

# ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ .....	3
ВСТУП .....	4
1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ОНЛАЙН БАНКІНГУ ТА ОБГРУНТУВАННЯ ТЕМИ БАКАЛАВРСЬКОГО ПРОЕКТУ .....	5
1.1. Опис існуючих рішень Інтернет банкінгу .....	5
1.2. Безпека при виконанні дистанційних платіжних операцій .....	10
1.2.1 Шифрування у банківських системах .....	11
1.2.2. Рівень захищених сокетів SSL .....	12
1.2.3. Двофакторний метод автентифікації (2FA).....	13
1.3. Обґрунтування теми проекту та обраних засобів розроблення .....	15
2. АНАЛІЗ ІНСТРУМЕНТІВ РОЗРОБКИ ПРОГРАМНОГО ТА АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ .....	16
2.1. IntelliJ IDEA, SAP PowerDesigner та MySQL Workbench .....	16
2.2. Характеристики веб-серверу .....	19
2.3. Характеристики серверу баз даних.....	21
2.3.1 Ядро бази даних InnoDB.....	22
2.3.2 RAID10 .....	22
3. РОЗРОБЛЕННЯ ПРОГРАМНОГО ТА АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ.....	24
3.1.Розробка концепції рішення .....	24
3.1.1 Етапи розробки системи банківських операцій користувача .....	26

					ІАЛЦ.467200.004 ПЗ			
Змін.	Арк.	№ докум.	Підпис	Дата				
Розробив	Горпинич-Радуженко І.О.				Веб-орієнтована система банківських операцій користувача.  Пояснювальна записка	Літ.	Аркуш	Аркушів
Перевірив	Орлова М.М.						1	52
Консульт.						НТУУ «КПІ» Кафедра СПіСКС Група КВ-41		
Н. контроль	Клятченко Я.М.							
Зав. каф.	Тарасенко В.П.							

3.2.Розробка модуля шифрування на основі алгоритму SHA-256.....	41
4. АНАЛІЗ ВИКОНАНОЇ СИСТЕМИ ОНЛАЙН БАНКІНГУ .....	43
4.1.Опис програмного забезпечення .....	43
ВИСНОВКИ .....	51
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....	52

## ДОДАТКИ

### Додаток 1. Копії графічного матеріалу

ІАЛЦ.467200.005 Е1. Взаємодія модулів системи. Схема структурна.

ІАЛЦ.467200.006 Е1. Структура бази даних. Схема структурна.

ІАЛЦ.467200.007 Е1. Робота модуля шифрування SHA-256. Схема структурна.

ІАЛЦ.467200.008 Е1. Система шифрування SHA-256. Схема структурна.

Плакат. Взаємодія блоків системи.

### Додаток 2. Лістинг програми.

### Додаток 3. Довідка про впровадження.

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		2

## СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

2FA	—	дворівневий метод автентифікації;
API	—	набір готових класів, процедур, функцій, структур і констант, що надаються додатком;
SMS	—	технологія, яка дозволяє здійснювати прийом і передачу коротких текстових повідомлень;
SSL	—	криптографічний протокол;
Байткод	—	компактне представлення програми, котра пройшла синтаксичний і семантичний аналіз;
Інкапсулювання	—	приховування інформації;
Кросс-платформеність	—	здатність програмного забезпечення працювати більш ніж на одній апаратній платформі;
Об'єкт	—	сукупність обладнання, яке підключено до інтерфейсного блоку;
Скрипт	—	програмний сценарій, який автоматизує деяку задачу;
Сокет	—	програмний інтерфейс для забезпечення обміну даними між процесами;
СУБД	—	система управління базами даних;
Токен	—	компактний пристрій, призначений для забезпечення ідентифікації його власника при доступі до віддалених інформаційних ресурсів;

## ВСТУП

На сьогоднішній день, дистанційні фінансові операції є невід’ємною частиною соціального існування кожної людини у світі. Цей факт особливо стосується людей малого та середнього фінансового достатку, що є більшою частиною земного населення. При виконанні банківських операцій вже не потрібна особиста присутність людини у банківській установі, достатньо мати ідентифіковані персональні дані у системі банківського контролю та доступ до мережі Інтернет, але є нагальна необхідність у підвищенні рівня банківської освіченості, оскільки дистанційні банківські операції виконуються в автоматичному режимі.

Банківські операції користувача включають в себе роботу з банківськими онлайн переказами, депозитними та кредитними рахунками, управління банківськими картками.

Банки та інші фінансові установи працюють над розробкою та впровадженням веб-орієнтованих систем через їх потенціал підвищення ефективності, скорочення витрат і залучення нових клієнтів. Споживачі починають використовувати ці технології через підвищену зручність та економію часу. В період з 1995 по 2003 рік, кількість електронних банківських операцій збільшилася у 8 разів. У період з кінця 2002 року до початку 2005 року, використання веб-орієнтованих банківських мереж збільшилася на 47 відсотків [1]. Лише в Україні на початок 2017 року системами дистанційних фінансових операцій користуються більш ніж 10 мільйон громадян [2].

Проте відсутність фінансової освіченості викликає цифровий розрив та віддалення людей малого та середнього фінансового достатку від використання усіх можливих переваг онлайн банкінгу. Тому є нагальна необхідність у розробці веб-орієнтованої банківської системи операцій користувача для здобуття необхідних навичок використання цих систем.

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		4

# 1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ТЕМИ БАКАЛАВРСЬКОГО ПРОЕКТУ

## 1.1. Аналіз існуючих систем онлайн банкінгу

Розглянемо та проаналізуємо характеристики існуючих систем онлайн банкінгу в Україні.

На сьогоднішній день в Україні існує багато веб-орієнтованих систем для надання банківських послуг як від самих банків, так і комерційні рішення інших виробників програмного забезпечення. Розглянемо декілька основних, з точки зору можливості навчання роботи з системами онлайн банкінгу нових користувачів.

Система «Privat24» [3]

Лідером на ринку України є система «Privat24», оскільки переважна кількість державного безготівкового грошового обігу виконується за допомогою цього додатку.

Програмна реалізація має необхідну, для теперішнього часу, дворівневу систему автентифікації, яка дозволяє запобігти несанкціонованому доступу до даних користувача.

Для початку роботи потрібно лише мати телефон, який вказаний при реєстрації у банку. Після декількох годин користувач автоматично буде зареєстрований у системі онлайн банкінгу (рисунок 1.1).

Недоліком є досить складний інтерфейс, який не має достатньої кількості підказок для нових користувачів.

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		5

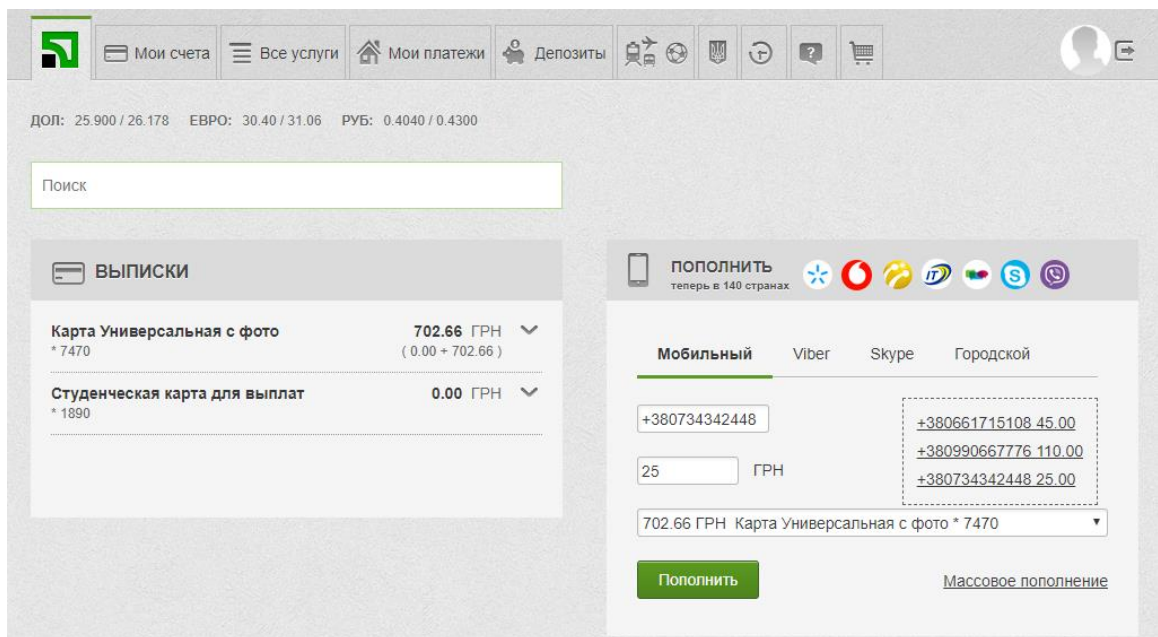


Рисунок 1.1 - Интерфейс онлайн банкінгу «Privat24»

#### Система «Ощад24» [4]

Система державного банку - «Ощадбанку» (рисунок 1.2). Процедура автентифікації дуже схожа на «Приват24», оскільки потребує підключення SMS банкінгу для можливості дворівневої автентифікації. В системі є базовий функціонал для роботи з депозитними рахунками (можливість роботи лише з декількома видами).

Недоліком є система переказу грошів між картками власного банку, яку необхідно виконувати лише за номером. При такому підході користувачеві дуже легко помилитися. Великим недоліком є неможливість використання альтернативних можливостей автентифікації, окрім SMS.

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		6

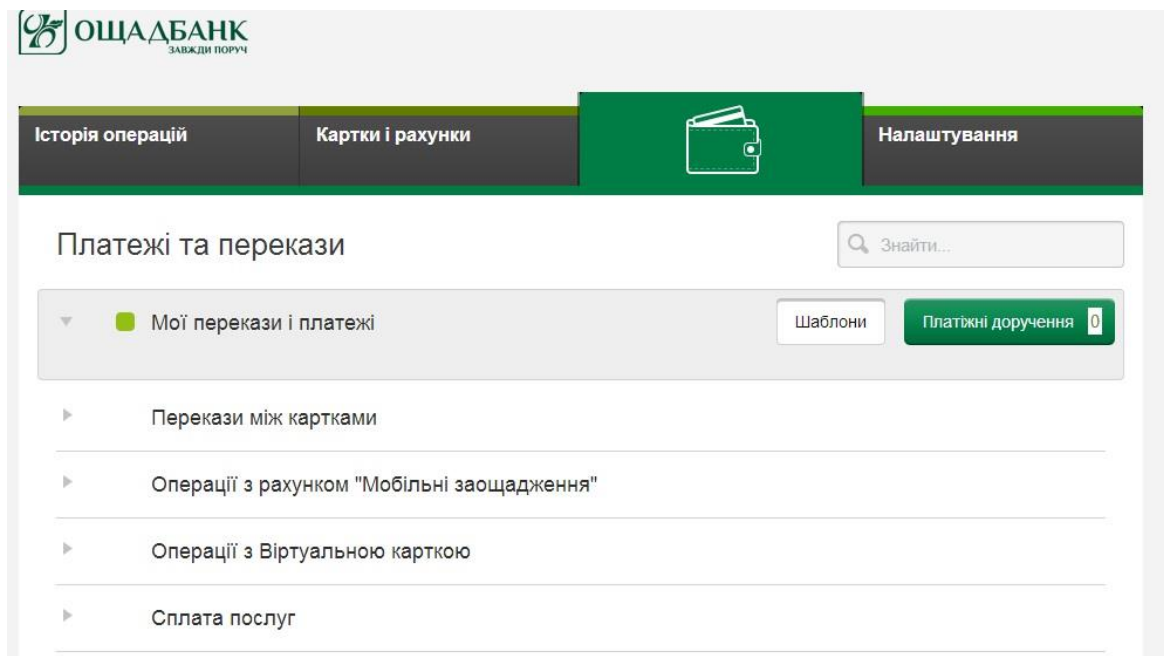


Рисунок 1.2 - Інтерфейс онлайн банкінгу «Ощад24»

#### Система «UKRSIB online» та «StarAccess» [5]

Один з найголовніших банків України теж запровадив систему дистанційного веб-орієнтованого онлайн банкінгу «UKRSIB online» - для фізичних осіб (рисунок 1.3), а «StarAccess» - для юридичних осіб.

Система побудована на базі новітніх технологій безпеки та багатоварової автентифікації клієнтів, використовує принцип RESTFul сервісів, тобто з мобільного додатку (рисунок 1.4) можна повноцінно користуватись онлайн банкінгом та усіма його функціями. Цікавою особливістю є контроль витрат і надходжень. UKRSIB online в автоматичному режимі проводить аналіз витрат і надходжень користувача, розбиває їх за категоріями та показує в графіках.

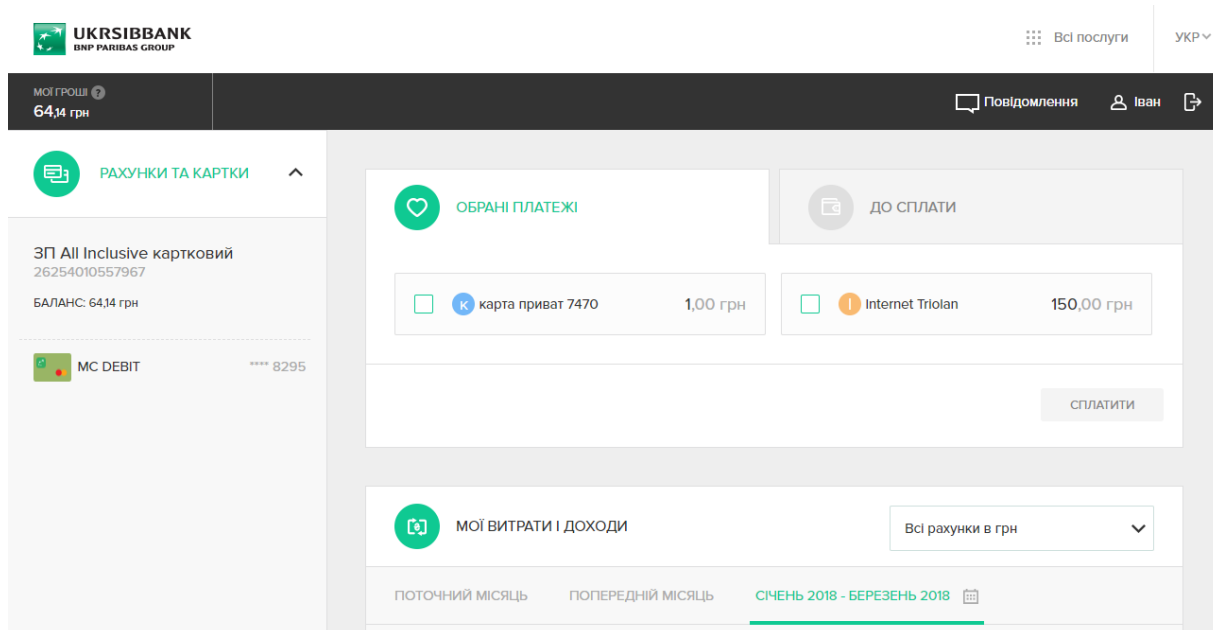


Рисунок1.3 - Інтерфейс онлайн банкінгу «UKRSIB online»

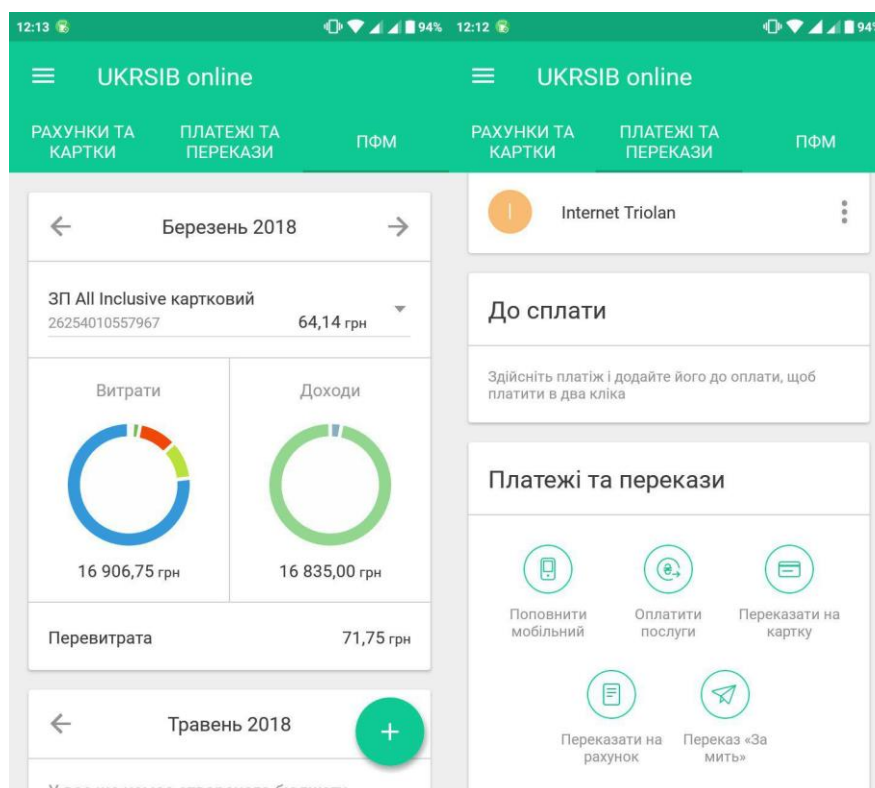


Рисунок 1.4 - Інтерфейс мобільного онлайн банкінгу «UKRSIB online»

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		8



Система онлайн банкінгу StarAccess надає зручний інструмент дистанційного управління коштами для юридичних осіб, відкриваючи перед ними широкий перелік можливостей, високий рівень надійності і гарантією безпеки (рисунок 1.5).

Система передбачає наявність у користувача спеціального брелока з електронним цифровим підписом, який підтверджує особистість при роботі з додатком.

Недоліком є підключення до системи - обов'язково потрібно відвідати філію банку і отримати електронний цифровий підпис. Але система представляє користувачеві стабільну роботу з достатнім набором функцій управління рахунком та високим рівнем безпеки.

Система StarrAccess використовує додаток, розроблений мовою Java для забезпечення максимальної надійності виконаних транзакцій.

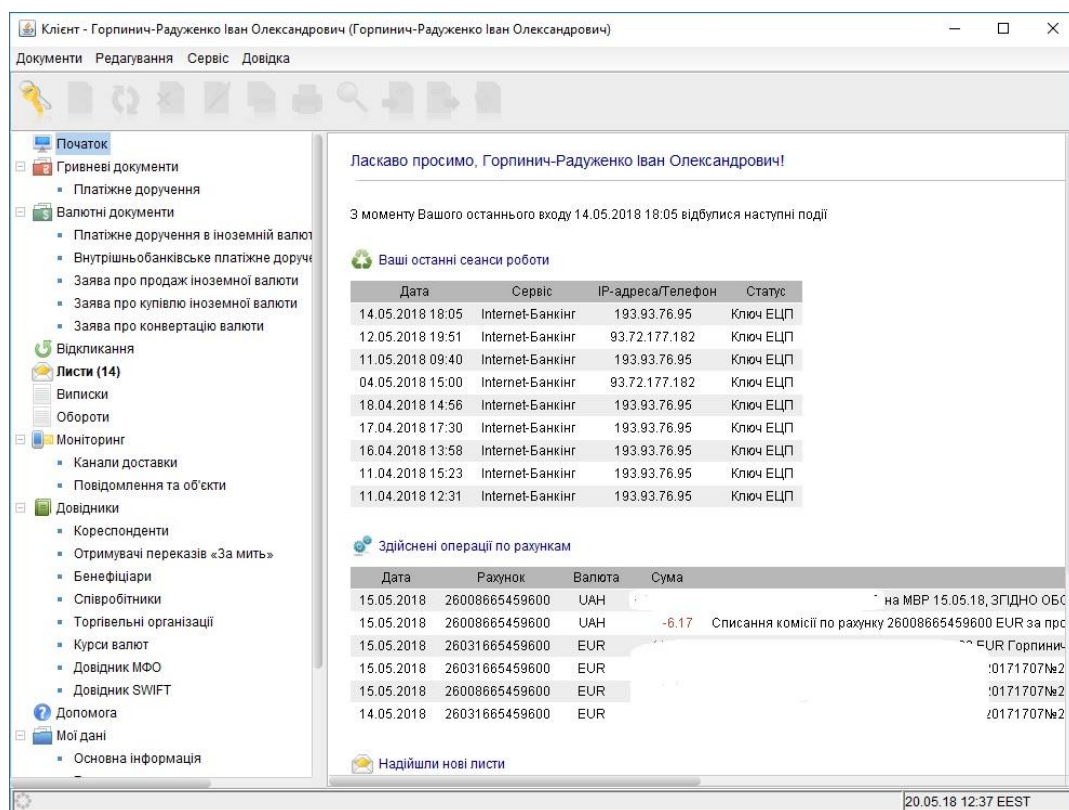


Рисунок 1.5 - Інтерфейс онлайн банкінгу «StarAccess»

Недоліком усіх розглянутих систем онлайн банкінгу є проблема початку роботи з додатком для нових користувачів. Не одна з систем не надає ґрунтовних знань для повноцінного початку роботи. При першому використанні, користувач не знає, які можливості відкриваються перед ним, та які можливі помилки він може виконати, при роботі з електронними рахунками та електронним грошовим оборотом. За рахунок того, що більшість можливостей роботи з банком перенесені у додаток, та виключається можливість втручання спеціаліста банку, який може підказати та розказати про переваги та недоліки використання різноманітних функцій, таких як відкриття депозитних рахунків, користування кредитними коштами тощо. Тому у кожного користувача з'являється можливість зробити неправильні дії зі своїми грошима, які можуть призвести до незворотних маніпуляцій. Наприклад, відкриття депозитного рахунку, передбачає виключення можливості знаття коштів на період часу активності депозиту.

Кожна з розглянутих систем не надає «навчального майданчику», для попереднього пробного використання функцій без використання реального еквіваленту грошей, для запобігання ризику втрати або неправильного розпорядження своїми грошовими збереженнями.

## 1.2. Безпека при виконанні дистанційних платіжних операцій

В онлайн банкінгу, як і в традиційних банківських системах, безпека є основною проблемою. Розробники спеціалізованого програмного забезпечення намагаються вживати всі необхідні запобіжні заходи, щоб переконатися, що інформація клієнтів передається безпечно та надійно. Найновіші методи в безпеці онлайнної банківської системи використовуються для збільшення моніторингу цілісності та безпеки системи.

					ІАЛЦ.467200.004 ПЗ	Арк.
						10
Змін.	Арк.	№ докум.	Підпис	Дата		

Захист онлайн банкінгу повинен розглядається на трьох рівнях. Перша проблема полягає в безпеці інформації про клієнтів, оскільки вона надсилається з персонального пристрою клієнта на веб-сервер. Друга частина стосується безпеки навколишнього середовища, в якому розташована мережа банківських серверів та база даних клієнтів. Нарешті, вживаються заходи безпеки, щоб запобігти входженню неавторизованим користувачам входити в розділ онлайн банкінгу веб-сайту [6].

### 1.2.1 Шифрування у банківських системах

Захист конфіденційності повідомлень між браузером та серверами банку забезпечується за допомогою шифрування. Шифрування відбувається наступним чином: коли користувач переходить на сторінку входу онлайн банкінгу, браузер встановлює безпечний сеанс на сервері користувача. Безпечний сеанс встановлюється за протоколом SSL (рівень захищених сокетів).

Цей протокол вимагає обміну між сервером та клієнтським додатком публічними та приватними ключами. Ключі - випадкові числа, вибрані для цього сеансу, і вони використовуються лише між браузером користувача та сервером банку. Після обміну ключами браузер використовуватиме їх для розбиття (шифрування) повідомлень, надісланих від браузера до сервера. Обидві сторони вимагають ключів, оскільки їм потрібно розшифрувати повідомлення після їх отримання. Протокол SSL не тільки забезпечує конфіденційність, але також гарантує, що жоден інший веб-користувач не зможе «видати себе» за іншого користувача, а також не змінить жодну надіслану інформацію.

					ІАЛЦ.467200.004 ПЗ	Арк.
						11
Змін.	Арк.	№ докум.	Підпис	Дата		

### 1.2.2. Рівень захищених сокетів SSL

Це стандартна технологія безпеки для встановлення зашифрованого зв'язку між сервером і клієнтом, як правило, веб-сервером (веб-сайтом) та веб-переглядачем, або поштовим сервером і поштовим клієнтом (наприклад, Outlook).

SSL дозволяє безпечно передавати конфіденційну інформацію, таку як номери кредитних карток, номери соціального захисту та облікові дані для входу. Зазвичай, дані, що надсилаються між веб-переглядачами та веб-серверами, надсилаються звичайним текстом, залишаючи користувача вразливим до прослуховування та крадіжки даних. Якщо злоумисник може перехопити всі дані, що надсилаються між веб-переглядачем та веб-сервером, він може бачити та використовувати цю інформацію.

Більш конкретно, SSL - це протокол безпеки. Протоколи описують, яким чином слід використовувати алгоритми. У цьому випадку протокол SSL визначає змінні шифрування як для прийнятих, так і для переданих даних.

Всі веб-браузери мають можливість взаємодіяти з захищеними веб-серверами за допомогою протоколу SSL. Однак браузеру та серверу потрібно те, що називається SSL-сертифікатом, щоб мати змогу встановити безпечне з'єднання.

SSL захищає мільйони даних людей в Інтернеті щодня, особливо під час онлайн-транзакцій або передачі конфіденційної інформації. Користувач може визначити, чи знаходиться його браузер у безпечному режимі, побачивши символ захисту у вікні веб-браузера (рисунок 1.6).

					ІАЛЦ.467200.004 ПЗ	Арк.
						12
Змін.	Арк.	№ докум.	Підпис	Дата		

Рисунок 1.6 - Приклад відображення захищеного режиму

### 1.2.3. Двофакторний метод автентифікації (2FA)

З кожним днем зростає кількість способів отримання доступу до приватних даних користувача при використанні стандартних процедур безпеки, що вимагають лише просте ім'я користувача та пароль. Шахраї намагаються отримати особисті та фінансові дані, для подальшого використання цієї інформації для здійснення шахрайських дій, загалом фінансового характеру.

Двофакторна автентифікація, також відома як 2FA, є додатковим рівнем безпеки, яка вимагає не тільки пароля та імені користувача, але й іншу додаткову унікальну інформацію, яку користувач повинен відразу пред'явити – це може бути інформація, яка надходить до телефону користувача у вигляді SMS, але це також може бути USB-ключ (апаратний токен) або інший пристрій, який може генерувати одноразові коди.

Користувач, який використовує сервіс з двофакторною автентифікацією, зазвичай надає свій номер мобільного телефону (також може використовуватися мобільний додаток, але це найпоширеніший спосіб використання двофакторної автентифікації.). Більшість форм двофакторної автентифікації вимагають входу за допомогою імені користувача та пароля, а потім введення коду, який надіслано через SMS. Цей метод не лише підтверджує, що користувач знає базові дані для авторизації (ім'я користувача та пароль), але і те, до чого має доступ лише він (мобільний телефон), номер якого зареєстрований як пристрій для отримання цих кодів.

Використання імені користувача та пароля разом з інформацією, яку знає лише користувач, ускладнює отримання потенційними злоумисникам доступу та викрадення особистих даних користувача.

Історично склалося так, що двофакторна автентифікація - це не нова концепція, але її використання стало більш поширеним на сьогоднішній день. У лютому 2011 року Google оголосила про двофакторну автентифікацію для доступу до своїх сервісів, потім до них приєдналися такі гіганти як MSN і Yahoo, Microsoft [7].

Використання процедури автентифікації за двома факторами може допомогти зменшити кількість випадків крадіжки особистих даних в Інтернеті, а також фішингу через електронну пошту, оскільки злочинцям потрібно більше, ніж просто дані про ім'я користувача та пароль (рисунок 1.7).

Недоліком цього процесу безпеки є те, що нові апаратні токени (у вигляді ключів або кард-рідерів) мають бути замовлені, а потім видані користувачеві, і це може призвести до уповільнення процесу початку роботи з сервісом, щоб отримати доступ до своїх приватних даних за допомогою цієї процедури автентифікації. Токени також зазвичай мають невеликі габарити, тому легко втрачаються, що викликає більше проблем для оформлення нової заяви для виробництва нового ключа.



Рисунок 1.7 - Приклад програмного токена для виконання дворівневої автентифікації у системі онлайн банкінгу «StarAccess»

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		14

### 1.3. Обґрунтування теми проекту та обраних засобів розроблення

Дана розробка є актуальною, оскільки виявлений високий рівень розвитку технологій у сфері дистанційних банківських операцій. Проте відсутність фінансової освіченості призводить до цифрового розриву та віддалення людей малого та середнього фінансового достатку від використання усіх можливих переваг онлайн банкінгу. Тому є необхідність у розробці веб-орієнтованої банківської системи операцій користувача для здобуття необхідних навичок використання цих систем.

Для розробки системи мною була обрана мова високого рівня Java. Технології мови Java включають великий набір API-інтерфейсів, інструментів і реалізацій які використовують алгоритми безпеки, механізми і протоколи шифрування. Програмний інтерфейс модуля безпеки Java охоплює широке коло областей, включаючи криптографію, інфраструктуру відкритих ключів, захищений зв'язок, автентифікацію та контроль доступу. Технологія безпеки Java надає розробнику всеохоплюючу основу для написання додатків, а також надає користувачеві або адміністратору набір інструментів для безпечного управління програмами.

					ІАЛЦ.467200.004 ПЗ	Арк.
						15
Змін.	Арк.	№ докум.	Підпис	Дата		

## 2. АНАЛІЗ ІНСТРУМЕНТІВ РОЗРОБКИ ПРОГРАМНОГО ТА АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ

### 2.1. IntelliJ IDEA, SAP PowerDesigner та MySQL Workbench

Розглянемо використані інструменти розробки:

#### IntelliJ IDEA

IntelliJ IDEA - це інтегроване середовище розробки (IDE) для розробки мовою високого рівня Java. Використовується для розробки комп'ютерного програмного забезпечення. Вона пропонує внутрішній аналіз, мультимедійний синтаксис, автодоповнення, історію змін, інтегроване підключення до бази даних, інтегроване рішення для контролю версій, можливість під'єднання серверу для автоматичного завантаження веб-проекту, підключення додатків для підтримки інших мов програмування.

IntelliJ IDEA відрізняється від аналогів простотою використання, гнучкістю та надійним дизайном. При використанні IntelliJ IDEA, розробник отримує інтегроване середовище розробки, але з великою кількістю доповнень, які роблять розробку набагато простішою і приємнішою.

Приведемо декілька з них:

- вкладка автодоповнення (імена класів, функції, методи, змінні);
- чіткіші повідомлення про помилки і їх виділення кольором;
- аналіз потоку даних у реальному часі;
- гарна інтеграція з модулями (GIT, Maven, TypeScript);
- розумний рефакторинг коду;
- виявлення дублікатів;

Особливістю IntelliJ IDEA є підтримка Spring Integration Framework - розширення для веб-програмування, що підтримує моделі корпоративної інтеграції.

					ІАЛЦ.467200.004 ПЗ	Арк.
						16
Змін.	Арк.	№ докум.	Підпис	Дата		



Недоліком цієї системи є платний доступ для використання. Але кожен студент має можливість зробити підписку у навчальних цілях.

### SAP PowerDesigner

SAP PowerDesigner – це програмний додаток для спільного інструментального моделювання архітектури програмного забезпечення та баз даних.

За допомогою цього додатку з'являється можливість швидкого проектування як самої архітектури програмного забезпечення, для наглядного і простого способу протоколювання механізмів взаємодії окремих програмних модулів, так і проектування моделей бази даних, не прив'язуючись до конкретної реалізації обраної системи бази даних, з подальшою можливістю компілювання логічних та концептуальних схем у скрип виконання певної обраної системи управління базою даних.

PowerDesigner підтримує дизайн моделей архітектури програмного забезпечення. PowerDesigner зберігає моделі за допомогою різноманітних розширень файлів. Внутрішня структура файлу може бути або XML, або у форматі, бінарного файлу. PowerDesigner також може зберігати моделі в сховищі бази даних.

PowerDesigner підтримує побудову таких видів моделей, як:

- модель вимог (RQM), яка допомагає проаналізувати будь-які письмові вимоги та пов'язати їх;
- модель корпоративної архітектури (EAM), яка допомагає аналізувати та документувати бізнес-функції;

					ІАЛЦ.467200.004 ПЗ	Арк.
						17
Змін.	Арк.	№ докум.	Підпис	Дата		

- концептуальна модель (CDM), що допомагає проаналізувати концептуальну структуру інформаційної системи;
- об'єктно-орієнтована модель (OOM), що допомагає проаналізувати інформаційну систему через об'єкти використання;
- логічна модель даних (LDM), яка допомагає проаналізувати структуру інформаційної системи;
- фізична модель даних (PDM), що допомагає проаналізувати таблиці, представлення даних та інші об'єкти в базі даних.

PowerDesigner має підтримку:

- автогенерації коду;
- моделювання даних (працює з більшістю основних систем RDBMS );
- моделювання складу даних ( WarehouseArchitect );
- моделювання об'єкта ( діаграми UML);
- аналіз вимог;
- XML моделювання - схеми і DTD стандарти.

Недоліком є можливість неспівпадіння типів даних при автоматичній генерації коду побудованої архітектури бази даних для конкретно обраної мови програмування.

### MySQL Workbench

MySQL Workbench - це візуальний інструмент розробки баз даних, який інтегрує розробку, адміністрування, створення баз даних, створення та підтримку SQL запитів в єдине інтегроване середовище розробки для бази даних.

MySQL Workbench надає розробникам повний набір візуальних інструментів для створення, редагування та управління SQL запитамі,

					ІАЛЦ.467200.004 ПЗ	Арк.
						18
Змін.	Арк.	№ докум.	Підпис	Дата		

з'єднаннями з базами даних та об'єктами.

Візуальний редактор SQL дозволяє розробникам створювати, редагувати та виконувати запити, створювати та редагувати дані, а також переглядати та експортувати результати.

MySQL Workbench надає такі можливості, як:

- підсвічування синтаксису;
- контекстно-конфіденційна довідка;
- автозаповнення;
- налагодження SQL-інструкцій.

Програма надає набір інструментів для підвищення продуктивності додатків.

За допомогою інтегрованого модулю аналізу бази даних можливо швидко переглянути основні показники ефективності, використовуючи інформаційну панель продуктивності. Звіти ефективності забезпечують легку ідентифікацію високоякісних SQL-операторів тощо. Крім того, розробники можуть бачити, де оптимізувати свій запит за допомогою покращеного та простого у використанні плану підказок.

## 2.2 Характеристики веб-серверу

Плануючи серверні вимоги до обладнання для розгортання веб-серверу, буде необхідно оцінити масштабованість серверів на основі пікових відвідувачів, співвідношення максимальної кількості користувачів до пікового значення одночасних підключень та загальної навантаженості систем бізнес логіки.

Коефіцієнт перегляду – відношення кількості відвідувачів які виконують оновлення, до тих, що переглядають лише вміст сторінки.

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		19

Загальний вміст системи краще всього оцінювати за рахунком загальної пропускної спроможності.

Стійкість системи найкраще оцінюється за умови постійного потоку одночасних користувачів.

Значення, наведені нижче, стосуються мінімального наявного устаткування, необхідного для запуску Apache Tomcat. Наприклад, мінімальний розмір стека становить 250 МБ. Адміністратору системи знадобиться додаткове фізичне обладнання, принаймні мережева карта, необхідна будь-яким додаткам, що працюють на сервері.

Мінімальна рекомендація щодо обладнання:

- процесор Intel Core 2 Duo 2.40 ГГц або вище;
- оперативна пам'ять обсягом 2 ГБ або більше;
- 64-розрядна операційна система;
- жорсткий диск не менше 1 ГБ.

У особливих випадках навантаження на сервер перш за все залежить від пікових відвідувачів, тому мінімальні системні вимоги важко оцінити. Ці цифри надаються як керівництво до абсолютного мінімуму, необхідного для запуску сервера, і для конкретної конфігурації, швидше за все, буде потрібно обладнання з кращими характеристиками.

Провівши навантажувальне тестування та проаналізувавши необхідні характеристики для запуску веб-серверу, були знайдені приблизні показники технічних характеристик для можливих випадків одночасного виконання операцій (таблиця 2.1). Зазначимо, що обсяг оперативної пам'яті стосується або загальної пам'яті сервера, або пам'яті, що виділяється для запуску віртуальної машини Java.

					ІАЛЦ.467200.004 ПЗ	Арк.
						20
Змін.	Арк.	№ докум.	Підпис	Дата		

Таблиця.2.1 - результати навантажувального тестування

Операції	Процесори	Процесор (ГГц)	ОЗУ (МБ)
150	1	2.6	1024
250	2	2.8	1,536
1000	4	3	2,048
4 000	2	3.8	2,048
6 000	2	3.6	4,096
8 000	2	3.6	4,096
10 000	4	2.6	4,096

### 2.3. Характеристики серверу баз даних

Система пред'являє наступні вимоги до апаратного та програмного забезпечення сервера:

Мінімальні вимоги до сервера:

- 2-х ядерний процесор з тактовою частотою 2 ГГц або вище;
- оперативна пам'ять обсягом 4 ГБ;
- один завантажувальний диск не менше 1 ГБ;
- 32-х розрядна операційна система.

Рекомендовані вимоги до сервера:

- 4-х ядерний процесор або більше;
- оперативна пам'ять обсягом 8 ГБ або більше;
- мінімум два логічних дискових елемента під'єднаних за технологією RAID10;
- один завантажувальний диск не менше 1 ГБ;
- 64-х розрядна операційна система.

Існує також можливість інсталяції системи на платформі Windows. При цьому вимоги до попередньо програмного забезпечення залишаються незмінними.

Для роботи з клієнтською частиною системи потрібен комп'ютер, підключений по протоколу TCP/IP до мережі, в якій знаходиться. Система пред'являє наступні вимоги до апаратного та програмного забезпечення сервера.

### 2.3.1 InnoDB

InnoDB - це ядро загальної обробки даних, яке забезпечує високу надійність та високу продуктивність. У MySQL, InnoDB є ядром обробки за замовчуванням. Розглянемо основні особливості:

- ядро InnoDB організовує таблиці даних на диску для оптимізації запитів на основі первинних ключів (PRIMARY KEY). У кожній InnoDB таблиці є індекс первинного ключа, який називається кластерним індексом, який організовує дані для мінімізації введення/виводу та пошуку первинних ключів, використовуючи алгоритм «B-tree»;
- для збереження цілісності даних, InnoDB підтримує зовнішні ключі (FOREIGN KEY). Вставки, оновлення та видалення перевіряються за зовнішніми ключами, щоб попередити загублення даних у пов'язаних таблицях.

### 2.3.2. RAID10

RAID (надлишковий масив незалежних дисків) - конфігурація жорстких дисків, яка використовує методи розмітки, дзеркальне відображення або парності для створення великих надійних сховищ даних з декількох загального призначення комп'ютерних накопичувачів на жорстких магнітних дисках (HDD).

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		22

RAID10 - гібридний RAID, використовується для отримання додаткової продуктивності, зменшення навантаження на носії, досягання більшої захищеності даних, в результаті об'єднання властивостей двох стандартних моделей підключення RAID (рисунок 2.1).

RAID 10, як правило, реалізується RAID-контролерами, та представляє собою масив дзеркальних сховищ типу RAID 0, який може являти собою двосторонні та тристоронні дзеркала та вимагає щонайменше чотирьох дисків. Також можливі масиви більше чотирьох дисків.

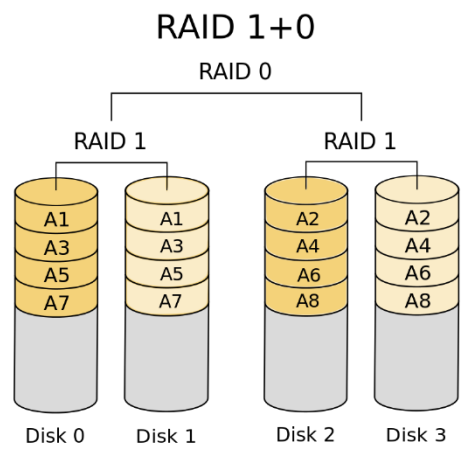


Рисунок 2.1 - Типова конфігурація RAID10

### 3. РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

#### 3.1. Розробка концепції рішення

##### Вибір мови програмування

Для розробки дипломного проекту мною була обрана мова високого рівня Java, тому що технології мови Java включають у себе [8]:

- API-інтерфейсів;
- інструментів і реалізацій часто використовуваних алгоритмів безпеки;
- механізмів протоколювання.

Java – це об’єктно-орієнтована мова програмування, тому ця парадигма має на увазі використання так званих «об’єктів», для реалізації будь яких зв’язків та операцій, які теж відбуваються безпосередньо на самих об’єктах, змінюючи їх стан.

При розробці бакалаврського проекту використовувалося API безпеки Java, тому що воно надає можливість використання широкого кола інструментів безпеки, таких як криптографію, інфраструктуру відкритих ключів, захищений зв’язок, автентифікацію та контроль доступу. Технологія безпеки Java надає всеохоплюючу основу для написання додатків, а також надає набір інструментів для безпечного управління програмою.

При виборі мови програмування для розробки системи онлайн банкінгу була обрана Java, тому що вона має можливість кросс-платформеного розгортання розробленого програмного забезпечення на будь якому апаратному інтерфейсі, для якого розроблена так звана «віртуальна машина», яка транслює компільований байткод, незалежно від конкретної реалізації центрального процесора (наприклад, у мові C, написаний код кожен раз компілюється відносно

					ІАЛЦ.467200.004 ПЗ	Арк.
						24
Змін.	Арк.	№ докум.	Підпис	Дата		



системи центрального процесора, і тому робота програми на різних пристроях може відрізнятись та приводити до важко виявляємих помилок).

Необхідний при виборі мови розробки підхід, описує парадигма мови Java: «Написана програма один раз, запускається на будь якій системі однаково (Write ones - run everywhere)».

Віртуальна машина Java (JVM) являє собою віртуальну систему, яка дозволяє комп'ютеру працювати Java - програмам, а також програмам, написаних на інших мовах і інтерпретованих на Java байткод (рисунок 3.1). JVM детально визначається специфікацією, яка формально описує, що вимагається від впровадження JVM. Наявність специфікації забезпечує сумісність програм Java з різними реалізаціями, так що авторам програм не потрібно турбуватися про особливі схеми базової апаратної платформи.

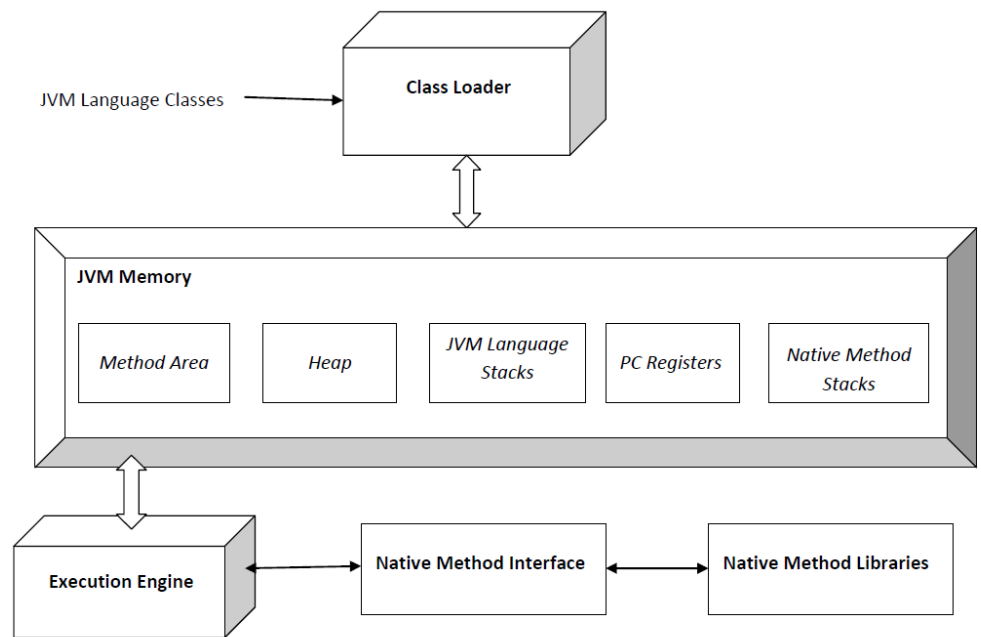


Рисунок 3.1 - Огляд віртуальної машини Java

За допомогою цих інструментів підвищується можливість абстрагування при розробці програмного забезпечення та більших можливостей при обробці великих масивів даних, використовуючи лише декілька команд.

Наприклад, при розробці сервісу для роботи з користувачами, даний функціонал допомагав перебирати велику кількість користувачів, для знайдення необхідного запису у базі даних, та перевірки правильності введеного паролю.

### 3.1.1 Етапи розробки системи банківських операцій користувача

Система, що розробляється в даному бакалаврському проєкті, передбачає реалізацію наступних кроків:

- розробка концепції об'єктів, які будуть використовуватися;
- вибір системи управління бази даних, яка задовольняє потребам;
- налаштування багатопоточного з'єднання з базою даних;
- розробка реляційної бази даних;
- розробка архітектури об'єктів DAO, за допомогою яких буде відбуватися спілкування програмного інтерфейсу конкретної програмної реалізації з інтерфейсами обраної бази даних;
- розробка трансферних об'єктів, які обробляють запити від бази даних, та створюють безпосередньо моделі об'єктів, для подальшої роботи з ними;
- реалізація сервісних об'єктів, які будуть виконувати конкретні зміни над об'єктами, які передбачені необхідністю виконання бізнес логіки;
- реалізація командних об'єктів, за допомогою яких буде виконуватись виклик заздалегідь обраного необхідного сервісу;
- розробка контролера, який буде обробляти запити від клієнта, та ставити на виконання необхідні команди;

					ІАЛЦ.467200.004 ПЗ	Арк.
						26
Змін.	Арк.	№ докум.	Підпис	Дата		

- розроблення контролера, який буде контролювати правильне відображення клієнтської частини;
- вибір та імплементація серверного програмного забезпечення;
- розробка моделі відображення (клієнтської частини) яка буде задовольняти потребам бізнес логіки.

Розглянемо кожен етап розробки більш детально.

Розробка концепції об'єктів програмних сутностей.

Об'єктно-орієнтоване програмування — одна з парадигм програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Тому для обраної концепції потрібно спроектувати (розробити) об'єкти, які будуть задовольняти потребам програмної реалізації та в повній мірі будуть описувати необхідну інформацію про сутності програми та їх взаємодію один на одного.

Для цього було розроблено об'єкти описання реальних сутностей з необхідними властивостями для конкретної задачі:

- об'єкт користувача, який містить у собі поля: ім'я, прізвище, електронної пошти, номера телефону, паролю та ролі;
- об'єкти рахунків, які містять у собі поля: номер рахунку, тип, ідентифікаційний номер власника, статусу та балансу. Для кодної окремої реалізації рахунків (депозитний, кредитний, дебетовий) були додані необхідні поля які властиві лише цьому виду рахунків. Це зроблено для запобігання надлишковості інформації, та для економії пам'яті у системі;
- об'єкти карток, які містять у собі поля: номер картки, ідентифікаційний номер рахунку, до якого прив'язана картка, пін-код, cvv, дата закінчення терміну дії картки, тип картки;

					ІАЛЦ.467200.004 ПЗ	Арк.
						27
Змін.	Арк.	№ докум.	Підпис	Дата		

- додаткові об'єкти типу (роль, статус, банківська операція, тип, кредитний запит).

На рисунку 3.2 представлено UML схему залежності розроблених сутностей.

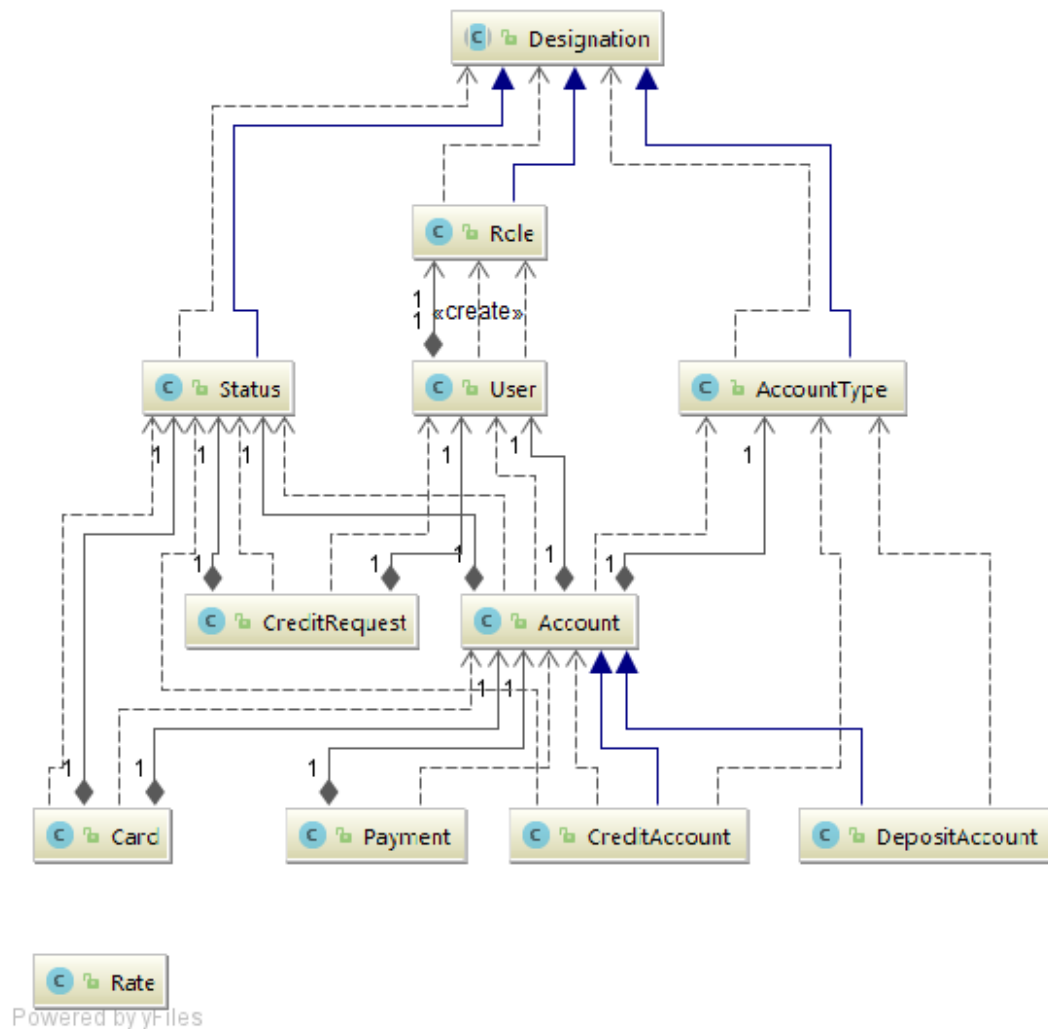


Рисунок 3.2 – UML схема залежності розроблених сутностей

Для кожного створеного об'єкта були створені методи взаємодії з їх полями та роботи з ними. Одною з особливостей розробки об'єктів програмної сутності, є використання шаблону проектування «Будівельник».

Часто об'єкти можуть бути складними, і їх створення вимагає виконання цілого набору операцій складання з простіших об'єктів. При цьому може знадобитися використовувати різні імплементації цих простих об'єктів або алгоритм складання може бути різним. Для відділення створення об'єкта від його внутрішніх деталей використовується шаблон «Будівельник» (Builder) [9].

У шаблоні Будівельник використовується алгоритм збірки і вибір реалізації частин об'єкта всередині класу конкретного будівельника, що реалізує базовий інтерфейс. Залежно від того, який саме алгоритм збірки потрібно, вибирається конкретна імплементація.

Також при розробці даних об'єктів було дотримано принципу успадкування — механізму утворення нових класів на основі використання вже існуючих. При цьому властивості та функціональність батьківського класу переходять до класу нащадку (дочірнього).

### Вибір системи управління бази даних

Незважаючи на те, що всі системи управління базами даних виконують одну і ту ж основну задачу (тобто дають можливість створювати, редагувати і отримувати доступ до інформації, що зберігається в базах даних), сам процес виконання цього завдання варіюється в широких межах. Крім того, функції і можливості кожної СУБД можуть істотно відрізнятися. Різні СУБД документовані по-різному: більш-менш ретельно. По-різному надається і технічна підтримка.

При порівнянні різних популярних баз даних, було враховано, чи зручна для розробки і масштабування дана конкретна СУБД, а також переконалися, що вона

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		29

буде добре інтегруватися з іншими інтерфейсами, які вже використовуються. Крім того, під час вибору було взято до уваги вартість системи і підтримки, що надається розробником.

Для розробки бакалаврського проекту була обрана система управління базою даних MySQL, яка є найпопулярнішою у світі базою даних з відкритим кодом. Завдяки своїй перевірненій продуктивності, надійності та простоті використання, MySQL є провідним вибором баз даних для веб-застосунків, що використовуються високопрофесійними веб-ресурсами.

MySQL - це система управління базами даних. У реляційних базах даних дані зберігаються не всі разом, а в окремих таблицях, завдяки чому можна досягти підвищення швидкості і гнучкості. Таблиці мають зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL як частина системи MySQL можна охарактеризувати як мову структурованих запитів плюс найбільш поширений стандартний мова, яка використовується для доступу до баз даних.

При розробці СУБД була налаштована система безпеки доступу до бази даних. Були задані ролі для звичайного зв'язку програмного інтерфейсу, який може змінювати лише дані у таблицях, та роль адміністратора, якому надаються можливості зміни архітектури таблиць та взаємозв'язків у таблицях.

Система безпеки заснована на привілеях та паролях з можливістю верифікації з віддаленого комп'ютера, за рахунок чого забезпечується гнучкість і безпека. Паролі при передачі по мережі при з'єднанні з сервером шифруються. Клієнтські інтерфейси з'єднуються з MySQL, використовуючи сокети TCP/IP, сокети Unix або іменовані канали.

При розробці СУБД MySQL була використана трирівнева структура: бази даних - таблиці - записи. Логічно - таблиця являє собою сукупність записів. А записи - це сукупність полів різного типу. Ім'я бази даних MySQL унікально в

					ІАЛЦ.467200.004 ПЗ	Арк.
						30
Змін.	Арк.	№ докум.	Підпис	Дата		

межах системи, а таблиці - в межах бази даних, поля - в межах таблиці. Один сервер MySQL може підтримувати відразу кілька баз даних, доступ до яких може розмежовуватися логіном і паролем. Знаючи ці логін і пароль, можна працювати з конкретною базою даних. Наприклад, можна створити або видалити в ній таблицю, додати записи і тощо. Зазвичай ім'я-ідентифікатор і пароль призначаються хостинг провайдерами, які і забезпечують підтримку самої бази даних.

#### Налаштування багатопоточного з'єднання з базою даних

При розробці програмного забезпечення завжди виникає необхідність поєднання розробленого інтерфейсу та обраної реалізації системи управління базою даних. У мові Java розроблений API для рішення цієї проблеми, так званий JDBC.

Для налаштування роботи було використано Java Database Connectivity (JDBC) - це інтерфейс прикладного програмування (API) для мови програмування Java, який визначає, як клієнтський інтерфейс може отримати доступ до бази даних. Це технологія доступу до даних на базі Java, яка використовується для підключення бази даних до реалізованого інтерфейсу на мові Java.

Для його налаштування достатньо лише задати параметри необхідної бази даних (адреса, логін, пароль, ім'я необхідної бази даних), та виконати інтеграцію драйвера конектора, який обирається відповідно до обраної системи управління базою даних.

Для розробки системи було використано конкретну реалізацію з'єднання MySQL для клієнтських додатків, розроблених на мові програмування Java - MySQL Connector/J, драйвера, який реалізує API Java Database Connectivity (JDBC).

Для вирішення проблеми багатопоточного підключення до бази даних, було обрано методологію «сховища з'єднань».

					ІАЛЦ.467200.004 ПЗ	Арк.
						31
Змін.	Арк.	№ докум.	Підпис	Дата		

Сховище з'єднань (Connection Pool) є кешем з з'єднань з базою даних, підтримуваних таким чином, що з'єднання можуть бути повторно використані. Сховища з'єднання використовуються для підвищення продуктивності виконання команд у базі даних. Відкриття та підтримка підключення до бази даних для кожного користувача є дуже дорогим та витрачає ресурси. Після створення з'єднання, воно розміщується в пулі та знову використовується, щоб не встановлювати нове з'єднання. Якщо всі з'єднання зайняті, створюється нове з'єднання і додається до сховища. Використання сховища також зменшує кількість часу, яке програмний інтерфейс повинен чекати, щоб встановити з'єднання з базою даних.

#### Розробка реляційної бази даних

При розробці реляційної бази даних, була необхідність повністю проаналізувати сферу її використання, та інтерпретувати всі аспекти для подальшої її використання.

Розробка архітектури зв'язків - один з найважливіших кроків у розробці. Якщо на цьому етапі не охопити усі подальші аспекти інформативності необхідних програмних сутностей, на наступних етапах розробки програмного забезпечення можуть виникати великі проблеми при реалізації необхідної логіки, а іноді навіть стає неможливим її інтегрування та подальше масштабування програми.

Насамперед, архітектура бази даних повинна максимально повністю повторювати інформативність розроблених заздалегідь сутностей (користувача, рахунків і т.д.), для можливості серіалізації об'єктів, тобто повного збереження стану об'єктів, з якими працює програмне забезпечення, та можливості у подальшому «відтворити його стан».

					ІАЛЦ.467200.004 ПЗ	Арк.
						32
Змін.	Арк.	№ докум.	Підпис	Дата		



Були розроблені наступні таблиці:

- користувачів;
- кредитних рахунків;
- депозитних рахунків;
- дебетових рахунків;
- запитів на відкриття кредитних рахунків;
- банківських карт;
- платежів;
- допоміжні (роль, статус, тип).

Цікавою особливістю є реалізація таблиць для рахунків. При їх розробці було використано принцип успадкування класів, але перенесеного на реляційну модель.

Для цього потрібна створена таблиця для класу рахунків, а атрибути об'єкта (тобто спеціальні поля, які притаманні кожному окремому виду рахунків) зберігаються у спеціальних таблицях, відповідно до кожного спеціалізованого типу рахунків (рисунок 3.3). Такий спосіб реалізації призводить до скорочення надлишковості інформації та можливості роботи з колекціями рахунків як з одним типом.

					ІАЛЦ.467200.004 ПЗ	Арк.
						33
Змін.	Арк.	№ докум.	Підпис	Дата		

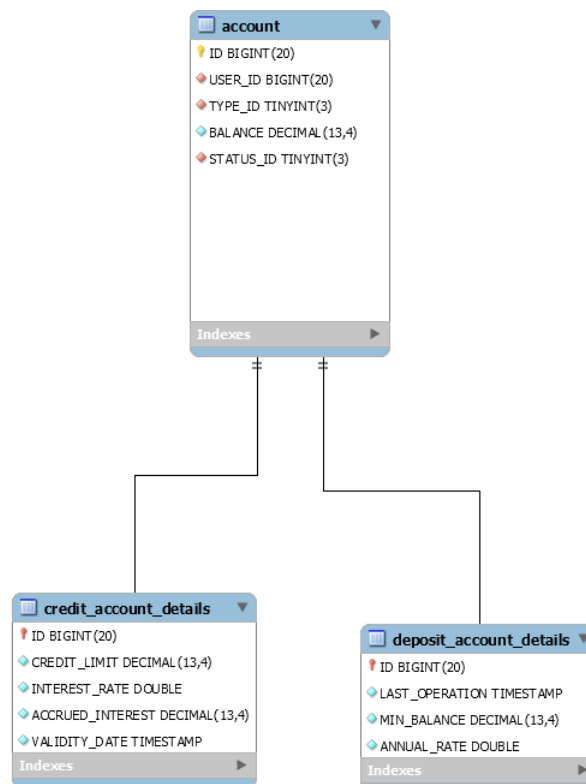


Рисунок 3.3 - Принцип успадкування на реляційній моделі бази даних

Розробка архітектури об'єктів доступу до бази даних на основі шаблону проектування DAO

DAO – Data Access Object (об'єкт доступу до даних) абстрагує і інкапсулює доступ до джерела даних. DAO управляє з'єднанням з джерелом даних для отримання та запису даних [10].

DAO реалізує необхідний для роботи з джерелом даних механізм доступу. DAO повністю приховує деталі реалізації джерела даних від клієнтів. Оскільки при змінах реалізації джерела даних представлений DAO інтерфейс не змінюється, цей шаблон дає можливість DAO приймати різні схеми сховищ без впливу на бізнес-компоненти. По суті, DAO прошарок виконує функцію адаптера між компонентом і джерелом даних.

У інтерфейсі прошарку DAO було реалізовано базові функції, необхідні для роботи з сутностями, які збережені у базі даних, такі як:

- знайти та повернути сутність у базі даних за заданими параметрами;
- завантажити нову сутність у базу даних;
- змінити існуючу сутність за заданими параметрами;
- видалити сутність за заданими параметрами.

На рисунку 3.4 зображена ієрархія основних компонентів розробленого прошарку DAO.

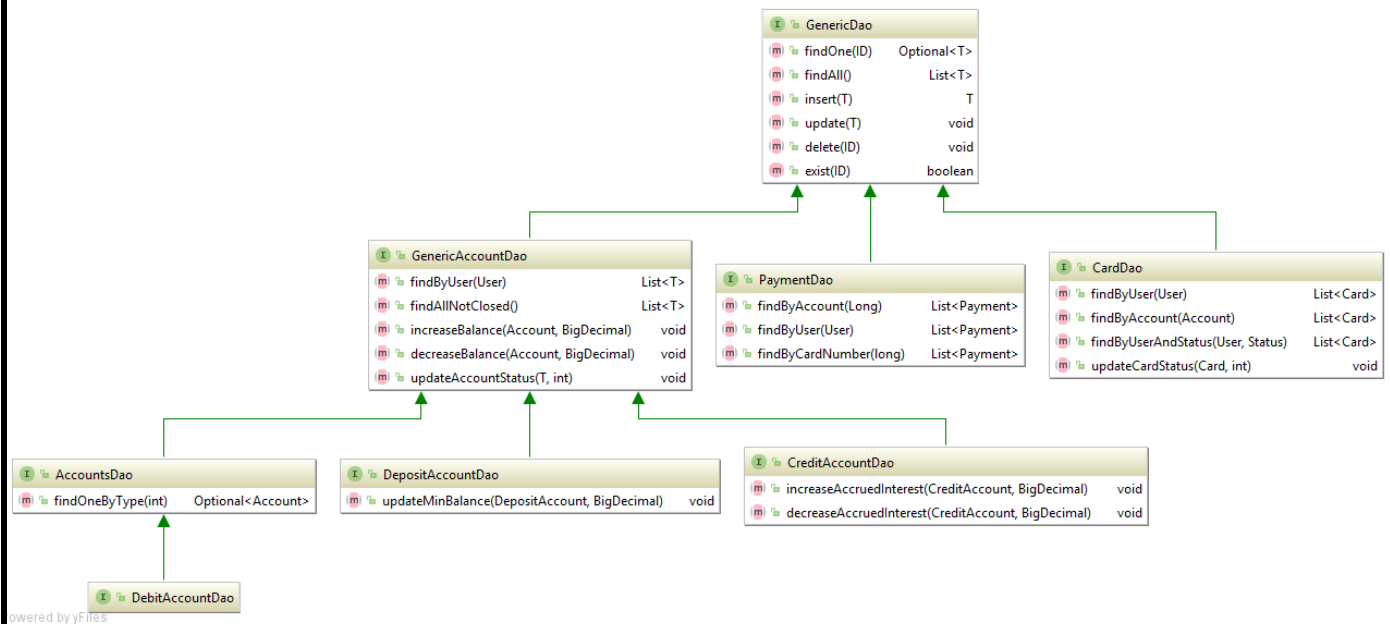


Рисунок 3.4 – Ієрархія основних інтерфейсів DAO

Розробка трансферних об'єктів.

Об'єкти для трансферу даних (Data Transfer Objects) це шаблон проектування який використовується моделювання спеціальних об'єктів для передачі даних між підсистемами розробленого додатку, але які не мають моделі поведінки. Загалом такий спосіб передачі необхідний між рівнем об'єктів DAO та рівнем сервісів, які працюють безпосередньо з сутностями, і їм не важливо,

яким чином ці об’єкти відтворилися у системі.

На рисунку 3.5 представлено приклад взаємодії об’єкту сервісу та об’єкту трансферу даних.

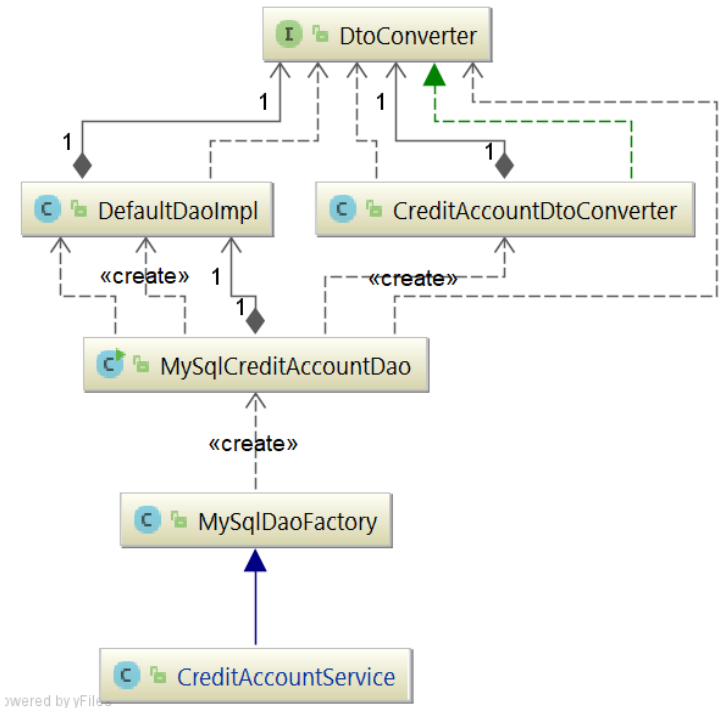


Рисунок 3.5 – Принцип взаємодії об’єктів сервісу з об’єктами трансферу даних

Реалізація об’єктів сервісу.

Об’єкт сервісу (Service object) – це програмна реалізація об’єкту, яка задає спосіб взаємодії клієнта з функціональними можливостями програми.

За допомогою сервісних об’єктів було задано необхідну обмежену поведінку програми та функціонал, який потрібен для виконання операцій описаних потребами бізнес логіки.

При виконанні проекту, були розроблені такі сервіси:

- сервіс роботи з користувачами;
- сервіс для маніпуляцій з кредитними, дебетовими, та депозитними рахунками;
- сервіс для роботи з банківськими картками;
- сервіс для роботи з платіжками;
- сервіс для роботи з кредитними запитами.

На рисунку 3.6 зображені основні об'єкти сервісів, які використовуються у роботі розробленого проекту.

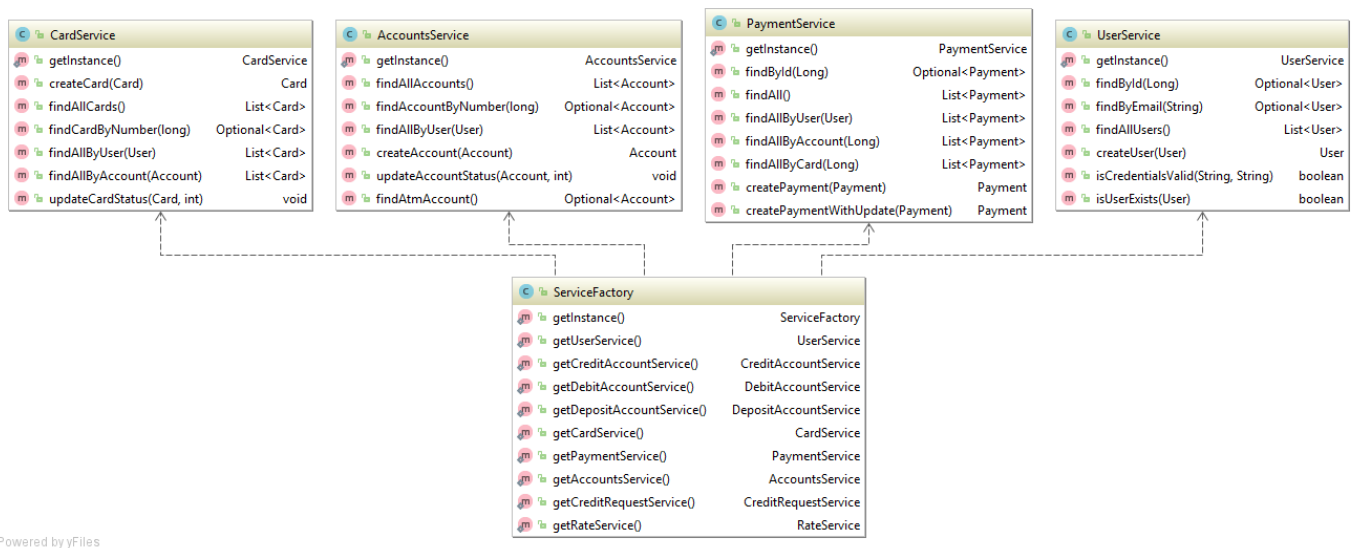


Рисунок 3.6 – Опис основних об'єктних сервісів додатку

## Реалізація об'єктів команд

Об'єкти команд – це програмна реалізація об'єктів, які інкапсулює обробку команд у вигляді об'єкта, що дозволяє зберігати, передавати в якості параметрів методам, а також повертати його у вигляді результату і виконувати інші маніпуляції як і з будь-яким іншим об'єктом.

При проектуванні даного проекту було розроблено 49 об'єктів команд, які виконують різноманітні функції, такі як:

- перевірка інформації про користувача при вході у програму;
- відображення усіх необхідних інформаційних даних, які стосуються безпосередньо користувача (кредитні картки, рахунки, історія);
- можливість створювання нових рахунків;
- переказ грошей між рахунками.

Для ролі менеджера:

- перегляд списку усіх користувачів системи;
- перегляд рахунків конкретного користувача;
- можливість маніпулювання рахунками та картками (закриття та блокування);
- підтвердження або відхилення кредитних запитів.

При виконанні команд, відбувається додаткова перевірка усіх даних на актуальність, які надходять з користувацького інтерфейсу, для запобігання

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		38

навмисної зміни даних у запити на не справжні. Такий метод підвищує завадостійкість системи та захист від помилок.

### Реалізація контролера

Контролер – це програмна реалізація об’єктів, яка реалізує виконання будь-якого об’єкту команди, не залежно від його реалізації.

На контролер надходить спеціальний «флаг» з користувацького інтерфейсу, за допомогою якого він розпізнає та вибирає на виконання заздалегідь підготовлені об’єкти команд, передає їм інформацію яка надійшла з клієнтського інтерфейсу та передає її на виконання команді.

Після її виконання, за допомогою поверненої інформації, контролер вирішує, яку частину візуального інтерфейсу потрібно відобразити користувачеві, для подальшої роботи з системою.

На рисунку 3.7 відображено принцип взаємодії об’єктів контролера, команди, та сервісів.

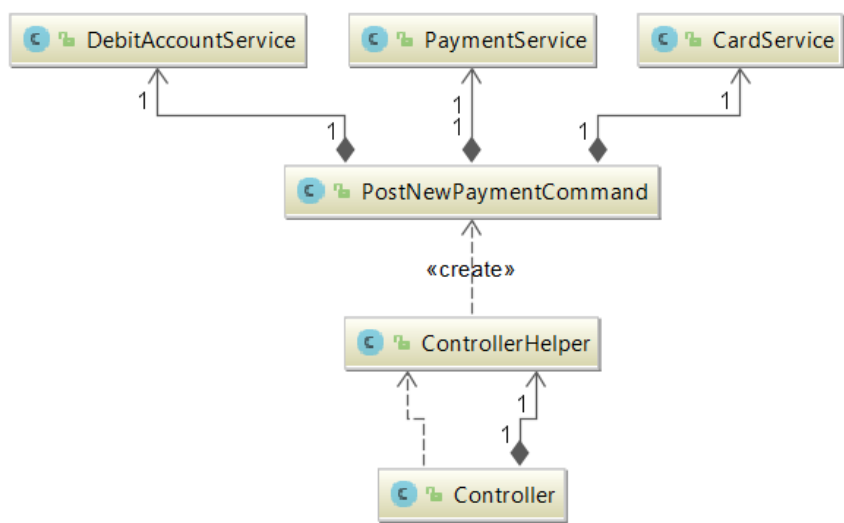


Рисунок 3.7 - Принцип взаємодії об’єктів контролера, команди, та сервісів

Контролер являє собою так званий сервлет - це Java програма, яка розширює можливості сервера. Хоча сервлети можуть відповідати на будь-які типи запитів, в даній розробці використовується сервлет для обробки HTTP запитів.

#### Розробка моделі відображення

При розробці моделі відображення потрібно було врахувати функціональність користувацького інтерфейсу та зрозумілість для нових користувачів.

Для розробки динамічних сторінок була використана технологія JSP.

JavaServer Page (JSP) - це технологія, яка допомагає розробникам програм створювати динамічні веб-сторінки на основі HTML, XML або інших типів документів.

JSP дозволяє Java-коду та деяким заздалегідь визначеним діям переплітатися з вмістом статичного вмісту веб-розмітки, наприклад, HTML, причому результуюча сторінка збирається та виконується на сервері. Скомпільовані сторінки, а також будь-які залежні бібліотеки Java містять Java-байт-код, а не машинний код.

Як і будь-яка інша програма Java, вона повинна бути виконана в рамках віртуальної машини Java (JVM), яка взаємодіє з операційною системою хосту сервера (рисунок 3.8).

З використанням технології JSP, було спроектовано та розроблено усі сторінки користувацького інтерфейсу, такі як:

- головна сторінка;
- сторінка входу до програми;
- сторінка рахунків;
- сторінка переказу коштів;



- сторінка операцій за банківськими картками;
- сторінка історії платежів.

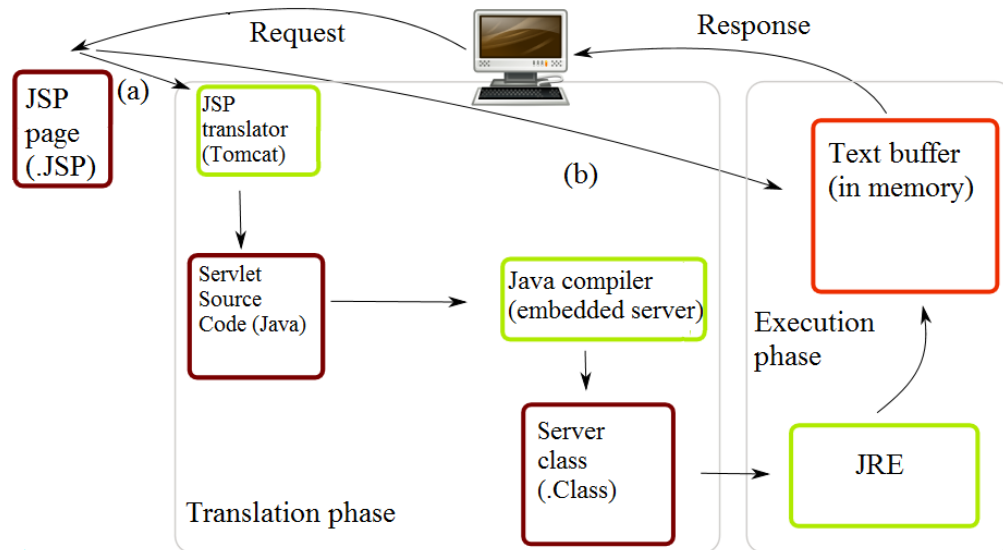


Рисунок 3.8 - Життєвий цикл JSP

### 3.2.Розробка модуля шифрування на основі алгоритму SHA-256

SHA-2 ( Secure Hash Algorithm 2 ) - це набір криптографічних хеш-функцій, розроблений Агентством національної безпеки США (NSA). [11] Вони побудовані з використанням структури Меркле - Дамгарда, з функції одностороннього стиснення, яка сама по собі побудована з використанням структури Девіса-Майєра з (класифікованого) спеціалізованого блочного шифру.

Криптографічні хеш-функції - це математичні операції, що виконуються на цифрових даних; шляхом порівняння обчисленого «хеш» числом з відомим і очікуваним значенням хешу, людина може визначити цілісність даних.

Ключовим аспектом криптографічних хеш-функцій є їх стійкість до колізій: ніхто не повинен мати змогу знаходити два різних вхідних значення, що призводять до того ж вихідного значення.

Даний метод шифрування використовується у розробленій системі для шифрування найважливіших користувацьких даних, таких як логін і пароль, необхідні для роботи у додатку.

Такий підхід використовується для запобігання можливого інформаційного витіку з самої системи, та можливої крадіжки даних користувача адміністраторами системи.

Для підвищення працездатності та пропускної спроможності запитів, у даній розробці є можливість використання зовнішнього периферійного пристрою шифрування.

За допомогою алгоритму обраної хеш-функції, мовою VHDL був розроблений концепт пристрою для створення хеш функції, який обробляє надіслані дані від користувача, та передає до серверу вже у зашифрованому вигляді. Структурну схему наведено у додатку 1.

					ІАЛЦ.467200.004 ПЗ	Арк.
						42
Змін.	Арк.	№ докум.	Підпис	Дата		

## 4. АНАЛІЗ ВИКОНАНОЇ СИСТЕМИ ОНЛАЙН БАНКІНГУ

### 4.1.Опис програмного забезпечення

Розглянемо розроблений проект онлайн банкінгу.

На рисунку 4.1 зображений інтерфейс розробленої системи. Доки користувач не ввійшов у систему, він може або зареєструватися як новий користувач (рисунок 4.2), або увійти, використовуючи його особисті персональні дані (рисунок 4.3).

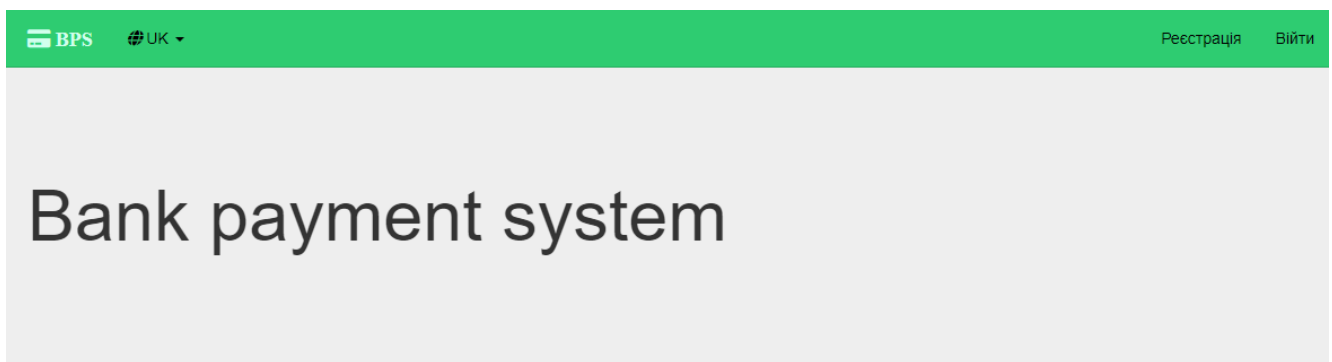


Рисунок 4.1 – Інтерфейс головної сторінки

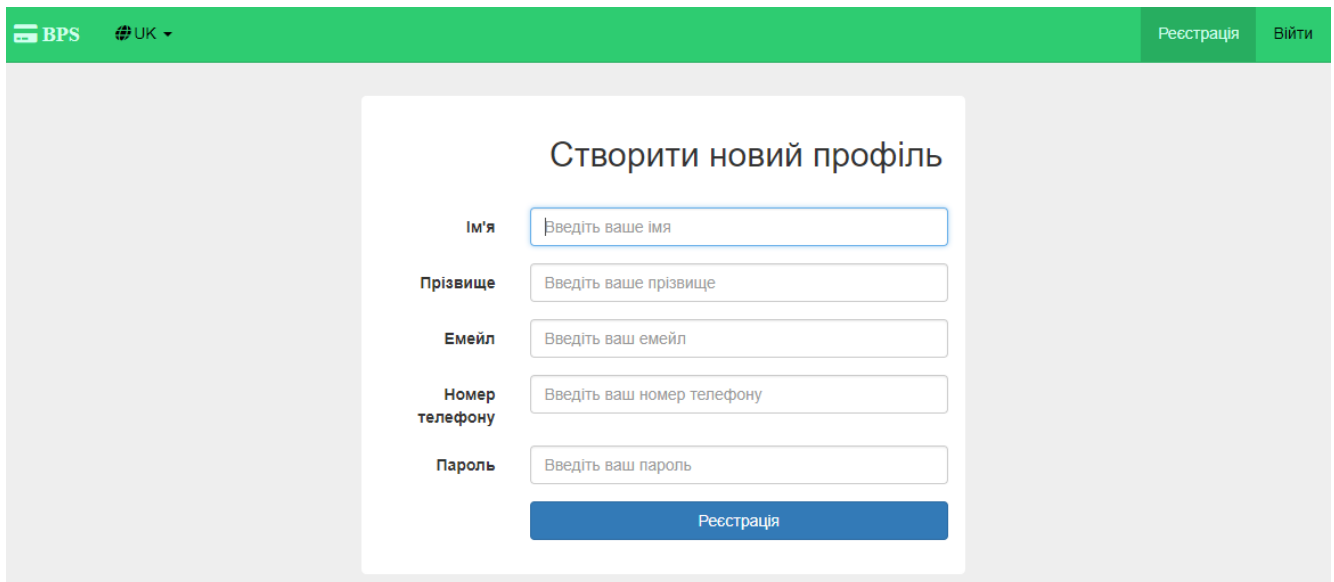


Рисунок 4.2 – Інтерфейс сторінки створення нового користувача у системі

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		43

Рисунок 4.3 – Інтерфейс сторінки створення нового користувача у системі

При вводі невірних даних, система попередить про це відповідним повідомленням (рисунок 4.4).

Рисунок 4.4 – Приклад відображення помилок у системі

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		44

Після входу до системи, або реєстрації, користувачеві відкриваються усі можливості онлайн банкінгу розробленої системи (рисунок 4.5).

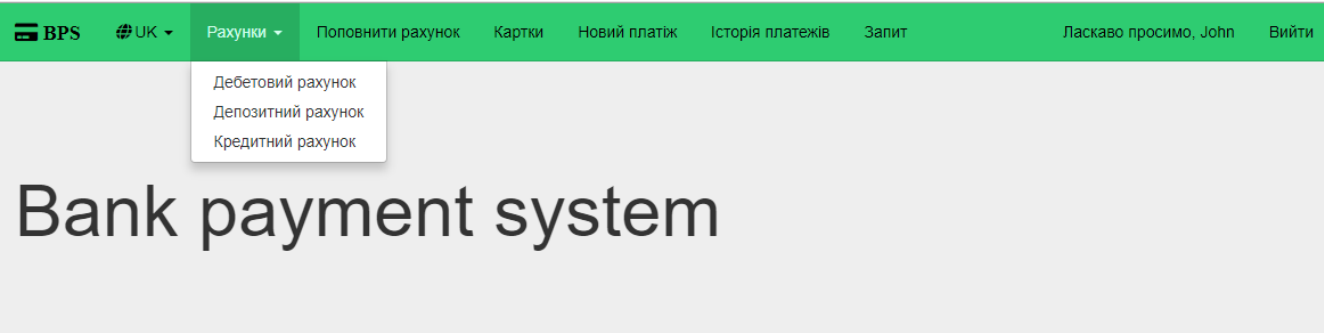


Рисунок 4.5 – Приклад інтерфейсу користувача у системі

Далі користувач може продивитися та виконати зарезервовані дії над рахунками (рисунок 4.6, 4.7, 4.8).

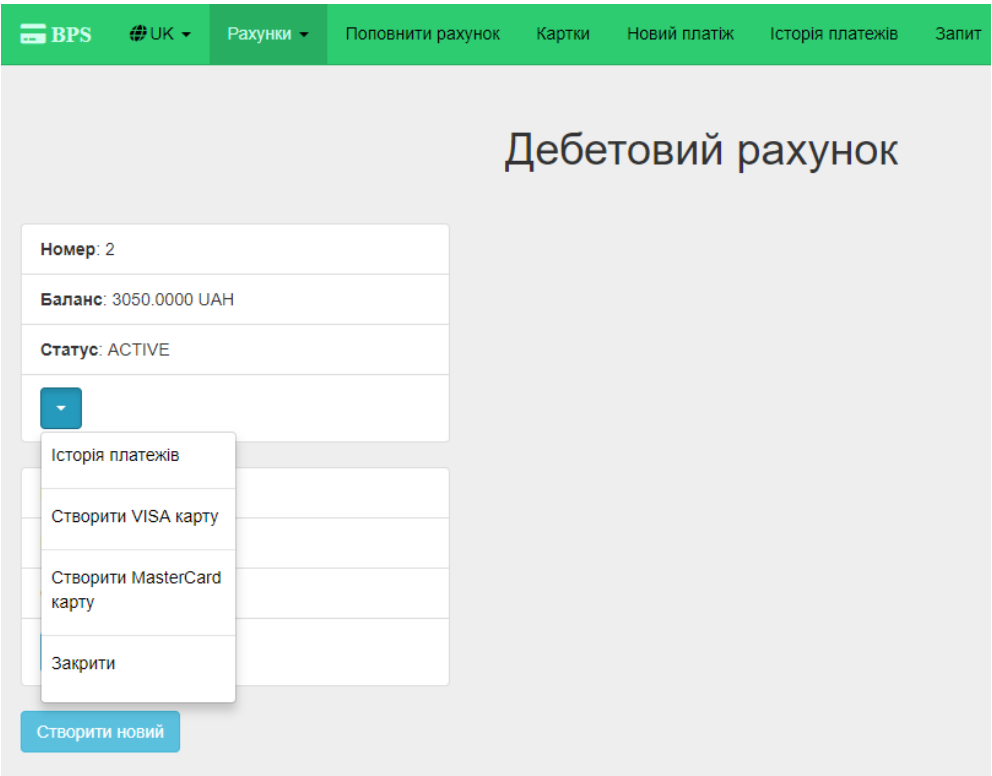


Рисунок 4.6 – Приклад роботи з дебетовим рахунком

BPS

UK

Рахунки

Поповнити рахунок

Картки

Новий платіж

Історія платежів

Запит

Депозитний рахунок

Номер: 6

Баланс: 1005.2500 UAH

Найменший баланс місяцю: 1000.0000 UAH

Річна ставка: 12.6%

Статус: ACTIVE

Зняти

Історія платежів

Поповнити рахунок

Закрити

Рисунок 4.7 – Приклад роботи з депозитним рахунком

BPS

UK

Рахунки

Поповнити рахунок

Картки

Новий платіж

Історія платежів

Запит

Статус: CLOSED

Термін дії: 18 лип. 2018

Номер: 5

Баланс: -2419.4294 UAH

Кредитний ліміт: 12.0000 UAH

Відсоткова ставка: 12.0%

Нараховані відсотки: 9.3294 UAH

Статус: BLOCKED

Термін дії: 15 серп. 2018

Історія платежів

Поповнити рахунок

Закрити

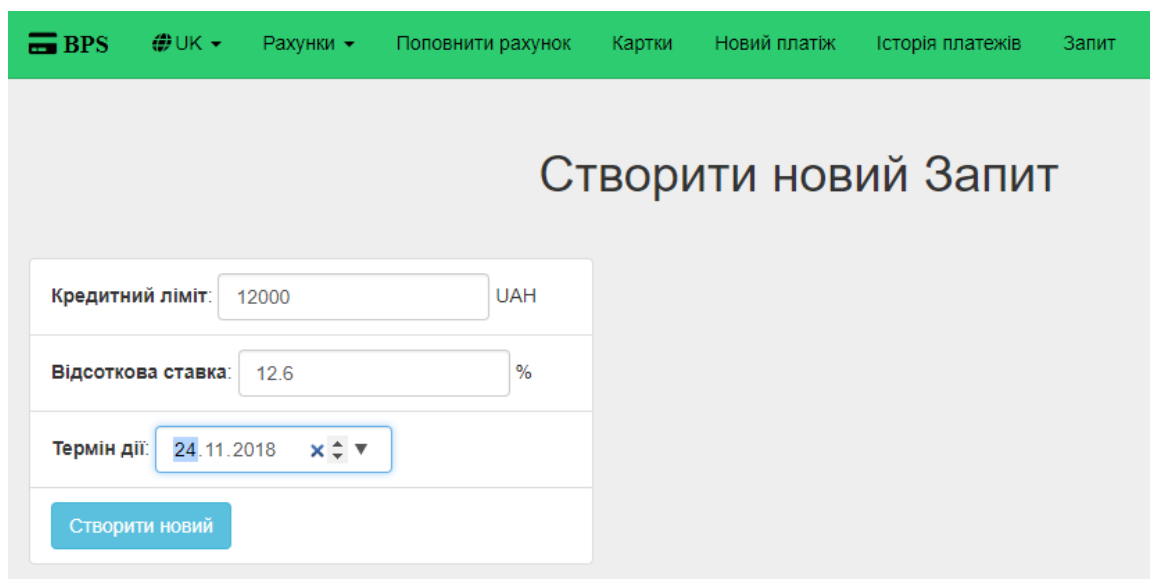
Рисунок 4.7 – Приклад роботи з депозитним рахунком

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		46

Користувач має можливість створювати необмежену кількість дебетових та кредитних рахунків, натиснувши кнопку «Створити».

Для створення кредитного рахунку, користувач не повинен мати активних кредитних рахунків, та не оброблених запитів на створення кредитного рахунку.

Для створення кредитного рахунку, користувачеві необхідно перейти на вкладку «Запити», та створити новий кредитний запит (рисунок 4.8), після чого менеджер побачить цей запит у системі (рисунок 4.9), и дивлячись на дані запиту, або підтвердить запит, після чого для користувача створиться кредитний рахунок з заданими параметрами, або відхилить його, після цього користувач зможе спробувати ще раз створити запит, але вже з іншими налаштуваннями.



The screenshot shows the BPS website interface. At the top is a green navigation bar with links: BPS, UK, Рахунки, Поповнити рахунок, Картки, Новий платіж, Історія платежів, and Запит. The main content area has a heading 'Створити новий Запит'. Below it is a form with three input fields: 'Кредитний ліміт' (Credit limit) with the value '12000' and currency 'UAH', 'Відсоткова ставка' (Interest rate) with the value '12.6' and unit '%', and 'Термін дії' (Term) with a date picker set to '24.11.2018'. A blue button labeled 'Створити новий' (Create new) is at the bottom of the form.

Рисунок 4.8 – Приклад створення кредитного запиту

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		47

BPS	UK	Користувачі	Запити	Рахунки	Картки	Поповнити рахунок	Історія платежів	Ставка	Ласкаво просимо, Ivan	Вийти
-----	----	-------------	--------	---------	--------	-------------------	------------------	--------	-----------------------	-------

Запити					
Користувач	Відсоткова ставка	Кредитний ліміт	Термін дії	Дія	
test@test.com	12.6%	12000.0000 UAH	24 лист. 2018	Підтвердити	Відхилити

Рисунок 4.9 – Приклад сторінки запитів до менеджера

Користувач та менеджер можуть бачити історію переказів. Є можливість переграду переказів за картою, рахунком, або окремим користувачем (рисунок 4.10).

BPS	UK	Рахунки	Поповнити рахунок	Картки	Новий платіж	Історія платежів	Запит	Ласкаво просимо, John	Вийти
-----	----	---------	-------------------	--------	--------------	------------------	-------	-----------------------	-------

Історія переказів					
Рахунок оплати	Номер рахунку відправника	Карта відправника	Номер рахунку отримувача	Сума	Дата
3	2	-	5	2419.4294 UAH	4 черв. 2018
2	8	10000000000000001	2	50.0000 UAH	1 черв. 2018

Рисунок 4.10 – Приклад сторінки історії переказів

Користувач також має можливість переглядати свої кредитні картки (які можуть прив'язуватися виключно до дебетових активних рахунків), та виконувати над ними деякі дії (рисунок 4.11).

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		48



**Ваші карти**

- Номер карти: 1000000000000000
- Номер рахунку: 2
- Баланс: 630.5706 UAH
- Дата закінчення: 31 трав. 2020
- CVV: 361
- PIN: 2848
- Статус: ACTIVE
- Тип: **VISA**
- Історія платежів
- Номер карти: 1000000000000002
- Номер рахунку: 2

Рисунок 4.11 – Приклад сторінки кредитних карток

Користувач також має можливість взаємодіяти з іншими клієнтами. Наприклад, знаючи номер активної банківської карти іншого користувача, можливо виконати переказ коштів з дебетового рахунку, до якого прив'язана існуюча картка (рисунок 4.12).

**Переказ**

Карта відправника: 1000000000000000 (630.5706 UAH)

Карта одержувача: 1234567891234567

Сума: 500 UAH

Введіть cvv:

Введіть pin:

**Виконати**

Рисунок 4.12 – Приклад сторінки переказу

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		49

Якщо були введені невірні дані, система обов’язково попередить про це (рисунок 4.13).

BPS

UK

Рахунки

Поповнити рахунок

Картки

Новий платіж

Історія платежів

Запит

Ласкаво просимо, John

Вийти

Переказ

Помилка! Карта не знайдена.

Карта відправника:

1000000000000000 (630.5706 UAH)

Введіть cvv:

Введіть ріп:

Карта одержувача:

1234567891234567

Сума:

Введіть суму

UAH

Виконати

Рисунок 4.13 – Приклад помилки на сторінці переказу

Цікавою функцією є динамічна зміна річної ставки за депозитом (тільки для новостворених рахунків). Змінити поточну річну ставку може менеджер (рисунок 4.14).

BPS

UK

Користувачі

Запити

Рахунки

Картки

Поповнити рахунок

Історія платежів

Ставка

Річна ставка

Поточна річна ставка: 0.1%

Нова річна ставка.:

13.6

%

Оновити

Рисунок 4.14 – Приклад зміни у системі поточної річної ставки за депозитом

## ВИСНОВКИ

Дистанційні фінансові операції займають невід’ємну частину соціального існування кожної людини у світі. Але проаналізувавши аналоги систем онлайн банкінгу, виявилось, що більшість із них не надають ґрунтовних знань для повноцінного початку роботи. При першому використанні, користувач не знає, які можливості відкриваються перед ним, та які можливі помилки він може виконати, при роботі з електронними рахунками та електронним грошовим оборотом.

Система, яка розроблена у бакалаврському проекті, надає можливість здобуття необхідних навичок роботи з системами дистанційних банківських операцій, та застосовувати її для попереднього пробного виконання функцій онлайн банкінгу без використання реального еквіваленту грошей.

Платформа онлайн банкінгу розроблена на базі мови високого рівня Java і веб-технологій Java Enterprise Edition, враховуючи основні міжнародні стандарти дистанційного банківського обслуговування, а також сучасних принципів проектування захищених веб-орієнтованих систем.

					ІАЛЦ.467200.004 ПЗ	Арк.
Змін.	Арк.	№ докум.	Підпис	Дата		51

## СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Consumer Financial Literacy [Електронний ресурс]. – 2008. – Режим доступу: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1745-6606.2008.00108.x>
2. Использование Интернет-банкинга в Украине [Електронний ресурс]. – 2017. – Режим доступу: <https://business.ua/finansy/item/299-bankir-ex-machina/>
3. Интернет-банк ПРИВАТ24 [Електронний ресурс]. – 2018. – Режим доступу: <https://privatbank.ua/ru/udalenniy-banking/privat24>
4. Ощад24/7 [Електронний ресурс]. – 2018. – Режим доступу: [https://www.oschadbank.ua/ua/private/web\\_bank\\_ng/](https://www.oschadbank.ua/ua/private/web_bank_ng/)
5. UKRSIB ONLINE [Електронний ресурс]. – 2018. – Режим доступу: [https://my.ukrsibbank.com/ua/personal/operations/ukrsib\\_online/](https://my.ukrsibbank.com/ua/personal/operations/ukrsib_online/)
6. How to Boost Your Banking Security [Електронний ресурс]. – 2018. – Режим доступу: <https://www.nerdwallet.com/blog/banking/online-banking-security/>
7. Multi-factor authentication [Електронний ресурс]. – 2012. – Режим доступу: [https://en.wikipedia.org/wiki/Multi-factor\\_authentication](https://en.wikipedia.org/wiki/Multi-factor_authentication)
8. The Java Tutorials [Електронний ресурс]. – 2017. – Режим доступу: <https://docs.oracle.com/javase/tutorial/java/>
9. Строитель (Builder). Порождающий шаблон проектирования [Електронний ресурс]. – 2017. – Режим доступу: <http://areznikov.pro/builder-pattern-java>
10. Паттерн Data Access Object [Електронний ресурс]. – 2017. – Режим доступу: <http://javatutor.net/articles/j2ee-pattern-data-access-object>
11. SHA-2 [Електронний ресурс]. – 2012. – Режим доступу: <https://www.nsa.gov/>

