Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти перший (бакалаврський)

Мобільний застосунок для FinTech-галузі

(тема)

Виконав:
здобувач 4 року навчання,
групи КІУКІ-21-4
Еліна КУЗЬМІНА
(власне ім'я, прізвище)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Комп'ютерна інженерія
(повна назва освітньої програми)

Керівник: доц. Наталія БОЛОГОВА
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти перший (бакалаврський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Кузьміній Еліні Олександровні
(прізвище, ім’я, по батькові)

1. Тема роботи Мобільний застосунок для FinTech- галузі

затверджена наказом по університету від “ 26 ” травня 2025 р. № 424 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 17 червня 2025 р.

3. Вхідні дані до роботи 1) мова програмування: Java; 2) середовище розробки: Android .

4. Перелік питань, що потрібно опрацювати у роботі

1) аналіз проблеми та огляд існуючих рішень;

2) аналіз існуючих технологій FinTech ;

3) аналіз вимог до створення застосунку;

4) розробка рішення реалізації застосунку;

5) програмна реалізація застосунку;

6) висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій 13 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
№	Назва етапів роботи	Строк / терміни	
1	Аналіз проблеми та огляд існуючих рішень	27.05.25-30.05.25	
2	Аналіз програмних засобів FinTech	31.05.25-03.06.25	
3	Розробка рішення реалізації застосунку	04.06.25-07.06.25	
4	Програмна реалізація застосунку	08.06.25-09.06.25	
5	Оформлення матеріалів кваліфікаційної	10.06.25-11.06.25	
	роботи		
6	Подання кваліфікаційної роботи керівникові та	12.06.25-13.06.25	
	її попередній захист		
7	Подання кваліфікаційної роботи на рецензу-	14.06.25-16.06.25	

Дата видачі завдання " 26 " травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Наталія БОЛОГОВА
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 52 с., 20 рис., 1 табл., 1 дод., 13 джерел.

МОБІЛЬНИЙ ЗАСТОСУНОК, JAVA, API, ІНТЕРФЕЙС, ANDROID, GUI, ПРОГРАМНА РЕАЛІЗАЦІЯ.

Метою кваліфікаційної роботи є розробка мобільного застосунку для фінансової галузі (FinTech), який дозволяє користувачам здійснювати онлайн-платежі, використовуючи сучасні технології Android-розробки.

У ході виконання кваліфікаційної робот були такі завдання:

- дослідити поточний стан і перспективи розвитку FinTech-галузі в Україні;
- визначити особливості впливу пандемії COVID-19 та повномасштабного вторгнення на розвиток цифрових фінансових сервісів;
- проаналізувати сучасні технології розробки мобільних додатків;
- реалізувати програмний продукт з використанням Kotlin, Firebase та архітектури MVP;
- протестувати створений застосунок та оцінити його функціональність.

Процеси та технології розробки програмного забезпечення для мобільних платформ у сфері FinTech.

Методи та інструменти проєктування й реалізації мобільного застосунку для здійснення онлайн-фінансових операцій.

ABSTRACT

Bachelor’s thesis: 52 pages, 20 figures, 1 tables, 1 appendices, 13 sources.

MOBILE APPLICATION, JAVA, API, INTERFACE, ANDROID, GUI, SOFTWARE IMPLEMENTATION.

The major goal of this thesis is the qualification work is to develop a mobile application for the financial industry (FinTech) that allows users to make online payments using modern Android development technologies.

In order to the qualification work, the following tasks were performed:

- to study the current state and prospects of the FinTech industry in Ukraine;
- to identify the impact of the COVID-19 pandemic and full-scale invasion on the development of digital financial services;
- to analyze modern mobile application development technologies;
- to implement a software product using Kotlin, Firebase, and MVP architecture;
- to test the created application and evaluate its functionality.

Processes and technologies for developing software for mobile platforms in the FinTech sector.

Methods and tools for designing and implementing a mobile application for online financial transactions.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

МЗ – мобільний застосунок

ОС – операційна система

ПЗ – програмне забезпечення

Agile – гнучка методологія розробки (англ., Agile Software Development)

API – набір правил, протоколів та інструментів (англ., Application Programming Interface)

BDD – процес створення (англ., Behavior-Driven Development)

DSL –предметно-орієнтована мова програмування (англ. Domain-specific language)

ВСТУП

У сучасному світі стрімкий розвиток цифрових технологій кардинально змінив способи взаємодії між бізнесом і споживачами. Однією з ключових сфер таких трансформацій є фінансова галузь, де на перший план виходять FinTech-рішення. Мобільні застосунки стали невід'ємною частиною повсякденного життя користувачів, забезпечуючи їм швидкий доступ до послуг, що раніше вимагали особистої присутності або значних зусиль. Особливо це стало помітно під час пандемії COVID-19 та повномасштабного вторгнення, коли цифрові сервіси виявилися єдиним стабільним каналом комунікації, роботи та навчання.

Розвиток FinTech-сервісів в Україні відкриває нові можливості для створення інноваційних мобільних рішень у сфері фінансових послуг. Враховуючи актуальні потреби ринку, розробка зручного, доступного та безпечного мобільного застосунку для здійснення онлайн-платежів є надзвичайно важливою і затребуваною задачею.

Метою кваліфікаційної роботи є розробка мобільного застосунку для фінансової галузі (FinTech), який дозволяє користувачам здійснювати онлайн-платежі, використовуючи сучасні технології Android-розробки.

Для досягнення поставленої мети необхідно виконати такі завдання:

- дослідити поточний стан і перспективи розвитку FinTech-галузі в Україні;
- визначити особливості впливу пандемії COVID-19 та повномасштабного вторгнення на розвиток цифрових фінансових сервісів;
- проаналізувати сучасні технології розробки мобільних додатків;
- реалізувати програмний продукт з використанням Java, Firebase та архітектури MVP;
- протестувати створений застосунок та оцінити його функціональність.

Процеси та технології розробки програмного забезпечення для мобільних платформ у сфері FinTech.

Методи та інструменти проєктування й реалізації мобільного застосунку для здійснення онлайн-фінансових операцій.

У процесі дослідження використовувалися: аналітичний метод (для вивчення FinTech-ринку), системний аналіз (для проектування структури додатку), методи об’єктно-орієнтованого програмування, методи модульного тестування, а також емпіричні методи дослідження під час тестування розробленого продукту, та полягає у практичній реалізації універсального мобільного застосунку, що забезпечує функціональність онлайн-гаманця з використанням сучасних технологій та платформ, адаптованих під вимоги українського FinTech-ринку.

Результати дослідження можуть бути використані для подальшого вдосконалення цифрових сервісів у сфері фінансів, а також як приклад розробки мобільного застосунку у навчальному процесі або для запуску стартапу в галузі FinTech.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз ринку

FinTech (Financial Technologies) є сучасними технологіями, які допомагають фінансовим установам та різним компаніям в управлінні фінансами бізнесу. FinTech включає: програми, програмні забезпечення та бізнес-моделі. Також FinTech є галуззю, в якій компанії пускають у хід різні сучасні фінансові технології для конкуренції із традиційними фінансовими установами. У більшості випадків це або техностартапи, або компанії, які використовують FinTech-інструменти для покращення своїх послуг.

Сьогодні надзвичайно важливо вміти ефективно використовувати постійно змінний ландшафт FinTechy на користь свого бізнесу. Нові лідери ринку формують стратегії для сталого розвитку і замислюються над тим, як FinTech може впливати – і вже впливає – на їхні організації. Водночас створюються і впроваджуються як інноваційні, так і перевірені технології. Нові продукти відкривають нові можливості, замінюючи застарілі рішення.

Простими словами Fintech є електронною платіжною системою, яка дозволяє клієнту банку або фінансової установи здійснювати фінансові або нефінансові транзакції онлайн через Інтернет. Ця послуга надає онлайн-доступ практично всім банківським послугам, традиційно доступним через місцеве відділення, включаючи грошові перекази, депозити та онлайн-платежі за рахунками клієнтам.

Користуватися FinTech програмами можуть як фізичні особи з активним банківським рахунком, так і будь-які фінансові установи, просто зареєструвавшись. Після цього клієнту для користування банківськими послугами не треба буде щоразу ходити до банку. Даний спосіб надання банківських послуг є не лише зручним, але й безпечним. Дані клієнтів банку в мережевих порталах захищені паролями та унікальними ідентифікаторами.

Мобільний застосунок, створений у межах цієї кваліфікаційної роботи, не орієнтований на конкретну цільову аудиторію – він розроблений як універсальний інструмент, яким може скористатися будь-хто. Основна мета полягає в тому, щоб забезпечити зручну альтернативу традиційним способам проведення фінансових операцій. Програма виконує функції онлайн-гаманця, подібно до сервісу Portmone: кожен користувач має можливість зареєструватися, створити обліковий запис, отримати доступ до віртуальної банківської картки та виконувати стандартні операції – перекази, оплати, поповнення рахунку тощо.

Таке рішення дозволяє спростити взаємодію з фінансовими послугами, особливо для тих користувачів, які прагнуть цифрової зручності й безпеки. Завдяки інтуїтивному інтерфейсу, сучасній архітектурі MVP та використанню хмарної платформи Firebase, додаток поєднує в собі простоту використання й функціональну гнучкість.

1.2 Аналіз українського ринку

Нині FinTech-ринок слід розглядати з одного боку як екосистему, що об'єднує всіх учасників фінансового ринку, включаючи FinTech-стартапи, регулятори, банки, міжнародні платіжні системи, асоціації банкірів та фінансистів, та постачальників, а з іншого боку – як складну систему, яка об'єднує сектори нових технологій та фінансових послуг, стартапів та пов'язану з ними інфраструктуру, яка наведена на рисунку 1.1.

Рисунок 1.1 – Учасники FinTech та їх взаємодія

Ключові показники за 2024 рік. У 2024 році в Україні функціонує понад 250 FinTech-компаній, які працюють у таких напрямках, як мобільний банкінг, цифрові платежі, онлайн-кредитування та блокчейн-рішення. За рік кількість учасників ринку збільшилась на 20%, що підтверджує стабільне зростання галузі та підвищення попиту на цифрові фінансові послуги.

Капіталізація FinTech-сектора в Україні у 2024 році зросла на 18%, завдяки залученню міжнародних інвестицій та співпраці з європейськими партнерами. Розширення ринку підтримується державними ініціативами, спрямованими на інновації, а також законодавчими змінами для прозорості та безпеки діяльності FinTech-компаній [1].

Технологічні інновації та законодавчі зміни. Українські FinTech-компанії у 2024 році впроваджують штучний інтелект для підвищення ефективності обслуговування клієнтів та покращення аналізу ризиків. Крім того, блокчейн-технології використовуються для збереження даних та забезпечення прозорості фінансових операцій. Законодавчі зміни в Україні цього року також підтримують впровадження цифрових валют і токенів, що сприяє зростанню ринку криптовалют [1].

Основні виклики та перспективи розвитку. Основними викликами для галузі залишаються посилення кібербезпеки та дотримання міжнародних стандартів конфіденційності даних. Українським FinTech-компаніям важливо відповідати вимогам міжнародного ринку, щоб продовжувати залучати іноземних інвесторів та розширюватися за кордон. Позитивна динаміка 2024 року свідчить про готовність FinTech-ринку України до подальших інновацій та стабільного розвитку.

Аналіз тенденцій. Зростання кількості компаній: З 2021 по 2024 рік кількість FinTech-компаній в Україні зросла майже вдвічі, що свідчить про активний розвиток галузі.

Міжнародна експансія: Хоча у 2023 році спостерігалося зниження частки компаній, що працюють на міжнародному ринку, до 33%, у 2024 році цей показник зріс до 47%, що вказує на відновлення та розширення міжнародної присутності українських FinTech-компаній [1].

Фінансова стабільність: Зростання частки компаній, що досягли беззбитковості (з 68% у 2023 році до 75% у 2024 році), та збільшення кількості самофінансованих компаній (з 66% до 79%) свідчить про зміцнення фінансової стійкості галузі.

Географічна концентрація: Більшість FinTech-компаній зосереджені в Києві (77% у 2023 році та 79% у 2024 році), що підкреслює роль столиці як центру фінансових інновацій.

Гендерна різноманітність та участь у ЗСУ: Хоча частка компаній, заснованих жінками, залишалася відносно стабільною (17% у 2023 році та 16% у 2024 році), варто відзначити значну участь представників FinTech-компаній у Збройних Силах України (55% у 2023 році та 60% у 2024 році), що свідчить про соціальну відповідальність галузі [1].

Оцінка ринку: У 2024 році загальна оцінка українського FinTech-ринку сягнула $1,2 млрд, що підкреслює його економічну значущість та інвестиційну привабливість, дані представлено в таблиці 1.1.

Таблиця 1.1 – Динаміка українського FinTech-ринку (2021–2024)

Рік	Кількість компаній	Частка на міжнародному ринку	Досягнення беззбитковості	Представники ЗСУ	Оцінка ринку ($)
2021	130+	~50%	~70%	–	–
2022	203	49%	–	–	–
2023	246	33%	68%	55%	–
2024	256	47%	75%	60%	$1,2 млрд

Українська асоціація FinTech та інноваційних компаній представила щорічне дослідження найактуальніших FinTech трендів, проаналізувавши звіти відомих аналітичних компаній світу та кейси провідних компаній. FinTech Insider вибрав основне з дослідження [2].

На думку опитаних в ході дослідження експертів, штучний інтелект, відкритий банкінг та кібербезпека є найбільш перспективними напрямками розвитку фінансово-технологічного сектору в Україні в найближчі два роки. Також у ТОП-5 входять діджитал банкінг, який замінив цифрове кредитування у порівнянні з попереднім роком, і військові технології (рисунок 1.2).

Рисунок 1.2 – Найбільш перспективні напрямки розвитку фінансовотехнологічного сектору в Україні у найближчі два роки

Найбільш перспективні напрямки розвитку фінансовотехнологічного сектору в Україні у найближчі два роки.

На думку експертів, найбільший вплив на фінансовий сектор у наступному році матимуть штучний інтелект (71% опитаних), відкритий банкінг (62%), KYC/ALM (34%) та e-commerce (34%) [2].

Як і минулого року, війна є основним бар'єром на шляху подальшого розвитку FinTechy в Україні – таку відповідь дали 67% опитаних експертів. Також дуже актуальним питанням є відтік кадрів за кордон (62% опитаних). Законодавчі обмеження та політична/економічна нестабільність як бар'єри на шляху розвитку FinTechy зазначили по 48% опитаних експертів, а брак фінансування – 38%.

Щодо того, що може дати поштовх розвитку FinTech-сектору в Україні, то тут опитані назвали закінчення війни та запуск екосистем з використанням Open API та відкритого банкінгу [3].

1.3 Обґрунтування вибору розробки

У сучасному цифровому суспільстві роль фінансових технологій (FinTech) стрімко зростає, набуваючи особливої ваги у період після пандемії COVID-19. Цей період став каталізатором масової цифровізації, і FinTech-рішення вийшли на перший план як зручна та ефективна альтернатива традиційним фінансовим послугам. Від платіжних платформ на зразок PayPal, Revolut чи українського Monobank, до інвестиційних додатків на кшталт Robinhood або Acorns – FinTech уже інтегрований у повсякденне життя користувачів. Ці сервіси забезпечують високу гнучкість, автоматизацію процесів та доступ до фінансів 24/7.

Розробка мобільного FinTech-застосунку є логічним кроком у цьому контексті. Серед головних причин вибору саме цієї теми виділяються наступні аспекти:

1. Зручність та доступність.

Мобільні додатки дають змогу здійснювати більшість банківських та платіжних операцій без відвідування банку – у будь-який час і з будь-якого місця. В умовах постійного ритму життя користувачам критично важливо мати можливість проводити фінансові операції швидко, просто й безпечно. Цифрові гаманці, онлайн-перекази, оплата комунальних послуг або покупок – усе це має бути реалізовано у кілька натискань у смартфоні.

2. Розширений функціонал.

Сучасні FinTech-додатки пропонують не лише базові платіжні функції, а й широкий спектр додаткових можливостей:

- персоналізовані рекомендації;
- аналітику витрат;
- інструменти бюджетування
- фінансові нагадування;
- віртуальні банківські картки тощо.

Все це підвищує цінність додатку як інструменту управління особистими фінансами та дає змогу користувачу не лише витрачати, а й планувати.

3. Безпека та захист даних.

Фінансові операції вимагають найвищого рівня безпеки. Саме тому сучасні мобільні фінансові додатки впроваджують:

- багатофакторну автентифікацію (2FA);
- біометричний захист (Face ID, Touch ID);
- динамічні паролі, одноразові коди (OTP);
- автоматичне виявлення підозрілої активності (behavioral analytics).

Система безпеки має бути комплексною, що забезпечує довіру користувачів та відповідає стандартам захисту фінансових даних (наприклад, PCI DSS).

4. Контроль та самостійне управління фінансами.

Мобільні додатки відкривають новий рівень автономії для

користувачів. У режимі реального часу можна:

- переглядати залишок на рахунку;
- керувати категоріями витрат;
- здійснювати платежі чи перекази;
- поповнювати рахунки або вкладати кошти;
- блокувати або перевипускати картки.

Така динаміка контролю стимулює фінансову грамотність та відповідальність користувачів.

5. Актуальність для українського ринку.

Україна демонструє високий темп діджиталізації у фінансовому секторі. За останні роки значно зріс попит на локальні FinTech-рішення: Monobank, Privat24, Portmone, SettlePay тощо. Водночас існує потреба у нових, більш гнучких та користувачоорієнтованих продуктах, які легко адаптуються під потреби різних цільових груп.

Розробка нового FinTech-додатку дозволить врахувати актуальні вимоги користувачів та забезпечити:

- зручний інтерфейс;
- легку інтеграцію з популярними платіжними системами (наприклад, LiqPay, Google Pay, Apple Pay);
- безпеку на рівні сучасних стандартів;
- інноваційний підхід до щоденних фінансових задач.

Отже, вибір теми розробки FinTech мобільного застосунку є цілком обґрунтованим як з боку технологічної актуальності, так і з боку практичної значущості. В умовах цифрової трансформації економіки та змін у споживчій поведінці – це напрям, який має високу соціальну та інженерну цінність, а отже, повністю відповідає завданням кваліфікаційної роботи бакалавра з програмної інженерії / інформаційних технологій.

2 ПРОГРАМНІ ЗАСОБИ FINTECH-СЕКТОРУ УКРАЇНИ

2.1 Аналіз програмних засобів FinTech

Фінансові технології (FinTech) є рушійною силою трансформації традиційного фінансового сектору, що забезпечує інноваційні рішення у банківських послугах, страхуванні, інвестуванні та управлінні фінансами. В Україні FinTech-сектор демонструє динамічний розвиток, що підтверджується як зростанням кількості стартапів, так і активною участю у глобальних фінансових тенденціях. Цей розділ присвячений аналізу програмного забезпечення, яке використовується у вітчизняному FinTech-середовищі.

Український FinTech-ринок включає понад 200 компаній, які охоплюють широкий спектр послуг: цифрові платежі, альтернативне кредитування, страхові технології (InsurTech), краудфандинг, криптовалюти, персональні фінансові сервіси тощо. Багато з цих компаній є не лише споживачами готових рішень, але й розробниками власного програмного забезпечення [4].

Серед програмних засобів, що активно використовуються FinTech-компаніями, слід виділити:

- платіжні шлюзи та процесингові системи: забезпечують швидку та безпечну обробку фінансових транзакцій. Наприклад, Fondy, Wayforpay, Portmone;
- мистеми управління ризиками та верифікації клієнтів (KYC/AML): використовують штучний інтелект і машинне навчання для виявлення шахрайських операцій та перевірки клієнтів. Популярні рішення: YouControl, Scanovate, IDWise;
- платформи для P2P-кредитування та краудфандингу: такі як Finmap, uCredit, які дозволяють оптимізувати фінансові потоки малого бізнесу та

фізичних осіб;

- мобільні банківські застосунки та цифрові гаманці: розробляються як традиційними банками (наприклад, Monobank, Sense SuperApp від Альфа-Банку), так і незалежними фінтех-компаніями;
- програмні інтерфейси (API): які дозволяють інтегрувати фінансові сервіси в інші цифрові продукти, наприклад, рішення Corezoid, Middleware.

Значну роль у функціонуванні таких систем відіграє вибір мови програмування, фреймворків, хмарних сервісів і баз даних. Зокрема, у FinTech-секморі України найчастіше застосовуються такі інструменти:

- мови програмування: Java, Python, JavaScript (Node.js), C#;
- фреймворки: Spring Boot (Java), Django (Python), .NET (C#), React/Angular (JavaScript);
- бази даних: PostgreSQL, MongoDB, MySQL, Oracle;
- хмарні платформи: AWS, Google Cloud, Microsoft Azure;
- devOps-інструменти: Docker, Kubernetes, Jenkins, GitLab CI/CD.

Варто також відзначити важливість дотримання норм кібербезпеки та відповідності регуляторним вимогам, зокрема стандартам PCI DSS, ISO/IEC 27001, а також українському законодавству в сфері захисту персональних даних.

Таким чином, програмне забезпечення є критичним компонентом інфраструктури FinTech-сектору України. Його вибір і впровадження визначають ефективність, масштабованість та безпеку фінансових сервісів, що надаються кінцевим користувачам [4].

2.2 Застосування програмних рішень та платформ

Фінансові технології (FinTech) являють собою інтеграцію сучасних цифрових рішень у фінансову сферу з метою оптимізації процесів, підвищення доступності послуг та задоволення зростаючих потреб споживачів. В умовах глобальної цифрової трансформації FinTech набуває

особливого значення, стаючи ключовим чинником модернізації банківського сектору, страхування, інвестиційної діяльності, а також персонального фінансового менеджменту. В Україні FinTech-сектор розвивається надзвичайно динамічно, попри зовнішні виклики та нестабільне економічне середовище.

Станом на 2024 рік, в Україні нараховується понад 200 FinTech-компаній, які функціонують у різних напрямах, зокрема: цифрові платежі, електронна комерція, альтернативне кредитування (P2P, P2B), InsurTech (страхові технології), криптовалюти та блокчейн-рішення, краудфандинг, автоматизовані платформи для управління особистими фінансами тощо. Багато з них не лише використовують готові програмні продукти, але й виступають розробниками власних інноваційних рішень, що підтверджує високий рівень ІТ-експертизи у країні [5].

Одним із головних чинників ефективності FinTech-компаній є застосування сучасного програмного забезпечення. В Україні активно впроваджуються як комерційні, так і open-source рішення, орієнтовані на обробку транзакцій, управління ризиками, аналітику, клієнтську підтримку, цифрову ідентифікацію та інші функції.

Серед найбільш популярних програмних рішень і платформ, що застосовуються в українському FinTech-секторі, можна виокремити такі категорії:

1. Платіжні шлюзи та процесингові системи.

Забезпечують прийом та обробку електронних платежів, включаючи підтримку банківських карток, Apple Pay, Google Pay, інтернет-банкінгу тощо. Найпоширенішими українськими рішеннями є:

- Fondy – платформа для прийому платежів онлайн, яка підтримує понад 300 методів оплати;

- Wayforpay – автоматизована система прийому онлайн-платежів з широким функціоналом для бізнесу;

- Portmone – мультифункціональний платіжний сервіс з підтримкою

комунальних платежів, переказів, онлайн-оплати товарів і послуг.

2. Системи KYC (Know Your Customer) та AML (Anti-Money Laundering).

Автоматизують процес ідентифікації та перевірки клієнтів відповідно до регуляторних вимог. Це дозволяє зменшити ризики шахрайства та запобігати відмиванню грошей. В Україні використовуються:

- YouControl – сервіс для перевірки бізнес-контрагентів та аналізу ризиків;

- IDWise – рішення для віддаленої ідентифікації клієнтів з використанням AI-технологій;

- Scanovate – платформа для комплексного управління відповідністю нормативним вимогам.

3. Платформи для краудфандингу та P2P-кредитування.

Ці рішення забезпечують пряме фінансування між користувачами без посередництва банків. Наприклад:

- uCredit – онлайн-платформа для мікрокредитування фізичних осіб%

- Finmap – фінансовий облік для малого бізнесу, що допомагає в управлінні грошовими потоками.

4. Цифрові гаманці та мобільні фінансові застосунки.

В Україні розвивається концепція «необанкінгу» (небанківські фінансові сервіси), що базується на мобільних додатках з повним спектром банківських функцій. Найяскравішими прикладами є:

- Monobank – перший український мобільний банк без фізичних відділень.

- Sense SuperApp – застосунок від Альфа-Банку, що поєднує банківські, інвестиційні та лайфстайл-сервіси.

5. Інтерфейси програмування застосунків (API).

API дозволяють інтегрувати фінансові сервіси до сторонніх продуктів, створюючи екосистеми для партнерських програм. В Україні використовуються:

- Corezoid – система управління процесами в реальному часі;
- Middleware – платформа для об'єднання цифрових сервісів із банківськими системами.

2.3 Технічна база FinTech-програмного забезпечення

Розробка програмного забезпечення для фінансового сектору потребує високого рівня надійності, безпеки та масштабованості. У FinTech-компаніях України активно використовуються такі інструменти:

1. Мови програмування:

- Java – для розробки бекенд-систем та обробки великого обсягу транзакцій;
- Python – у сфері аналітики, обробки даних, машинного навчання;
- JavaScript / Node.js – для створення інтерфейсів і обробки запитів у режимі реального часу;
- C# – у фінансових системах із прив’язкою до Windows-інфраструктури.

2. Фреймворки та платформи:

- Spring Boot, .NET, Django, Express.js, React, Angular.

3. Системи керування базами даних:

- PostgreSQL, MongoDB, Oracle, MySQL – залежно від вимог до надійності, масштабування та обробки транзакцій.

4. Хмарні платформи:

- Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure – для розгортання масштабованих сервісів із високою доступністю.

5. DevOps-інструменти:

- Docker, Kubernetes, Jenkins, GitLab CI/CD – для автоматизації розгортання, тестування та оновлення програмного забезпечення.

2.3.1 Безпека та відповідність регуляторним вимогам

FinTech-сектор потребує суворого дотримання стандартів інформаційної безпеки та конфіденційності. Найпоширенішими стандартами є:

- PCI DSS – стандарт безпеки для компаній, які працюють із банківськими картками;
- ISO/IEC 27001 – міжнародний стандарт управління інформаційною безпекою;
- Закон України «Про захист персональних даних» – регулює обробку, зберігання та передачу персональних даних клієнтів [6-7].

Таким чином, український FinTech-сектор активно використовує як вітчизняні, так і міжнародні програмні рішення, які відповідають сучасним вимогам цифрової економіки. Висока гнучкість, інноваційність та технологічна обізнаність розробників дозволяють створювати конкурентоспроможні продукти на глобальному ринку. Програмне забезпечення є основним інструментом, що забезпечує функціонування, масштабування та безпеку фінансових сервісів у цифровому середовищі.

3 РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

3.1 Технологічна база проєкту

Для ефективної розробки мобільного застосунку необхідно використовувати перевірені, сучасні та підтримувані інструменти, що забезпечують стабільність, зручність у роботі, а також масштабованість продукту. У межах даного проєкту було обрано Android Studio як основне середовище розробки, Java як мову програмування, Firebase як хмарну платформу для зберігання та обробки даних, а також застосовано архітектурний патерн MVP (Model-View-Presenter) для покращення структури програми. Нижче розглянуто доцільність використання кожного з цих інструментів.

Android Studio – інтегроване середовище розробки. Android Studio — це офіційне середовище розробки програмного забезпечення для Android, створене компанією Google на основі IntelliJ IDEA. Середовище підтримує мову Java, включає зручний візуальний редактор, шаблони макетів, потужну систему складання на Gradle, а також можливість миттєвого внесення змін у запущений застосунок (Instant Run) [8].

Проєкт у Android Studio складається з модулів, що містять вихідний код і ресурсні файли для Android-додатків, бібліотек або хмарної платформи Google App Engine.

Редактор коду надає підказки, перевірку синтаксису та можливість автоматичного виправлення помилок. Після компіляції можна отримати .apk файл для подальшого розміщення застосунку в Google Play.

Java – мова програмування. Java – це сучасна об'єктно-орієнтована мова програмування, розроблена компанією JetBrains. Вона є офіційною мовою для розробки Android-додатків з 2017 року [9].

Java працює на Java Virtual Machine (JVM) та вирізняється статичною

типізацією, лаконічністю синтаксису, високою безпекою та зворотною сумісністю з Java. Завдяки відкритому коду, активній підтримці спільноти та інтеграції з Android Studio, мова Java стала провідною серед Android-розробників.

Підтримка Java на рівні Google забезпечує доступ до найновіших можливостей платформи Android.

Firebase – хмарна платформа для мобільних застосунків. Firebase – це платформа, розроблена Google, яка надає повний набір інструментів для створення, запуску та масштабування мобільних застосунків. Вона дозволяє зосередитись на розробці бізнес-логіки, автоматизуючи роботу з backend-частиною [10].

Серед основних можливостей платформи – бази даних у реальному часі, система аутентифікації, аналітика, push-повідомлення, зберігання файлів та інші сервіси.

Оскільки Firebase функціонує у хмарному середовищі, це дає змогу масштабувати застосунок у міру зростання користувачів без потреби у власному серверному рішенні.

Архітектура MVP (Model-View-Presenter). MVP – це архітектурний патерн, який дозволяє розділити логіку програми на три основні компоненти [11]:

- Model – відповідає за обробку даних, доступ до API, бази даних або кешу;
- View – представляє інтерфейс користувача та займається лише візуалізацією;
- Presenter – виступає посередником між Model та View, координуючи взаємодію між ними.

Завдяки такому розподілу відповідальностей, MVP забезпечує кращу тестованість коду, спрощує масштабування і підтримку застосунку, що є критично важливим для довгострокових проєктів.

3.2 Вибір інструментів для розробки

1. Переваги Android1 Studio.

Android Studio, крім розширеного редактора коду IntelliJ та широкого набору інструментів для розробників, надає додаткові можливості, що істотно підвищують ефективність створення Android-додатків. До таких можливостей належать:

- гнучка система збирання проєктів, яка базується на Gradle;
- високошвидкісний та функціонально багатий емулятор;
- єдине середовище для розробки застосунків під усі типи Android-пристроїв;
- можливість застосовувати зміни до коду та ресурсів без необхідності перезапуску додатку;
- шаблони коду та інтеграція з GitHub, що полегшують реалізацію стандартних функцій і імпорт прикладів;
- наявність розвинених засобів і фреймворків для тестування;
- інструменти Lint для виявлення проблем продуктивності та несумісності версій;
- підтримка розробки на C++ через інтеграцію з NDK;
- вбудована інтеграція з Google Cloud Platform для полегшення роботи з Google Cloud Messaging та App Engine.

2. Переваги Java.

Мова Java має низку переваг, які дозволяють їй перевершувати Java, особливо у контексті розробки під Android. Java спроектовано для усунення недоліків у дизайні API Java, зберігаючи при цьому сумісність.

- Короткість. Java дозволяє писати менш громіздкий і легший для сприйняття код. Завдяки мінімалізму синтаксису зменшується кількість шаблонного коду, що знижує імовірність помилок. При цьому зберігається читабельність, не жертвуючи зрозумілістю заради стислості;
- модулі, написані на Java, можуть працювати в існуючих Java-

проєктах без ускладнень. Це можливо завдяки тому, що компілятор Java створює байт-код, сумісний з JVM;

- безпека. Java включає систему типів із підтримкою null-безпеки. Це дозволяє уникнути поширеної помилки NullPointerException, яка часто виникає в Android-розробці;

- відсутність необроблених типів. Java не дозволяє використовувати необроблені типи, що усуває потенційні помилки, пов’язані з CastClassException, ще на етапі компіляції. Це підвищує надійність і безпечність коду.

3. Переваги Firebase.

Firebase є комплексною платформою для створення веб- і мобільних застосунків. Вона дозволяє покращити якість продукту, спростити підтримку та сприяти зростанню бізнесу.

Ключові переваги платформи Firebase:

- керування даними в реальному часі, що забезпечує швидкий та ефективний обмін інформацією між користувачами, наприклад, для чатів чи стрімінгових сервісів;

- миттєва синхронізація даних між Android, iOS і вебпристроями без потреби в ручному оновленні;

- інтеграція з Google Ads, AdMob, DoubleClick, Play Store, Data Studio, BigQuery та Slack, що забезпечує централізоване керування розробкою та аналітикою;

- включення аналітики, бази даних, звітів про помилки — все в єдиній екосистемі;

- Firebase Realtime Database – хмарна NoSQL-база, яка синхронізує JSON-дані між усіма клієнтами у реальному часі;

- підтримка офлайн-режиму, де дані зберігаються локально та автоматично синхронізуються після з’єднання з мережею;

- висока інтеграція з Firebase Authentication, що спрощує побудову безпечних моделей доступу;

- відсутність потреби в серверному забезпеченні завдяки мобільним і веб SDK.

4. Переваги MVP.

Архітектурний шаблон MVP (Model-View-Presenter), що базується на підході MVC, надає високий рівень гнучкості та дозволяє реалізовувати нові функції із мінімальними витратами, підвищуючи якість кінцевого продукту.

Основні переваги використання MVP:

- повторне використання коду. Принцип розділення відповідальності дозволяє створювати логічно розмежовані компоненти, що полегшує їх повторне застосування у майбутніх проєктах;

- тестованість. Ізоляція компонентів (View, Presenter, Model) дозволяє легко проводити модульне тестування, що важливо для перевірки стабільності програми;

- гнучкість у розробці. Якісний дизайн та ізоляція компонентів спрощують внесення змін, дають змогу застосовувати кілька видів уявлень і джерел даних;

- багаторівнева структура. Чіткий поділ між логікою відображення (View) і бізнес-логікою (Model/Presenter) забезпечує прозору архітектуру та полегшує підтримку коду [12-13].

У розділі проаналізовано програмне забезпечення, використане під час розробки онлайн платіжної системи, зокрема функціональні можливості Android Studio, переваги мови Java порівняно, переваги платформи Firebase та доцільність застосування шаблону архітектури MVP для створення користувацького інтерфейсу.

4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДКТУ

4.1 Розробка функціоналу програми

Функціонал програми онлайн платіжної системи складатиметься з реєстрації, входу/виходу з облікового запису, платежів, переказів та історії платежів.

При відкритті програми користувач побачить сторінку входу. Якщо користувач вже зареєстрований, він може увійти в обліковий запис, використовуючи свій логін і пароль. В іншому випадку, користувачеві необхідно зареєструватися, ввівши свій логін, пароль, ім'я та прізвище. При вході в обліковий запис користувач потрапить на сторінку з платежами, знизу буде навігаційна панель, за допомогою якого можна переходити на інші сторінки. Серед інших сторінок можна перерахувати:

- сторінку з банківською картою, де буде написано її номер, дата закінчення терміну та виробник, також там можна заблокувати картку;
- сторінку з історією, де будуть вказані суми, дати та вид платежів;
- сторінку переказів, де можна надіслати гроші іншому користувачеві, що зареєструвався, за його номером картки;
- сторінку із повідомленнями;
- сторінку з профілем користувача з ім'ям та прізвищем та кнопкою виходу з облікового запису в правому верхньому кутку.

При переказі або платежі перевіряється рахунок користувача, якщо сума переказу або платежу перевищує суму на рахунку, то з'явиться спливаюче вікно з повідомленням, що не вистачає грошей на рахунку. При успішному переказі або платежі з'явиться сторінка з повідомленням про деталі операції. Якщо користувач вирішить заблокувати карту, він не зможе здійснювати жодні операції з цього облікового запису.

4.2 Створення сторінки програми

Для початку, у проекті потрібно створити макети сторінок додатків. Почав робити макети сторінок після входу до облікового запису.

На рисунку 4.1 показані списки операцій, доступні користувачеві за допомогою електронної банківської картки.

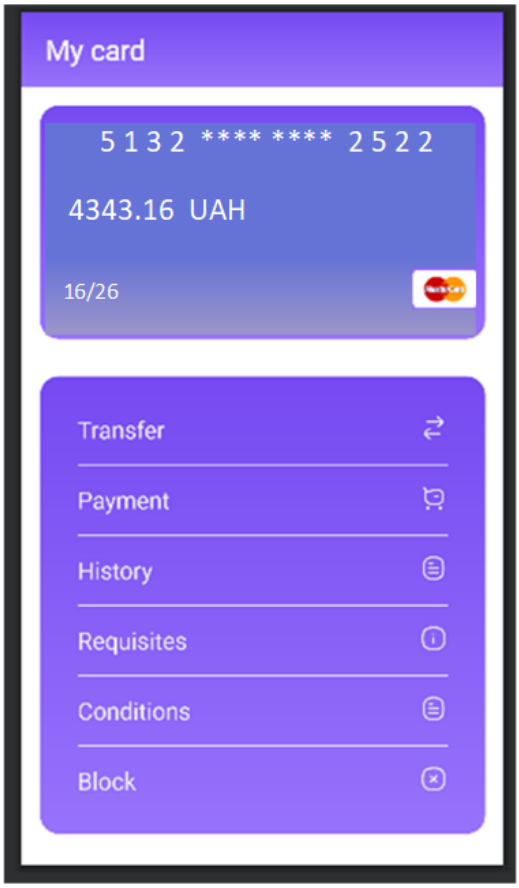

Рисунок 4.1 – Макет сторінки з карткою та списком операцій

На рисунку 4.2 показані всі операції з платежів, зроблені останнім часом.

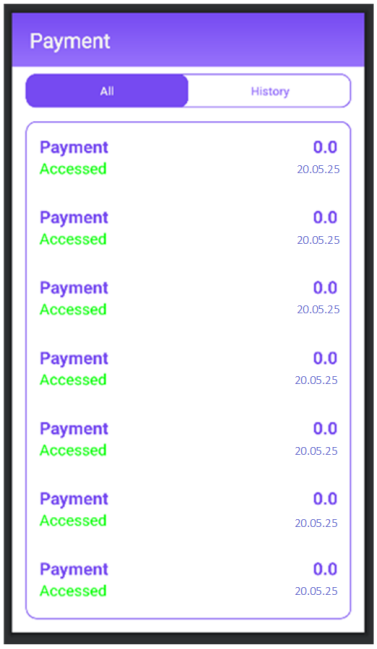

Рисунок 4.2 – Макет сторінки історій платежів

На рисунку 4.3 показані дані для здійснення платежу, номер платежу (мобільний телефон, комунальні послуги тощо) та сума. Натиснувши кнопку Transfer, можна здійснити платіж.

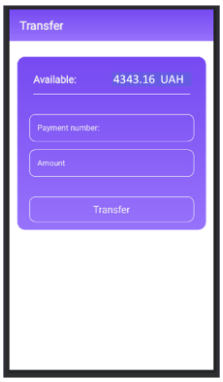

Рисунок 4.3 – Макет сторінки здійснення платежу

На рисунок 4.4 показані дані для здійснення переказу, номер картки користувача та сума. Натиснувши кнопку Transfer можна переказати гроші користувачу.

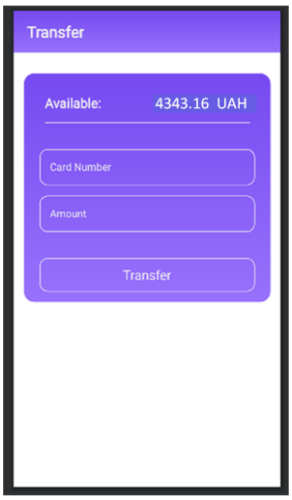

Рисунок 4.4 – Макет сторінки здійснення перекладу

На рисунку 4.5 можна розглянути сторінку профілю користувача Там є прізвище та ім'я користувача, його аватар і списки доступних операцій, такі як перекази, платежі та вихід з облікового запису на правому верхньому кутку

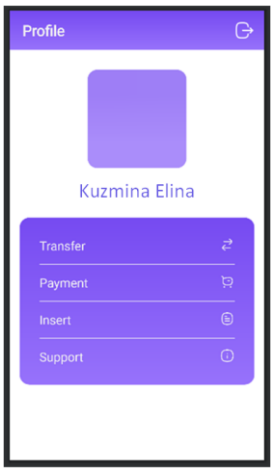

Рисунок 4.5 – Макет сторінки профілю

На рисунку 4.6 вказані дані для входу в обліковий запис, логін та пароль. Унизу кнопки входу та реєстрації.

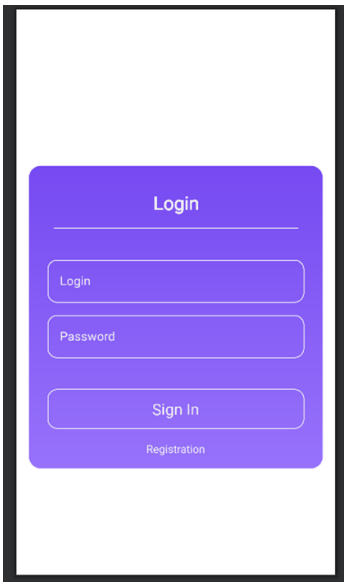

Рисунок 4.6 – Макет сторінки входу до облікового запису

Рисунок 4.7 вказані дані для реєстрації нового користувача, логін, пароль, ім'я та прізвище. Унизу знаходиться кнопка реєстрації.

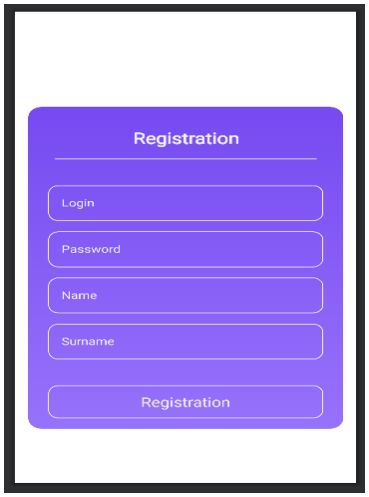

Рисунок 4.7 – Макет сторінки результату платежу

4.3 Процес розробки програми

На початковому етапі розробки програми в класі MainActivity відбувається створення та ініціалізація всіх глобальних змінних, як показано на рисунку 4.8.

```
class MainActivity : AppCompatActivity() {

    lateinit var binding: ActivityMainBinding
    lateinit var navController:NavController
    lateinit var transfer:Transfer
    lateinit var payment:String
    lateinit var paymentObject:Payment
    var isAccess = false
    lateinit var token:String

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
        init()
    }


    private fun init() {
        Constants.mainActivity = this
        token = Constants.token
        val navHostFragment =
            supportFragmentManager.findFragmentById(R.id.fragmentContainerView) as NavHostFragment
        navController = navHostFragment.navController
        binding.bottomNavigation.setupWithNavController(navController)
    }

}
```

Рисунок 4.8 – Створення та ініціалізація глобальних змінних

Оскільки під час розробки проєкту була використана архітектура MVP, було створено відповідні класи та інтерфейси презентерів. У цих компонентах реалізуються необхідні запити (рисунок 4.9).

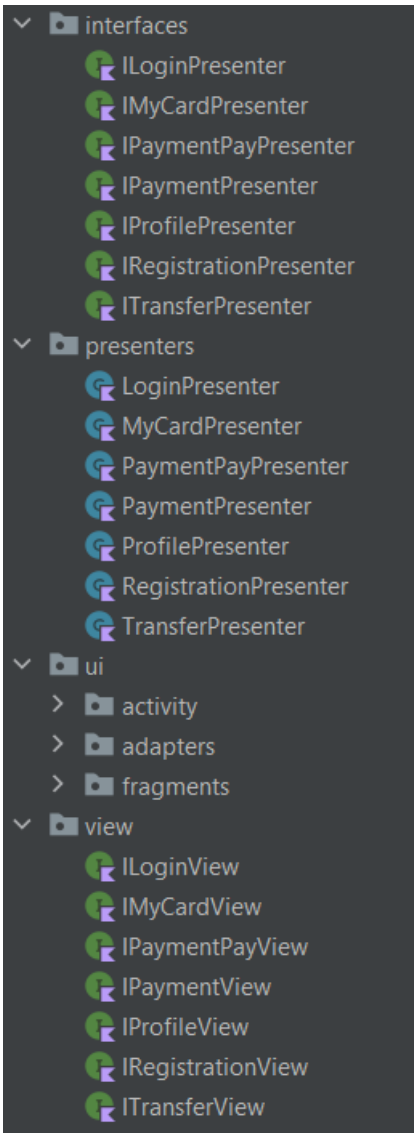

Рисунок 4.9 – Створення класів та інтерфейсів

Також було розроблено класи моделей для представлення створених об'єктів, підключення моделей у файлі FirebaseDao,, додавання користувача до бази даних FirebaseDao як показано на рисунах 4.10-4.12.

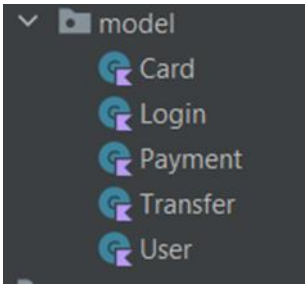

Рисунок 4.10 – Створення моделей для об'єктів

```
package com.example.fintech.data.firebase

import android.util.Log
import com.example.fintech.data.model.Card
import com.example.fintech.data.model.Login
import com.example.fintech.data.model.User
import com.example.fintech.utils.Constants
import com.google.firebase.firestore.FirebaseFirestore
import com.google.firebase.firestore.SetOptions
import java.util.*
import kotlin.collections.HashMap

class FirebaseDao (
    private var db:FirebaseFirestore,
    private var token:String
        ) {
```

Рисунок 4.11 – Підключення моделей у файлі FirebaseDao

```
fun addLogin(login: Login) {
    val loginHash = hashMapOf(
        "login" to login.login,
        "password" to login.password,
        "token" to login.token
    )

    db.collection( collectionPath: "logins")
        .document(login.token)
        .set(loginHash)
}

fun addUser(user: User) {
    val userHash = hashMapOf(
        "name" to user.name,
        "last_name" to user.last_name,
        "card" to user.card,
        "bonuses" to user.bonuses,
        "userImagePath" to user.userImagePath,
        "uuid" to user.uuid
    )

    db.collection( collectionPath: "users")
        .document(user.uuid)
        .set(userHash)
}
```

Рисунок 4.12 – Додавання користувача до бази даних FirebaseDao

У класі FirebaseDao дані бази даних беруться з моделей (рисунок 4.11). Далі стоять функції додавання користувачів (рисунок 4.12) та функція відображення суми з картки користувача (рисунок 4.13).

```
fun setMoney (card:Card, cardHashMap: HashMap<String, Card>) {
    Constants.db.collection( collectionPath: "users")
        .document(card.uuid)
        .set(cardHashMap, SetOptions.merge())
        .addOnSuccessListener { documentReference ->
            Log.d( tag: "TAG",  msg: "DocumentSnapshot added with ID: ${documentReference}")
        }
        .addOnFailureListener { e ->
            Log.w( tag: "TAG",  msg: "Error adding document", e)
        }
}
```

Рисунок 4.13 – Відображення суми з картки користувача у базі даних

У класі MyCardPresenter створюються запити для отримання даних картки користувача, блокування та розблокування картки (рисунок 4.14).

```
class MyCardPresenter (
    private val view:IMyCardView
): IMyCardPresenter {
    lateinit var card:Card
    private val firebaseDao = FirebaseDao(Constants.db, Constants.token)

    override fun getCard(token: String) {
        Constants.db.collection( collectionPath: "users")
            .whereEqualTo( field: "uuid", token)
            .get()
            .addOnSuccessListener { result ->
                for (document in result) {
                    card = document.toObject(User::class.java).card
                    Log.d( tag: "TAG+token", token)
                }
                view.setData(card)
            }
            .addOnFailureListener { exception ->
                Log.w( tag: "TAG",  msg: "Error getting documents.", exception)
            }
    }

    override fun blockCard() {
        card.isBlock = true
        val cardFromHash = hashMapOf(
            "card" to card
        )
        firebaseDao.setMoney(card, cardFromHash)
    }

    override fun unBlockCard() {
        card.isBlock = false
        val cardFromHash = hashMapOf(
            "card" to card
        )
        firebaseDao.setMoney(card, cardFromHash)
    }
}
```

Рисунок 4.14 – Запити для отримання карток

Платіж проходить перевірку стану рахунку. Тобто, уточнюються питання щодо нестачі чи хватки певної суми на рахунку (рисунок 4.15). Також рахунок перевіряється на доступ, тобто, заблокований чи ні. Усі вихідні запити знаходяться у класі FirebaseDao.

```
override fun doPayment (amount:Double, paymentNumber: String) {
    if (myCard.amount > amount && !myCard.isBlock) {
        myCard.amount -= amount
        val cardFromHash = hashMapOf(
            "card" to myCard
        )
        val payment = Payment(UUID.randomUUID().toString(), Date(),  isAccess: true, Constants.mainActivity.payment,  description: "", amount)
        myCard.historyPayment.add(payment)
        firebaseDao.setMoney(myCard, cardFromHash)

        view.doPayment(payment)
    }
}
```

Рисунок 4.15 – Перевірка рахунку

Перед переказом грошей, перевіряється присутність інших карток, доступність суми на рахунку, доступ до операцій (рисунок 4.16).

```
override fun doTransfer(cardNumber: String, amount: Double): Boolean {
    if (checkTransfer(cardNumber,amount)) {
        changeAmount(amount)
        Toast.makeText(Constants.mainActivity, text: "Transfer successful", Toast.LENGTH_SHORT).show()
        return true
    }
    Toast.makeText(Constants.mainActivity, text: "Transfer failed", Toast.LENGTH_SHORT).show()
    return false
}

private fun checkTransfer(cardNumber:String, amount: Double):Boolean {
    var isCardTrue = false
    for (card in cards) {
        if (card.cardNumber.filterNot { it.isWhitespace() } == cardNumber) {
            isCardTrue = true
            secondCard = card
        }
    }

    return !myCard.isBlock || isCardTrue || myCard.amount > amount
}

private fun changeAmount(amount: Double) {
    myCard.amount -= amount
    secondCard.amount += amount
    val cardFromHash = hashMapOf(
        "card" to myCard
    )
    val cardToHash = hashMapOf(
        "card" to secondCard
    )

    val transfer = Transfer(myCard.cardNumber, secondCard.cardNumber, amount)

    myCard.historyTransfer.add(transfer)
    firebaseDao.setMoney(myCard, cardFromHash)
    firebaseDao.setMoney(secondCard, cardToHash)

    view.doTransfer( isAccess: true, transfer)
}
```

Рисунок 4.16 – Функції перекладу, перевірки та оновлення даних під час перекладу

У класі LoginPresenter написані перевірки при вході до облікового запису. Функція getLogins бере дані для входу з бази даних. checkLogin перевіряє правильність написання даних при вході та виводить відповідне повідомлення при неправильному його введенні (рисунок 4.17).

```
class LoginPresenter (
    private val view:ILoginView
        ) : ILoginPresenter {

    var logins:List<Login> = mutableListOf()
    override fun getLogins() {
        Constants.db.collection( collectionPath: "logins")
            .get()
            .addOnSuccessListener {  it: QuerySnapshot!
                logins = it.toObjects(Login::class.java)
            }
    }

    override fun checkLogin(login: String, password: String) {
        var myLogin = Login()
        for (data in logins) {
            if (login == data.login && password == data.password) {
                myLogin = data
            }
        }
        if (myLogin.token == "") {
            Toast.makeText(Constants.loginActivity,  text: "Incorrect password or login", Toast.LENGTH_SHORT).show()
        } else {
            view.signIn(myLogin)
        }
    }

}
```

Рисунок 4.17 – Функції для реєстрації та перевірки даних

У класі RegistrationPresenter знаходяться функції реєстрації та перевірки даних. Функція getLogins бере дані користувачів із бази даних. Функція registration перевіряє, чи існує логін, що вводиться під час реєстрації, в базі даних, і якщо він вже є, то треба вибрати інший логін. При правильному заповненні всіх форм дані нового користувача записуються в базу даних і йому присвоюються номер картки і токен (рисунок 4.18).

```
class RegistrationPresenter (
    private val view: IRegistrationView
        ) : IRegistrationPresenter {
    var logins = mutableListOf<Login>()
    lateinit var firebaseDao:FirebaseDao
    override fun getLogins() {
        Constants.db
            .collection( collectionPath: "logins")
            .get()
            .addOnSuccessListener {  it: QuerySnapshot!
                logins = it.toObjects(Login::class.java)
            }
    }

    override fun registration(login: String, password: String, name: String, surname: String) {
        var isAccess = true
        for (data in logins) {
            if (data.login == login) {
                isAccess = false
            }
        }

        if (isAccess) {
            val token = UUID.randomUUID().toString()
            val loginObject = Login(login, password, token)
            val cardNumber = "4169 2500 4600 ${(1000..9999).random()}"
            val card = Card(cardNumber,  date: "11/24",  code: 882,  amount: 10000.0, uuid=token)
            val userObject = User(name, surname, card,  bonuses: 0.0,  userImagePath: "", token)
            firebaseDao = FirebaseDao(Constants.db, token)
            firebaseDao.addUser(userObject)
            firebaseDao.addLogin(loginObject)
            view.registration(token)
        }
    }
}
```

Рисунок 4.18 – Функції для реєстрації та перевірки даних

Після авторизації в Firebase через Google обліковий запис можна перейти в консоль і звідти перейти в Firestore Database для перегляду бази даних. У базі даних зберігаються дані про користувачів, їх дані картки, суми на рахунку, історію переказів та платежів (рисунок 4.19).

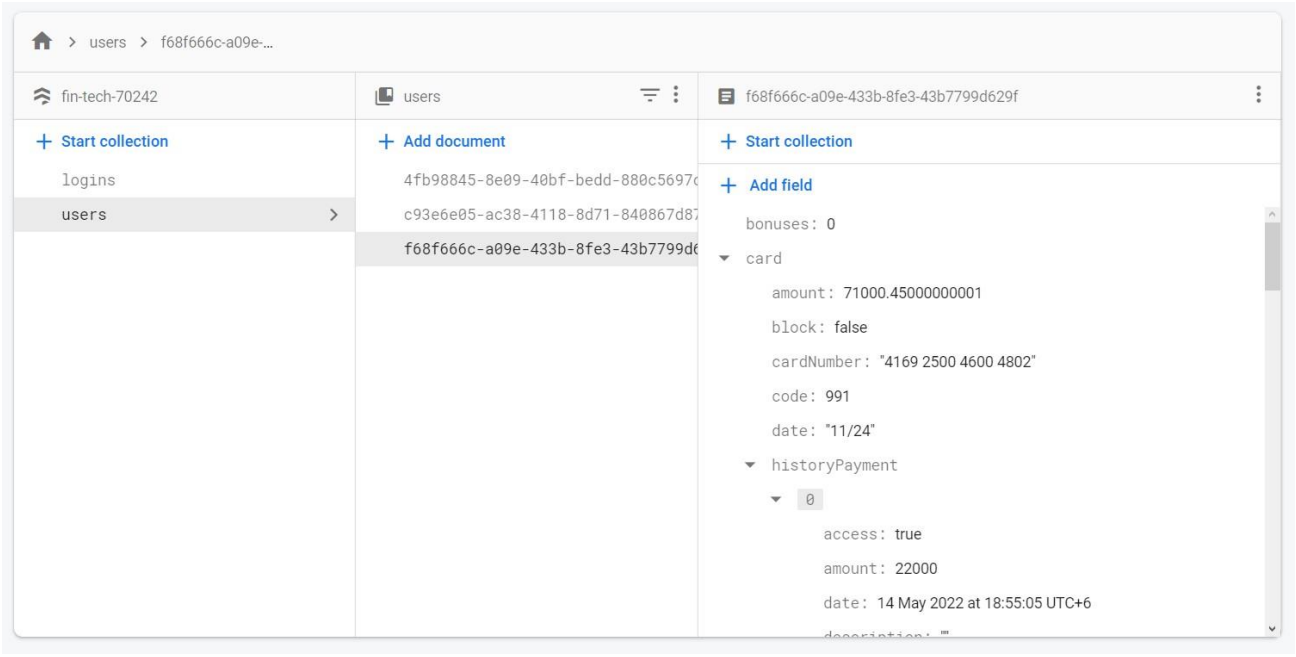

Рисунок 4.19 – Firestore Database

У базі даних зберігаються дані про користувачів, їх дані картки, суми на рахунку, історію переказів та платежівмм Під час переказу або платежів дані про рахунок користувача будуть оновлюватись у реальному часі.

ВИСНОВКИ

Результатом кваліфікаційної роботи є автоматизований мобільний додаток для FinTech галузі. Продукт є загальнодоступною програмою для користувачів різних вікових категорій. Додаток використовується для ведення гаманця онлайн і проведення операцій з наявним рахунком онлайн. Основною вимогою до додатку була простота використання. За підсумками роботи можна сказати, що додаток є більш ніж зручним у використанні та приємним для візуального враження.

У процесі дослідження сучасного фінансових справ після пандемії COVID-19. На основі дослідження зроблено наступний висновок: сучасні великі компанії використовують змінний ландшафт фінтеху на благо свого бізнесу. Нові гіганти бізнесу, що ростуть, ставлять перед собою стратегію для кращого майбутнього, і думають про те, як фінтех може – і буде – впливати на вашу організацію. Розробляються та впроваджуються нові та старі технології. А нові продукти відкривають нові можливості та замінюють застарілі рішення.

У процесі проектування уточнювалися вимоги в створенні мобільного застосунку в середовищі Android Studio. мови програмування Java, як одного з основних інструментів у процесі розробки мобільного додатка.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Національний банк України. Звіт про фінансову стабільність [Електронний ресурс]. – Київ: НБУ, 2023. – Режим доступу: https://bank.gov.ua.

2. PwC Україна. FinTech в Україні: стан та перспективи розвитку [Електронний ресурс]. – PwC, 2022. – Режим доступу: https://www.pwc.com/ua.

3. Васильченко І. В. Цифрова трансформація фінансових послуг: глобальні тренди та українські реалії // Вісник економіки транспорту і промисловості. – 2021. – № 75. – С. 40–45.

4. Коваленко О. М. Розвиток FinTech в Україні: виклики та можливості // Економіка та держава. – 2023. – № 3. – С. 87–92.

5. Ernst & Young. Global FinTech Adoption Index 2023 [Електронний ресурс]. – EY, 2023. – Режим доступу: https://www.ey.com.

6. Deloitte Україна. Фінансові технології в Україні: нові можливості та ризики [Електронний ресурс]. – Deloitte, 2022. – Режим доступу: https://www2.deloitte.com/ua.

7. Закон України «Про захист персональних даних» : Закон України від 01 черв. 2010 р. № 2297-VI [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2297-17.

8. Android Framework [Електронний ресурс]. – 2025. – Режим доступу: https://www.quora.com/What-is-meant-by-Android-framework

9. What is Java? The Java alternative explained [Електронний ресурс] // InfoWorld. – Режим доступу: https://www.infoworld.com/article/3224868/what-is-Java-the-java-alternative-explained.html

10. Java vs Java – Which is the Better Option for Android App Development? [Електронний ресурс] // Clearbridge Mobile. – Режим доступу: https://clearbridgemobile.com/java-vs-Java-which-is-the-better-option-for-android-

app-development/

11. Why Firebase is the Best Backend Platform for Mobile App Development [Електронний ресурс] // TriState Technology. – Режим доступу: https://www.tristatetechnology.com/blog/firebase-backend-mobile-app.

12. Get Started with Firebase for Android [Електронний ресурс] // Tuts+ Code. – Режим доступу: https://code.tutsplus.com/ru/tutorials/get-started-with-firebase-for-android--cms-27248.

13. Model View Presenter (MVP) [Електронний ресурс] // Medium. – Режим доступу: https://medium.datadriveninvestor.com/model-view-presenter-mvp-5c3439227f83