхронізує дані телефону з даними сервера 7. Користувач бачить нещодавно додану книгу
Пост-умова:	
Альтернативний потік:	Випадок А: 1. Відсутнє підключення до Інтернету 2. З'являється відповідне повідомлення «Немає доступу до Інтернету»
Примітка:	Якщо користувач вже ввійшов у систему та завантажив документ без підключення до Інтернету, він буде завантажений у базу даних після встановлення з’єднання.

2.2 Встановлення нефункціональних вимог

2. Після встановлення функціональних вимог, необхідно встановити також не функціональні вимоги, які відповідають за якість ПЗ. Розроблені не функціональні вимоги представлені в табл. 2.2.

3. Таблиця 2.2

4. Нефункціональні вимоги до засобу

Назва	Опис вимоги
Обмеження платформи	Додаток повинен бути доступним для використання як на платформах Windows, так і на Android

Налаштування	Додаток слід встановлювати з файлу .exe (для ОС Windows) або з файлу .apk (для ОС Android)
Доступність	Додаток повинен бути доступний у будь-який час
Надійність	Програма повинна реагувати на будь-яку помилку, що сталася в ній, та виконувати відповідні дії. Коли помилка сталася через помилку всередині системи, вона повинна повідомити користувача про відкриття відповідного вікна та, залежно від типу помилки, закрити програму або повернутися до попереднього кроку користувача.
Тривалість	Додаток повинен забезпечувати захист даних користувачів, доки вони існують.
Маштабованість	Система повинна відображатися правильно у всіх можливих розмірах екрана, обертанні екрана та забезпечувати спеціальні версії для двох основних платформ: телефону та комп'ютера.
Зручність використання	Додаток повинен мати зручний для користувача інтерфейс / інтерфейс UX
Безпека	Додаток повинен забезпечувати захист усієї конфіденційної інформації користувача, як логін та пароль
Виконуваність	Всі дії з БД повинні виконуватися в окремому потоці, відповідь інтерфейсу повинна бути швидкою і повідомляти користувача про затримки у випадках, якщо дія не може бути закінчена більше 5 секунд
Розширюваність	Додаток повинен забезпечувати розширюваність лише розробником через повне оновлення системи
Локалізація	Мова інтерфейсу – англійська та українська

Висновки по розділу

У цій главі були визначені функціональні та нефункціональні вимоги. Специфікації були створені для таких сценаріїв: запуск програми, завантаження книги, маніпуляції з книгою та колекцією та використання різних пристроїв. Дані функціональні вимоги були представлені за допомогою UML. Для детального розгляду системних процесів були створені та описані відповідні діаграми.

РОЗДІЛ 3

ПРОЕКТУВАННЯ ЗАСОБУ

3.1 Діаграма послідовності виконання

Діаграми послідовності UML - це діаграми взаємодії, що детально описують спосіб виконання операцій. Вони фіксують взаємодію між об’єктами в контексті співпраці. Діаграми послідовності - це фокус на часі, і вони наочно показують порядок взаємодії, використовуючи вертикальну вісь діаграми, щоб представити час, які повідомлення надсилаються та коли [4].

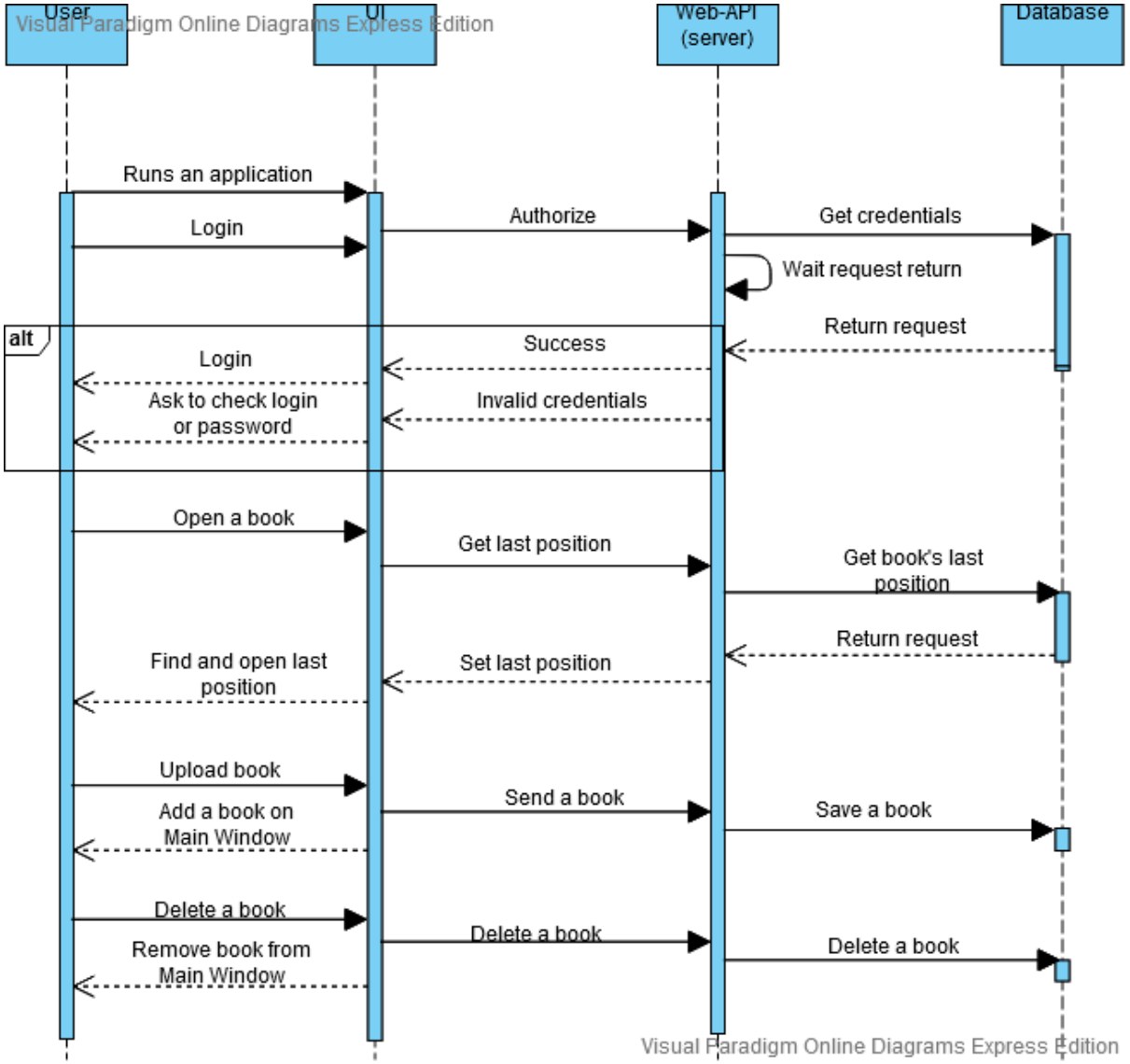

Рис. 3.1. Діаграма послідовності виконання

З UML діаграми (рис. 3.1) видно, що UI, тобто програма та сервер мають однаковий життєвий цикл. Це означає, що програма не може повноцінно функціонувати без зв’язку з сервером.

Користувач запускає програму, а потім входить в обліковий запис. Потім сервер перевіряє облікові дані користувача шляхом порівняння з даними в базі даних. А після цього повертається відповідь. Якщо логін і пароль правильні - користувач отримує успішний вхід та синхронізує дані програми з базою даних. В іншому випадку користувачеві буде запропоновано перевірити облікові дані та спробувати ввійти ще раз.

Після успішного входу користувач може продовжити свою роботу з додатком та відкрити книгу для читання. Отже, програма знайде останню позицію та відкриє її для користувача.

Завантаження та видалення книги ведуть себе подібним чином. Після того, як вхідна інформація користувача надсилається на сервер, де вона загортається у відповідний запит SQL, а в кінцевому підсумку запит SQL виконується в базі даних, оновлюючи свої дані.

Для несанкціонованої послідовності користувачів набагато простіше, оскільки програма працює лише з даними користувача та локального кешу. Вся інформація про завантажені книги, її закладки, остання позиція тощо піддається серіалізації у кеш-файл у папці програми, а за допомогою десеріалізації отримується необхідна інформація про книгу.

3.2 Архітектура програмного засобу

Архітектура програмного забезпечення – це процес перетворення таких функцій програмного забезпечення, як гнучкість, масштабованість, впровадження, повторне використання та безпека, в структуроване рішення, яке відповідає як технічним, так і бізнес-вимогам.

На рівні розробки архітектури додатків слід вирішити такі основні завдання:

• поліпшення та підвищення продуктивності процесів;

• зниження витрат;

• підвищення експлуатаційних показників;

• підвищення ефективності управління;

• зниження ризику;

• підвищення ефективності роботи 1Т агрегатів;

• підвищення продуктивності роботи користувачів;

• поліпшення здатності та прозорості взаємодії

• зменшення витрат на "підтримку" життєвого циклу;

• покращені функції безпеки;

• підвищення керованості.

Для E-Document Reader найбільш підходящою архітектурою є архітектура клієнт-сервер (рис. 3.2)

Він складається з трьох основних елементів: клієнта, Інтернету та сервера. І клієнт, і сервер є абсолютно незалежними одиницями в цій архітектурі. Сервер одночасно обробляє дані кількох клієнтів, а клієнти, в свою чергу, не знають один про одного.

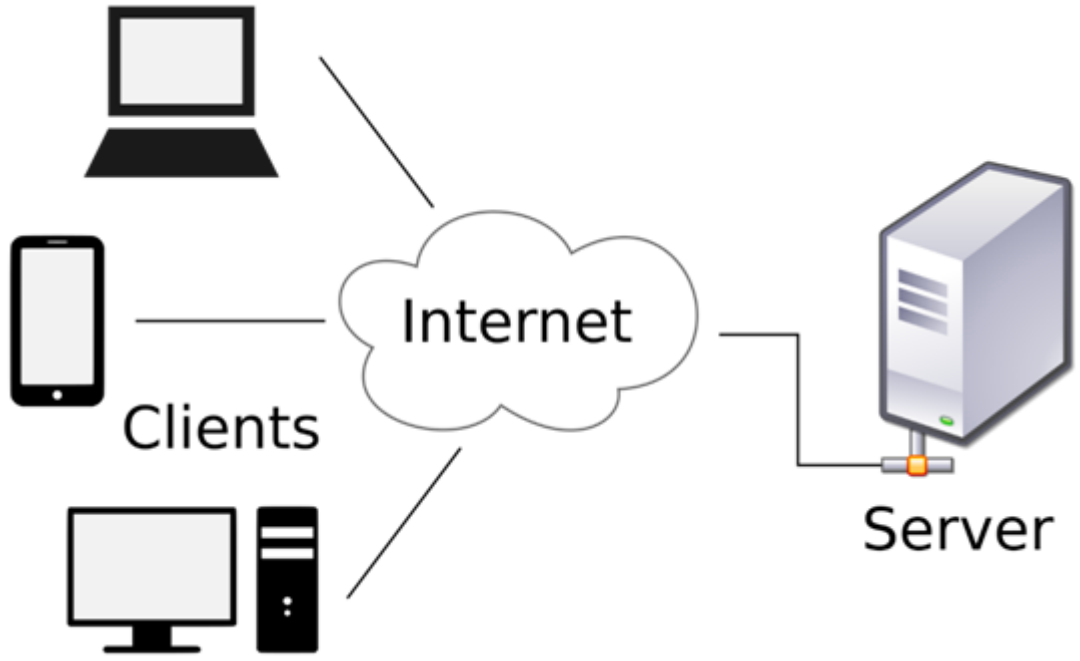

Рис. 3.2. Клієнт-серверна архітектура

Більш детальну архітектуру засобу можна побачити нижче на рис 3.3. Тут чіткіше показано зв’язок між елементами моделі. Таким чином, користувач вводить деякі дані за допомогою інтерфейсу. Далі ці дані обгортає сама програма, а потім запит надсилається на сервер.

Важливо зазначити, що основою сервера є веб-API (веб-API буде детально описаний нижче). Web API, у свою чергу, працює безпосередньо з базою даних. На вході він приймає певний набір параметрів, тип запиту, а потім обертає ці дані у запит SQL. Який потім надсилається до бази даних.

Далі база даних надсилає результат на сервер, сервер розгортає їх у прийнятних для клієнта даних, і програма повертає ці дані. В кінцевому підсумку програма відображає результат на екрані пристрою.

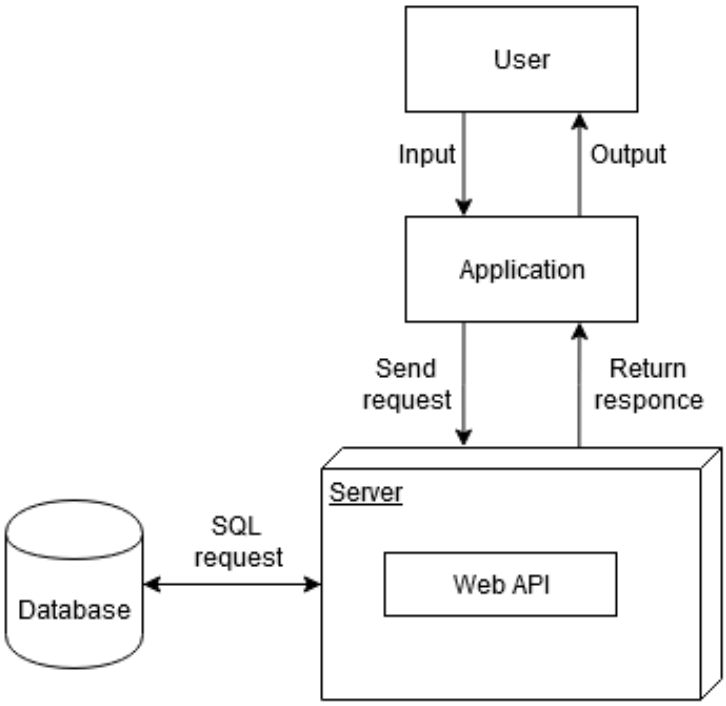

Рис. 3.3. Архітектура засобу

Розробка нашого засобу умовно була розділена на три частини: написання серверної частини та створення бази даних, створення WPF застосунку та створення Android застосунку. WPF та Android не пов'язані між собою і представляють клієнта для певної ОС. У той же час серверна частина

і база даних у цьому проекті відіграють важливу роль і з'єднують клієнтів за допомогою авторизації.

3.3 Інструментальні засоби для розробки

Для розробки засобу використовувались такі програмні засоби:

- Visual Studio 2019;
- Android Studio;
- MongoDB;

Вбудоване середовище розробки Visual Studio - це креативна стартова платформа, якуможна використовувати для редагування, налагодження та побудови коду, а потім опублікувати програму. Інтегроване середовище розробки (IDE) – це багатофункціональна програма, яка може бути використана для багатьох аспектів розробки програмного забезпечення. Крім стандартного редактора та налагоджувача, які надає більшість середовищ розробки, Visual Studio включає компілятори, засоби доробки коду, графічні дизайнери та багато інших функцій для полегшення процесу розробки програмного забезпечення. [6]

C # був обраний основною мовою для розробки програми. C # розроблений та розроблений спеціально для використання з .NET Framework. Призначення .NET Framework - служити середовищем для підтримки розробки та виконання високо розподілених компонентних програм. Він забезпечує спільний доступ до різних мов програмування, а також безпеку, портативність програм та загальну модель програмування для платформи Windows.

Android Studio

Існує кілька способів розробки додатків для Android, але сьогодні офіційним і найпопулярнішим способом є Android Studio. Він забезпечує інтерфейс для створення додатків та піклується про більшість складних файлових файлів.

MongoDB

MongoDB впроваджує новий підхід до побудови баз даних, де відсутні таблиці, схеми, SQL-запити, зовнішні ключі та багато іншого, що властиво об'єктно-реляційним базам даних.

На відміну від реляційних баз даних, MongoDB пропонує документ-орієнтовану модель даних, яка робить MongoDB швидшим, масштабованішим та простішим у використанні.

3.4 Діаграма компонентів

Переглянувши основні інструменти, можна перейти до більш детального процесу розробки та його опису. І як було помічено раніше, процес розробки був розділений на три етапи:

1) Сервер

Метою цього етапу було створення бази даних та написання веб-API для подальшої комфортної роботи над реалізацією клієнтів.

Веб-API забезпечує спосіб створення додатка, спеціально розробленого для роботи в стилі REST. Архітектура REST передбачає використання наступних методів або типів HTTP-запитів для взаємодії з сервером:

- Отримати
- Опублікувати
- Покласти
- Видалити

Веб-API - це веб-служба, до якої можуть отримати доступ інші програми. Таким чином, веб-API служить своєрідним шаром між базою даних та клієнтом.

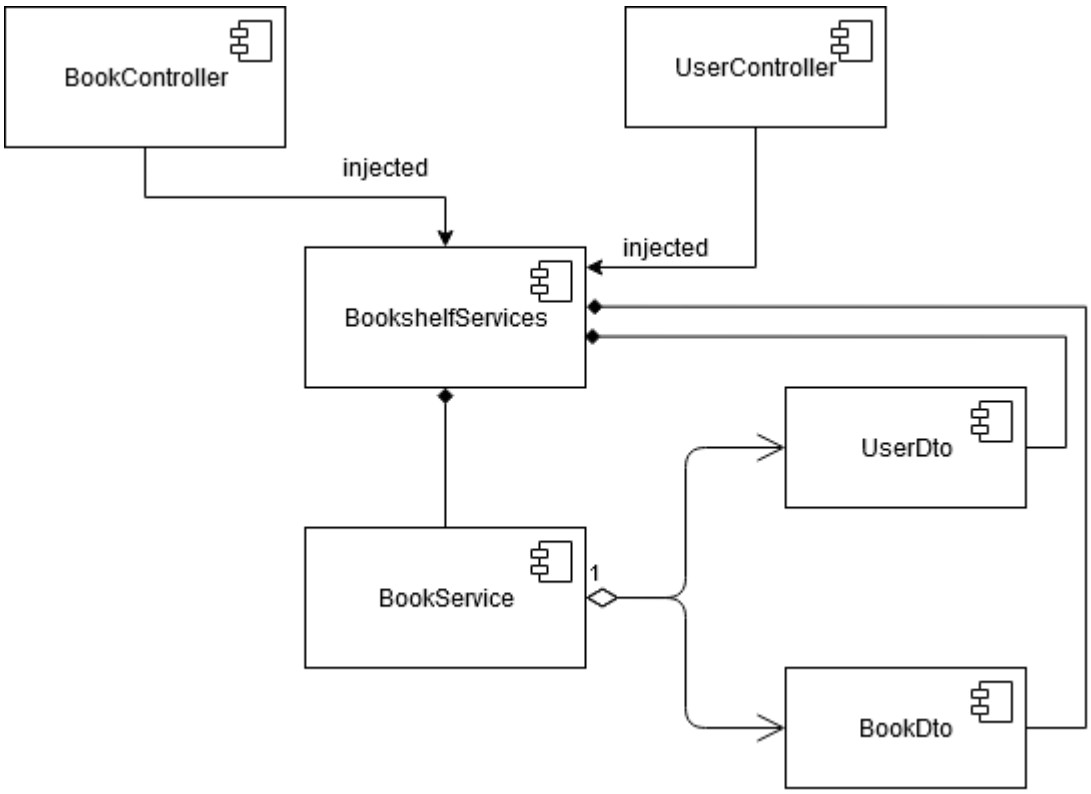

Рис. 3.4. Схема компонентів веб-API

Отже, на рисунку 3.4 показані основні компоненти веб-API. Ін’єкція означає, що дані про об’єкт класу беруться зі сховища DependebcyInjection. Це значно спрощує створення об'єктів і вирішує проблему цілісності системи як такої.

BookshelfService - це компонент з’єднувач в сучасній архітектурі. Це шар між контролерами та службами і визначає, яка служба відповідає за який контролер. Це рішення спрощує роботу контролера в цілому, йому не потрібно знати, до якої служби йому потрібно звернутися, воно просто має набір вхідних і вихідних значень, які обробляються в BookshelfService, який потім відправляє їх до відповідної служби.

BookDto та UserDto - це моделі, які відповідають таблицям з бази даних.

2) клієнт Windows

Основою для клієнта Windows є WPF. Технологія WPF є частиною екосистеми платформи .NET і є підсистемою для побудови графічних інтерфейсів.

Шаблон MVVM дозволяє відокремити логіку програми від візуальної частини (подання). Цей шаблон є архітектурним, тобто визначає загальну архітектуру програми.

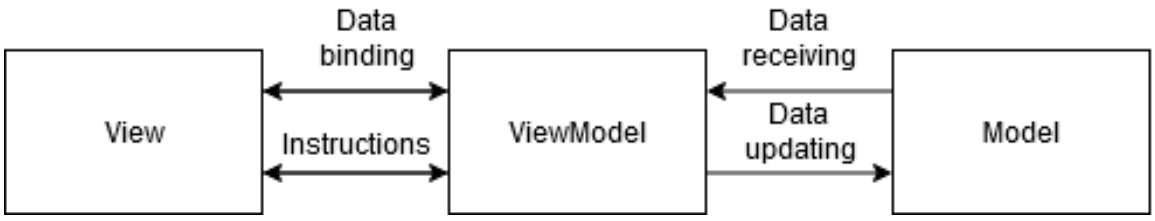

Рис. 3.5. MVVM Pattern

Як показано на рис. 3.5, MVVM складається з трьох компонентів: Models, Models of View (ViewModel) and Views (View). [7]

Модель описує дані, що використовуються в додатку. Моделі можуть містити логіку, безпосередньо пов’язану з цими даними, наприклад, логіку перевірки властивостей моделі. У той же час модель не повинна містити жодної логіки, пов'язаної з відображенням даних та взаємодією з візуальними елементами управління.

Часто модель реалізує інтерфейси INotifyPropertyChanged або INotifyCollectionChanged, які дозволяють повідомляти систему про зміни у властивостях моделі. Це полегшує прив’язку до подання, хоча знову ж немає прямої взаємодії між моделлю та видом.

Перегляд визначає інтерфейс, за допомогою якого користувач може взаємодіяти з додатком. Для WPF використовує форми xaml як основу для користувацького інтерфейсу, який визначається в таких візуальних елементах, як кнопки, перегляд тексту та інші.

Хоча вікно (клас Window) у WPF може містити як інтерфейс у xaml, так і прив'язаний до нього код C #, код C # не повинен містити жодної логіки, крім конструктора, який викликає метод InitializeComponent і ініціалізує вікно. Вся основна логіка програми зберігається в компоненті ViewModel.

ViewModel пов'язує модель і переглядає механізм прив'язки даних. також містить логіку для отримання даних із моделі, які потім передаються у подання. I, VewModel визначає логіку оновлення даних у моделі.

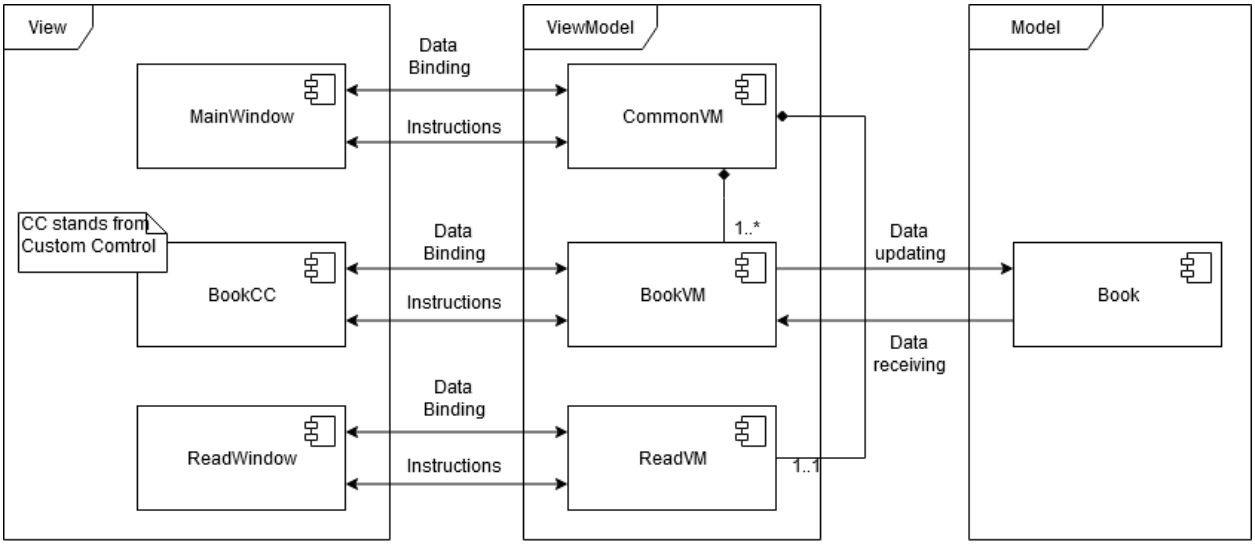

Рис. 3.6. Структура WPF застосунку

Таким чином, якщо взяти структуру програми, її можна легко розділити на три групи (як показано на рис. 3.6).

Перша група відповідає лише за користувальницький інтерфейс та зв’язок з користувачем. Усі елементи подання є файлами .xaml і мають розмітку з елементами.

Наприклад, нижче розміщена розмітка xaml для додавання елементів керування пошуком на головну сторінку (рис. 3.7):

```
<DockPanel Grid.Row="2" Width="200"
                    HorizontalAlignment="Right"
                    Background="Lavender">
            <Button BorderThickness="0" DockPanel.Dock="Left"
                    HorizontalAlignment="Right"
                    Height="30" Width="30"
                    Background="Transparent">
                <Image Source="F:\NAU\Diploma\EBookrReader\EBookrReader\Recources\search.png"/>
            </Button>
            <TextBox BorderThickness="0"
                     FontSize="16"
                     HorizontalAlignment="Stretch"
                     Background="Transparent"
                     VerticalContentAlignment="Center"/>
        </DockPanel>
```

Рис. 3.7. Приклад розмітки Xaml

Друга група відповідає за взаємозв'язокModel та View. Відмінною рисою є загальний абстрактний клас ViewModelBase, який успадковує інтерфейс INotifyPropertyChanged, щоб інформувати View про зміни даних моделі.

І остання група - це група моделей. Суть цієї групи полягає в тому, що всі класи є класами даних, тобто клас Book має властивості імені, автора, дати завантаження тощо, але не має жодних маніпуляцій з даними.

Таким чином, можна сказати, що у програмі WPF неухильно дотримується структура MVVM.

3) Додаток для Android

Структура програми для Android є більш складною, ніж структура WPF. Він використовує такі компоненти, як Dagger2, модернізований клієнт Http та контейнер Fragment.

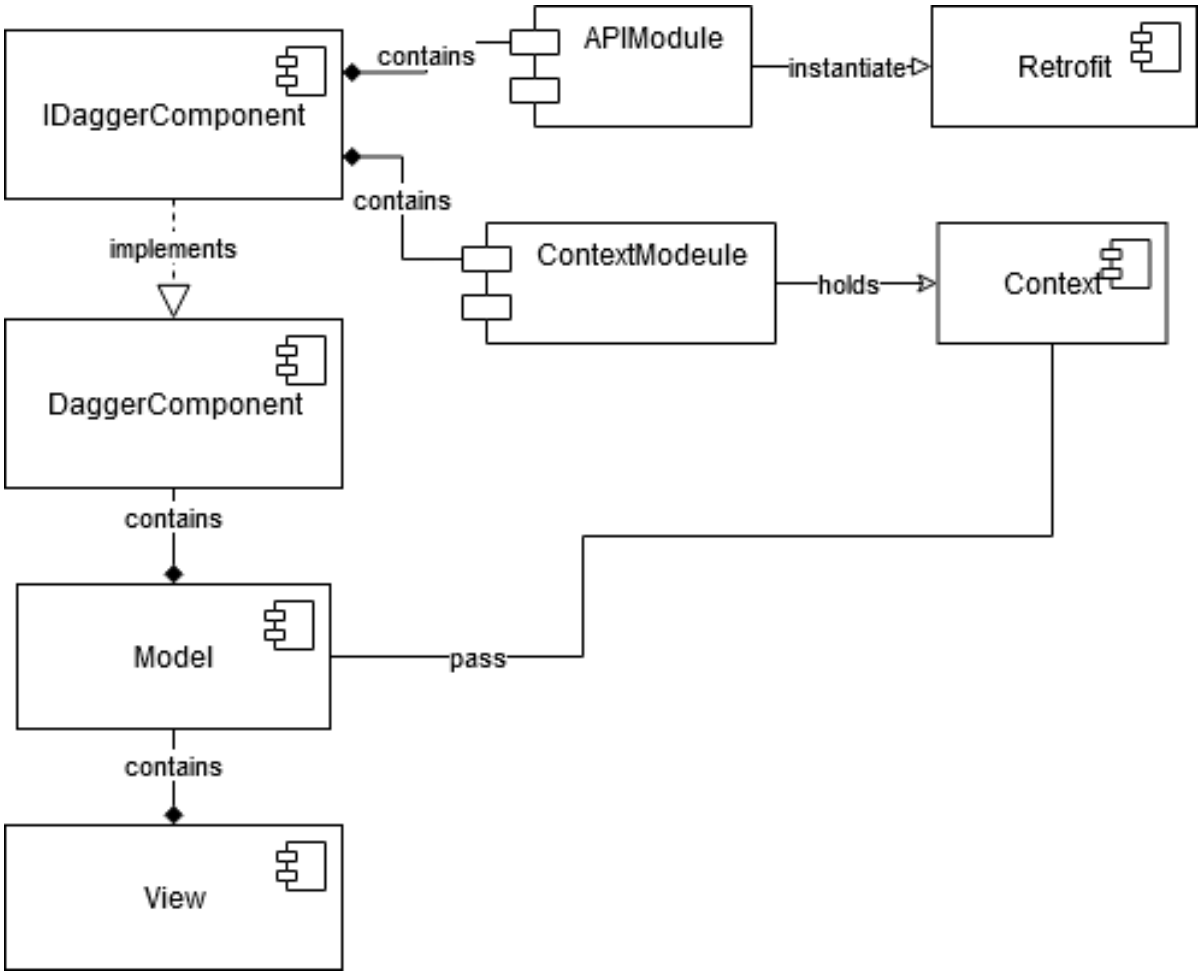

Рис. 3.8. Структура Android застосунку

На рис. 3.8 вище елементом з’єднання для цього додатка для Android є IDaggerComponent, який реалізований у DaggerComponent, який, у свою чергу, пов’язаний із шарами Model і View.

Dagger 2 - це бібліотека, яка допомагає розробнику реалізувати шаблон ін'єкції залежності, що, в свою чергу, є "специфічною формою інверсії контролю".

Процес забезпечення зовнішньої залежності програмного компонента. Це специфічна форма інверсії контролю (IoC), коли вона застосовується до управління залежностями. У повній відповідності з принципом єдиного зобов'язання об'єкт дбає про побудову залежностей, які йому потрібні, до зовнішнього, спеціально розробленого для цього загального механізму.

Отже, Dagger 2 просто подбає про створення цього загального механізму.

Модернізація - це REST-клієнт для Java та Android. Це полегшує отримання та завантаження JSON через веб-службу, що базується на REST. Модернізація використовує бібліотеку OkHttp для запитів HTTP. Отже, це корисний пакет для побудови запитів на сервері.

Контекстний модуль необхідний для DI, він відповідає за зберігання та визначення відповідних введених блоків.

Ще один важливий момент пов'язаний з інтерфейсом користувача. Для того, щоб зрозуміти це, ви повинні спочатку зрозуміти, що таке Діяльність та її життєвий цикл.

Діяльність є ключовим компонентом для створення візуального інтерфейсу. Діяльність надає вікно, в якому програма малює свій інтерфейс. Це вікно зазвичай заповнює екран, але може бути меншим за екран і розміщуватися поверх інших вікон. Як правило, одна діяльність реалізує один екран у програмі. [8]

Діаграма станів – діаграма, що визначає зміну станів об'єкта у часі, одна з діаграм моделювання поведінки в UML. Діаграма станів є графом спеціального виду, який представляє певний автомат. Вершинами графа є

можливі стани автомата, зображувані відповідними графічними символами, а дуги позначають його переходи зі стану в стан. Діаграми станів можуть бути вкладені одна в одну для більш детального представлення окремих елементів моделі.

Діаграми станів застосовуються для того, щоб пояснити, яким чином працюють складні об'єкти. Діаграма станів показує, як об'єкт переходить з одного стану в інший. Очевидно, що діаграми станів служать для моделювання динамічних аспектів системи.

Діаграма станів корисна при моделюванні життєвого циклу об'єкта. Від інших діаграм діаграма станів відрізняється тим, що описує процес зміни станів тільки одного примірника певного класу - одного об'єкта, причому об'єкта реактивного, тобто об'єкта, поведінка якого характеризується його реакцією на зовнішні події. Поняття життєвого циклу вдаються якраз до реактивних об'єктів, даний стан яких обумовлено їх минулим станом. Але діаграми станів важливі не тільки для опису динаміки окремого об'єкта. Вони можуть використовуватися для конструювання виконуваних систем шляхом прямого і зворотного проектування.

Діаграми станів найчастіше використовуються для опису поведінки окремих об'єктів, але також можуть бути застосовані для специфікації функціональності інших компонентів моделей, таких як варіанти використання, актори, підсистеми, операції та методи.

Діяльність має життєвий цикл - початок, коли Android створює екземпляр активності, проміжний стан, і кінець, коли екземпляр руйнується системою та звільняє ресурси. Діяльність може бути у трьох станах:

- активний (активний або запущений) - активність знаходиться на передньому плані екрану. Користувач може взаємодіяти з активним вікном;

- призупинено - діяльність втратила фокус, але користувач все ще бачить. Тобто діяльність знаходиться зверху і частково перекриває цю діяльність. Призупинена діяльність може бути знищена системою в критичних ситуаціях із нестачею пам'яті;

- зупинено - якщо ця діяльність повністю закрита іншою діяльністю. Це більше не видно користувачеві і може бути знищено системою, якщо для більш важливого процесу потрібна пам’ять.

Якщо діяльність, яка була знищена системою, потрібно знову показати на екрані, її слід повністю перезапустити та відновити до попереднього стану.

Однак відновити попередній стан активності в сучасних реаліях є надзвичайно нетривіальним завданням, і тому зазвичай після закриття діяльності разом із ним знищуються всі незбережені дані.

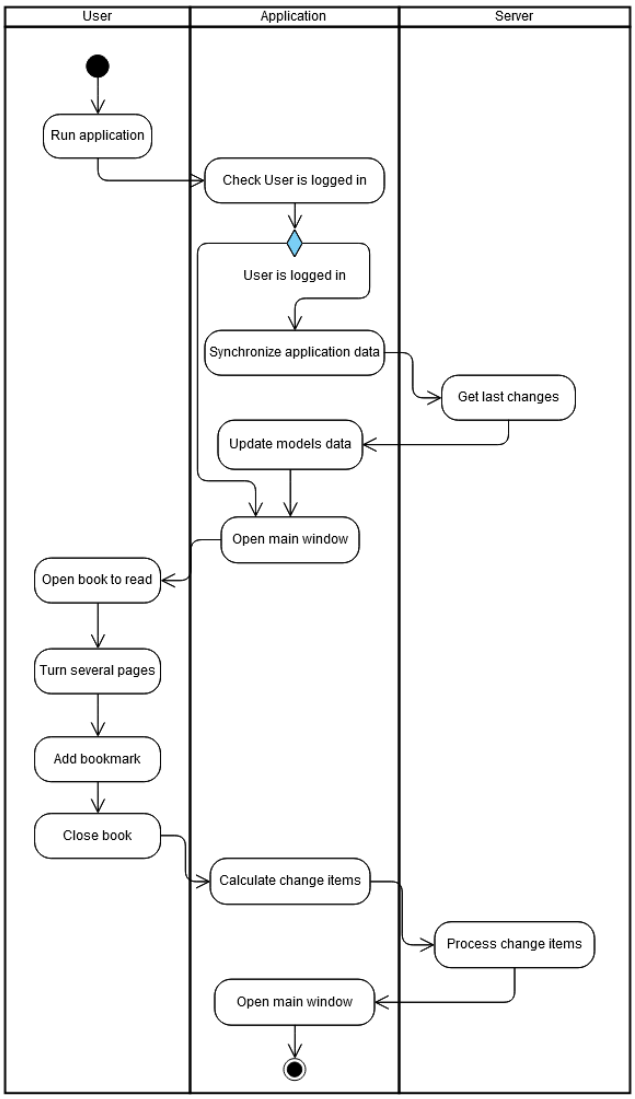

Рис. 10. Діаграма станів засобу

Щоб вирішити цю проблему, вони зараз вдаються до використання однієї діяльності для всієї програми, а фрагменти використовуються для розділення вікон між собою.

Таким чином, у цьому кроці було використано одну діяльність, а в ній три фрагменти: основна вкладка, вкладка входу / реєстрації та вкладка читання.

Висновки по розділу

В даному розділі описане проектування мобільного додатку для читання книг іноземною мовою з можливістю перекладу невідомих слів, що включає в себе розробку архітектури, діаграми компонентів, діаграми послідовності виконання тощо.

РОЗДІЛ 4

ПРОТОТИП ЗАСОБУ

4.1 Керівництво користувача

Android

Можливо, варто почати з мобільного додатка. Клієнт встановлюється на пристрій через файл .apk.

Після встановлення та запуску програми перший користувач відкриває головне вікно “Книжкова полиця” (рис. 4.1):

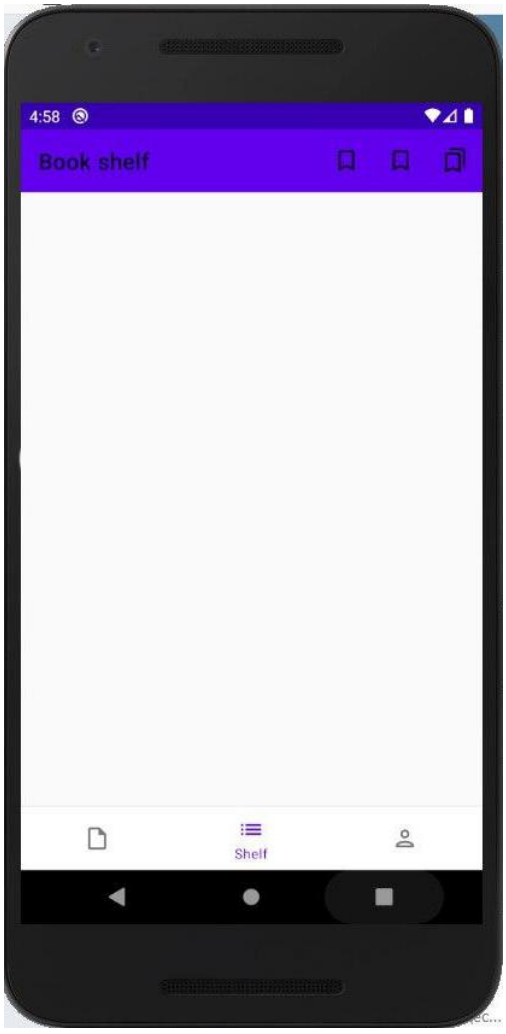

Рис. 4.1. Головне вікно програми

Потім користувач переходить на вкладку Профіль, де йому пропонується зареєструватися або увійти до свого облікового запису (рис. 4.2). Тепер система реєстрації дуже проста, вона включає лише ім’я користувача та

пароль, і залежно від кнопки, яку натискає користувач, відбудеться дія. Логін для кожного користувача повинен бути унікальним, і якщо користувач намагається зареєструвати новий обліковий запис із наявним логіном, програма покаже помилку та запропонує вибрати новий логін.

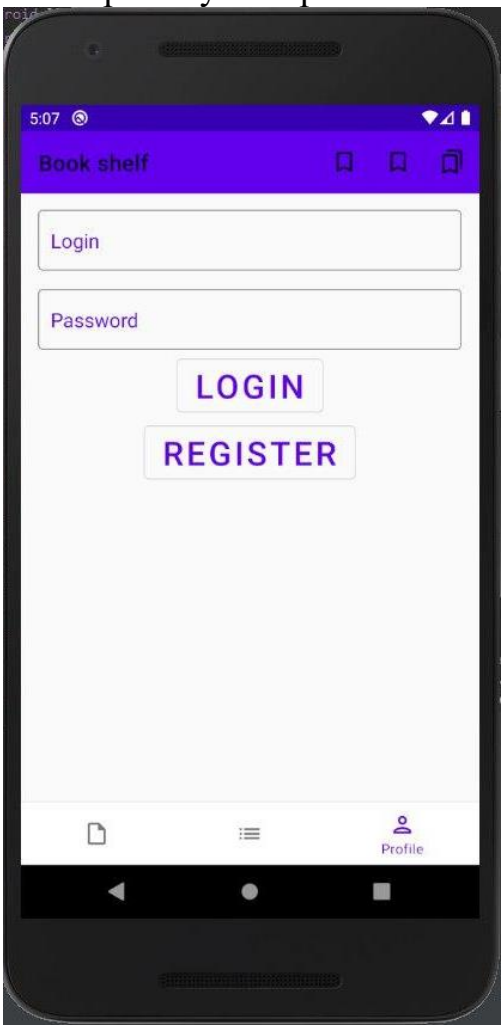

Рис. 4.2. Вікно входу в програму

Після того, як користувач зареєструвався та увійшов до програми, він знову відкриває вкладку Shelf (Полиця) та додає дві книги (рис. 4.2). Ці книги негайно зберігаються в базі даних для подальшого маніпулювання ними.

Далі користувач відкриває одну з книг. Зміст книги розділено на сторінки (рис. 4.3). Номер сторінки можна натиснути, тому користувач може переходити з однієї сторінки на іншу, не монотонно гортаючи. Ця функція особливо актуальна для документів великого обсягу.

А панель інструментів зверху має функцію закладки. Перший елемент - це додавання закладки, другий - видалення закладки, а останній відображатиме всі закладки в цій книзі.

Рамки з боків показують межі перегортання сторінок.

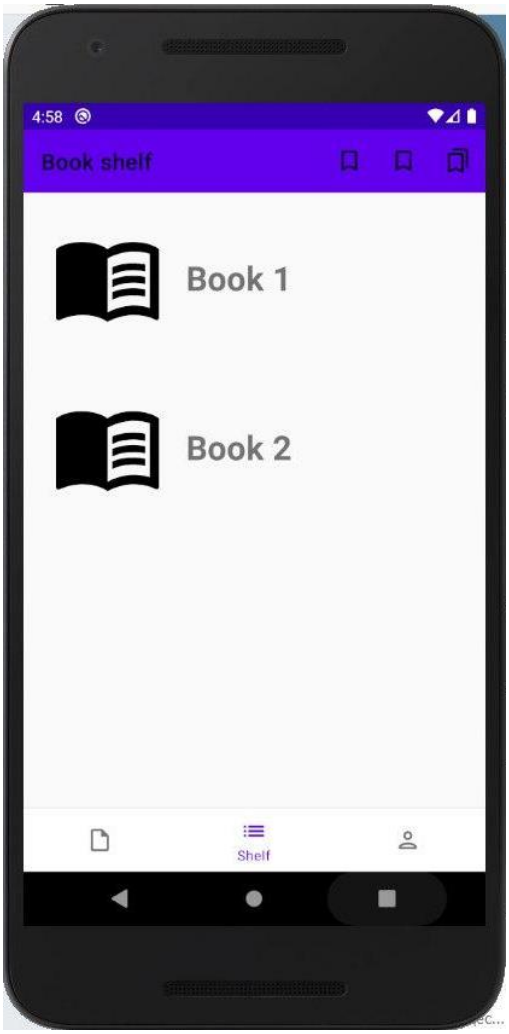

Рис. 4.2 Bookshelf з книгами

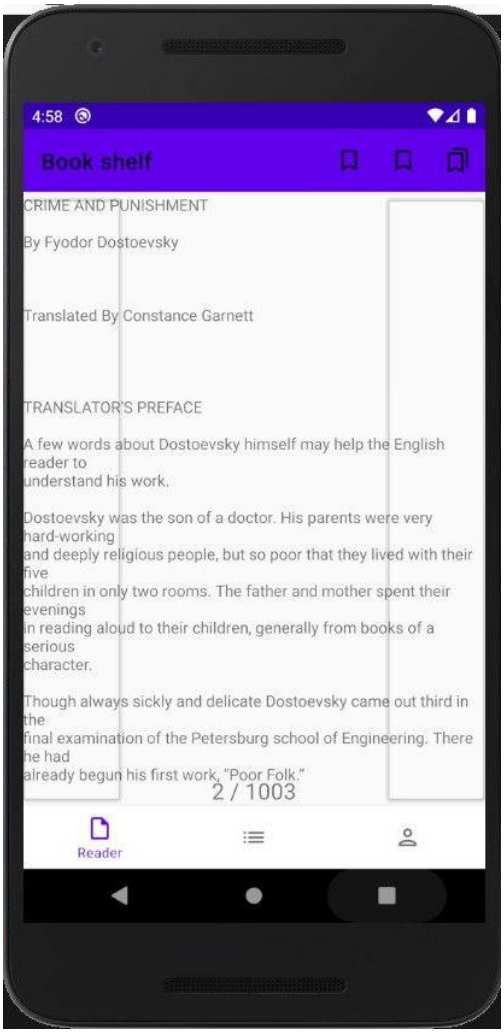

Рис. 4.3 Вкладка зчитувача

Windows

Як і мобільний додаток, настільний додаток також починається з головного вікна (рис. 4.4), яке складається з бокового меню, де користувач може ввійти, і вікна з книгами. Угорі є пошук, за допомогою якого користувач може швидко знайти потрібний йому документ. Крім того, у правому нижньому куті є кнопка для додавання книг до бібліотеки. Після того, як користувач запустив програму, він може або негайно почати використовувати

зчитувач за призначенням, або спочатку увійти в систему. Як уже згадувалося вище, авторизація дозволяє синхронізувати клієнтів на різних пристроях.

Рис. 4.4. Windows за стосунок головного вікна

Після того, як користувач натисне кнопку Вхід, з’явиться модальне вікно з полями для введення облікових даних (рис. 4.5).

Рис. 4.5 Windows за стосунок вікна входу

І коли користувач вводить інформацію про обліковий запис, яку він використовував у мобільному додатку, і натискає кнопку Вхід, програма синхронізує дані з даними, які зберігаються для цього облікового запису в базі даних. Таким чином, ті книги, які він додав у клієнт по телефону, тепер доступні на комп'ютері (рис. 4.6-4.7).

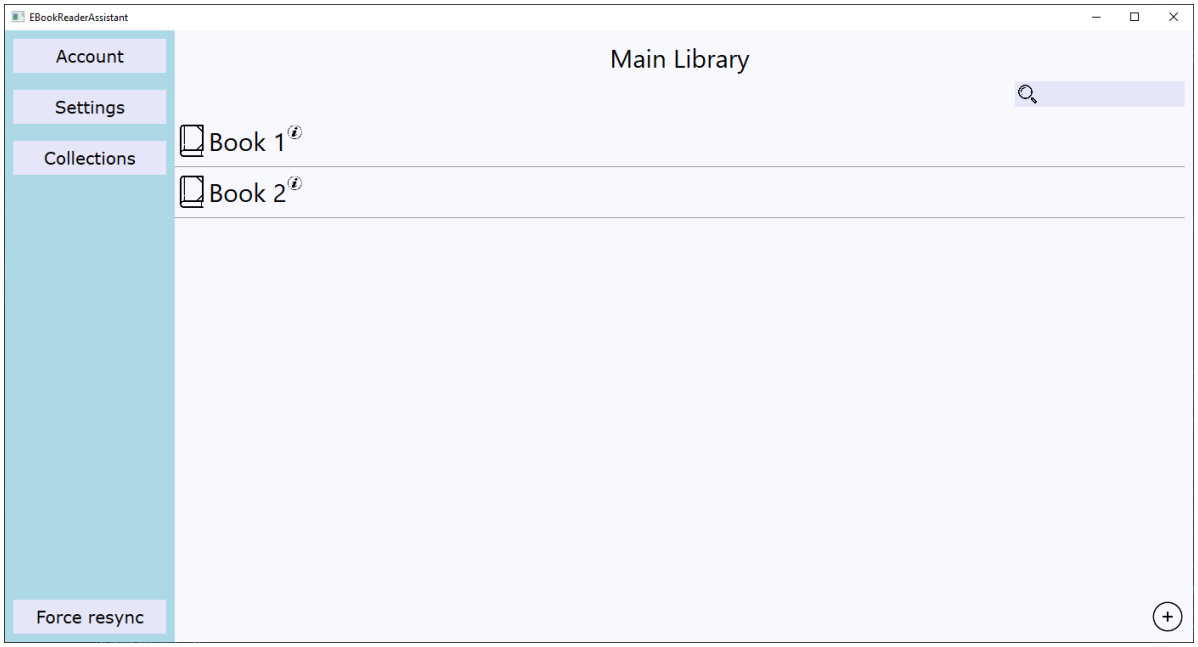

Рис. 4.6 Windows застосунок входу користувача

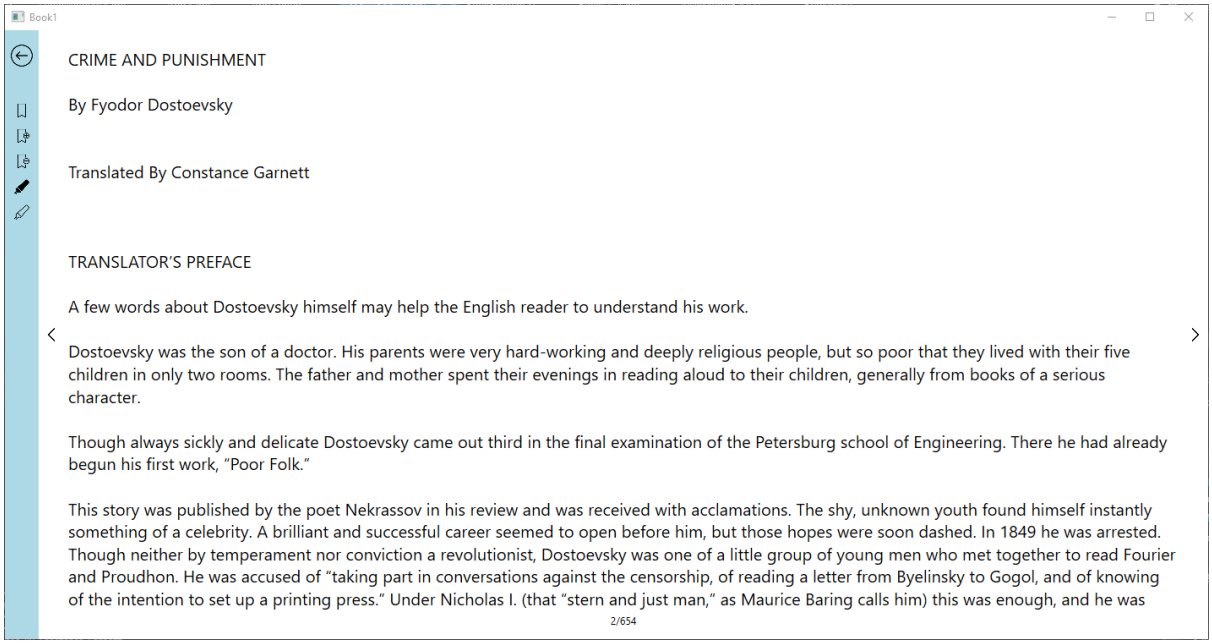

Рис. 4.7 Windows застосунок відкриття книги

Після того, як користувач увійшов в обліковий запис, кнопка «Вхід» змінилася на «Обліковий запис», що означає, що тепер цей елемент керування приведе користувача до його особистого кабінету, де зберігається інформація про його ім'я та завантажені книги.

Як і в мобільній версії, у вікні для читання книги є панель інструментів. Ця панель розташована ліворуч і містить такі елементи, як додавання, видалення та перегляд усіх закладок, а також маркер для виділення тексту.

Кнопка Force Resync оновлює дані програми даними з бази даних. Наявність цієї кнопки пояснюється тим, що на даний момент програма отримує дані з бази даних лише в той момент, коли користувач запускає програму. Отже, якщо додаток працював протягом певного часу, і дані в базі даних якимось чином поповнювались діями на іншому клієнті, має сенс використовувати цю кнопку замість перезапуску програми.

Висновки по розділу

В даному розділі було описано основні функції мобільних та настільних клієнтів. Якщо порівняти з вимогами, сформульованими у другому розділі, на етапі проектування цього програмного забезпечення можна побачити, що більшість із них виконані. Також були названі основні елементи інтерфейсу та їх функції. Сам інтерфейс досить інтуїтивно зрозумілий та зручний.

ВИСНОВКИ

Незважаючи на різке зростання популярності електронних документів, їх актуальність тим часом лише посилюється. Однак надзвичайно важко знайти програмне забезпечення, придатне для всіх потреб, щоб задовольнити всі потреби, незважаючи на те, що сучасний ринок сповнений всілякими електронними зчитувачами.

Отже, в ході дипломного проектування був спроектований та розроблений програмний продукт, який реалізує та охоплює майже всі необхідні функціональні можливості для читання та перекладу електронних документів.

Етап проектування включав встановлення функціональних вимог та реалізацію основних діаграм, які відносяться до етапу проектування.

Для таких програмних засобів найбільш підходящою архітектурою є архітектура клієнт-сервер, оскільки ця архітектура є з'єднанням між клієнтом і сервером, незалежно від того, який це клієнт і скільки їх існує, сервер може обробляти запити від кількох клієнтів одночасно .

Останнім етапом була безпосередньо розробка програмного продукту. Цей етап був найдовшим за часом і також був розділений на три підетапи: розробка серверної частини, розробка десктопної версії та розробка мобільного Android додатку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Voronctsov A.V. Reading as a social problem // Universum: Herald of the University of Herzen – 2009. – 159 p.
2. E-Book Reading Statistics [Electronic resource]. – Access Mode: https://sites.google.com/site/tabletspcru/statistika-ctenia-elektronnyh-knig
3. IIEEE Standard Glossary of Software Engineering Terminology, 1990.
4. What are UML Diagrams? [Electronic resource]. – Access Mode: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml-diagrams/
5. Client-server Model Architecture. Client-server model diagram. [Electronic resource]. – Access Mode: https://en.wikipedia.org/wiki/Client%E2%80%93server_model
6. Welcome to the Visual Studio IDE. What is Visual Studio[Electronic resource]. – Access Mode: https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019
7. MVVM Pattern. MVVM Pattern Description [Electronic resource]. – Access Mode: https://metanit.com/sharp/wpf/22.1.php
8. Introduction to Android development [Electronic resource]. – Access Mode: https://developer.android.com/guide/development/intro-activities