Міністерство освіти і науки України
Сумський державний університет
Навчально-науковий інститут бізнес-технологій «УАБС»
Кафедра економічної кібернетики

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему «РОЗРОБКА ПРОТОТИПУ АВТОМАТИЗОВАНОГО МОДУЛЮ ПРОЦЕСУ ВИЯВЛЕННЯ ШАХРАЙСЬКИХ ОПЕРАЦІЙ З БАНКІВСЬКИМИ КАРТКАМИ»

Виконав студент 2 курсу, групи ЕК.м-71а
Спеціальності 051 «Економіка»
(«Економічна кібернетика»)
Клімов С. В.
Керівник к.е.н., доцент Яровенко Г. М.

Суми 2018

РЕФЕРАТ

кваліфікаційної магістерської роботи на тему

«РОЗРОБКА ПРОТОТИПУ АВТОМАТИЗОВАНОГО МОДУЛЮ ПРОЦЕСУ ВИЯВЛЕННЯ ШАХРАЙСЬКИХ ОПЕРАЦІЙ З БАНКІВСЬКИМИ КАРТКАМИ»

студента Клімова Сергій Володимировича

Актуальність кваліфікаційної магістерської роботи визначається великою кількістю шахрайських операцій з банківськими картками, що загрожує кібербезпеці України. Кількість випадків шахрайських операцій з банківськими картками залишається досить великою, незважаючи на заходи з безпеки, які застосовують банки. Існуючі системи виявлення шахрайських операцій не підходять до новітніх технологій, які все частіше використовують шахраї.

Мета кваліфікаційної магістерської роботи полягає у розробці та реалізації прототипу автоматизованого модулю для виявлення шахрайських операцій з банківськими картками.

Об’єктом дослідження є процес виявлення шахрайських операцій з банківськими картами.

Предметом дослідження є автоматизовані рішення та інформаційні системи для виявлення шахрайських операцій з банківськими картками.

Задачами кваліфікаційної роботи є дослідження шахрайських операцій з банківськими картками, дослідження існуючих систем виявлення шахрайських операцій, визначення вимог до розроблювальної системи, моделювання бізнес-процесів, відбір технології для реалізації задачі, розробка проекту автоматизованої системи, реалізація інформаційного та алгоритмічного забезпечення, розробка прототипу автоматизованого модуля, розрахунок економічного ефекту від впровадження системи.

Результатом виконаної роботи є реалізований автоматизований модуль виявлення шахрайських операцій з банківськими картками. Даний модуль рекомендується використовувати середнім магазинам електронної комерції, які прагнуть зменшити кількість шахрайських замовлень.

Результати апробації основних положень кваліфікаційної магістерської роботи розглядалися на III Всеукраїнській науково-практичній on-line конференції «Проблеми та перспективи розвитку фінансово-кредитної системи України» 22-23 листопада 2018 року в м. Суми.

Робота виконана в рамках держбюджетної науково-дослідної роботи № 0118U003574 «Кібербезпека в боротьбі з банківськими шахрайствами: захист споживачів фінансових послуг та зростання фінансово-економічної безпеки України».

Ключові слова: шахрайські операції, шахрайські платежі, кібербезпека, автоматизація, банківські картки, автоматизована система, виявлення шахрайства, кіберзагроза, бізнес-процеси, антифрод, anti-fraud, ефективність системи, електронна комерція.

Основний зміст кваліфікаційної магістерської роботи викладено на 51 сторінці. Список використаних джерел із 75 найменувань, розміщений на 7 сторінках. Робота містить 15 таблиць, 17 рисунків, а також 4 додатки, розміщених на 17 сторінках.

Рік виконання кваліфікаційної роботи – 2018 рік.

Рік захисту роботи – 2018 рік.

Міністерство освіти і науки України

Сумський державний університет

Навчально-науковий інститут бізнес-технологій «УАБС»

Кафедра економічної кібернетики

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.е.н., доцент
_____ О.В. Кузьменко
«__» _____2018 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ
(спеціальність 051 «Економіка («Економічна кібернетика»)»)
студенту 2 курсу, групи ЕК.м-71а

Клімова Сергія Володимировича
(прізвище, ім'я, по батькові студента)

1. Тема роботи розробка прототипу автоматизованого модулю процесу виявлення шахрайських операцій з банківськими картками затверджена наказом по університету від «28» жовтня 2018 року № 2277-III

2. Термін подання студентом закінченої роботи «__» грудня 2018 року

3. Мета кваліфікаційної роботи полягає у розробці та реалізації прототипу автоматизованого модулю для виявлення шахрайських операцій з банківськими картками.

4. Об'єкт дослідження: процес виявлення шахрайських операцій з банківськими картами.

5. Предмет дослідження: автоматизовані рішення та інформаційні системи для виявлення шахрайських операцій з банківськими картками.

6. Кваліфікаційна робота виконується на матеріалах: Української міжбанківської асоціації членів платіжних систем ЕМА, розробників програмного забезпечення (PHP, JavaScript).

7. Орієнтовний план кваліфікаційної роботи, терміни подання розділів керівникові та зміст завдань для виконання поставленої мети

Розділ 1 Теоретичні засади процесу виявлення шахрайських операцій з банківськими картками

У розділі 1 здійснити характеристику процесу вивлення шахрайських операцій з банківськими картками, проаналізувати автоматизований процес виявлення шахрайських операцій у ході здійснення он-лайн платежів, сформувати вимоги до автоматизованого модулю виявлення шахрайських операцій

Розділ 2 Розробка проекту автоматизованого модулю виявлення шахрайських операцій

У розділі 2 розробити моделі бізнес-процесу, здійснити вибір архітектури та технології для розробки модулю, охарактеризувати: склад функціональної частини, підсистеми забезпечення функціональної частини (програмне забезпечення, технічне забезпечення, організаційне забезпечення)

Розділ 3 Реалізація прототипу модулю виявлення шахрайських операцій з банківськими картками

У розділі 3 розробити структуру та описати особливості реалізації інформаційного забезпечення; розробити структуру та описати особливості реалізації алгоритмічного забезпечення; розробити контрольний приклад; оцінити очікуваний ефект від впровадження модуля автоматизаці

8. Консультації з роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Доц. Яровенко Г.М.		
2	Доц. Яровенко Г.М.		
3	Доц. Яровенко Г.М.		

9. Дата видачі завдання: «___»_____ 20__року

Керівник кваліфікаційної роботи _____ Г. М. Яровенко
(підпис) (ініціали, прізвище)

Завдання до виконання одержав _____ С. В. Клімов__
(підпис) (ініціали, прізвище)

ЗМІСТ

ВСТУП

Виявлення шахрайства є проблемою сьогодення. Завдяки широким можливостям виконання операцій з рахунком у банку та банківськими картками вчасності (наприклад через Інтернет, мобільний банкінг, телефонний банкінг) проблема виявлення шахрайства суттєво зросла.

Шахрайські операції з банківськими картками – це те, що може загальмувати розвиток онлайн-бізнесу. Якщо товаром або послугою скористався шахрай, втрачається і товар, і гроші. Дуже просто купити товар на сайті, ввівши при оплаті номер карти й інші цифри, які надруковані на ній. Але при цьому карта буде чужа – введені дані можна сфотографувати або підглянути, роздобути за допомогою технологічних махінацій з банкоматами або через слабо захищені сайти інших інтернет-магазинів.

Після виконання шахрайської операції справжній власник картки обов'язкового напише заяву в банк про повернення списаної без його відома суми. У разі проходження несанкціонованої операції по банківській карті через інтернет-магазин банк-емітент за дорученням власника картки опротестує транзакцію і онлайн-крамниця буде зобов'язана відшкодувати всю вартість покупки.

Актуальність створення даної системи обумовлюється зростанням сфери електронної комерції, популяризацією безготівкових розрахунків та збільшенням кількості шахрайств в мережі Інтернет. Наукова новизна роботи полягає у вдосконалені існуючої системи виявлення шахрайських операцій, що виникають при здійснені онлайн-платежів.

Об'єктом роботи виступає процес виявлення шахрайських операцій з банківськими картами. Предметом дослідження є автоматизовані рішення та інформаційні системи для виявлення шахрайських операцій з банківськими картками.

.

Метою магістерської кваліфікаційної роботи є розробка та реалізація прототипу автоматизованого модулю для виявлення шахрайських операцій з банківськими картками.

Поставлена мета визначила наступні завдання:

- дослідити шахрайські операції з банківськими картками;
- дослідити існуючі системи виявлення шахрайських операцій;
- визначити вимог до розроблювальної системи;
- розробити моделі бізнес-процесів;
- відібрати технології для реалізації задачі;
- розробити проект автоматизованої системи;
- реалізувати алгоритмічне та інформаційне забезпечення;
- розробити прототип автоматизованого модулю;
- розрахувати економічний ефект внаслідок впровадження автоматизованого рішення.

Апробація кваліфікаційної магістерської роботи здійснена на III Всеукраїнській науково-практичній on-line конференції «Проблеми та перспективи розвитку фінансово-кредитної системи України» 22-23 листопада 2018 року.

Робота виконана в рамках держбюджетної науково-дослідної роботи № 0118U003574 «Кібербезпека в боротьбі з банківськими шахрайствами: захист споживачів фінансових послуг та зростання фінансово-економічної безпеки України».

РОЗДІЛ 1 ТЕОРЕТИЧНІ ЗАСАДИ ПРОЦЕСУ ВИЯВЛЕННЯ ШАХРАЙСЬКИХ ОПЕРАЦІЙ З БАНКІВСЬКИМИ КАРТКАМИ

1.1 Характеристика шахрайських операцій з банківськими картками

У сфері грошового обігу банківські карти є засобами організації безготівкових розрахунків. В даний час інтерес щодо використання банківських карт з кожним днем стає все більше. Використання банківських карт дозволяє клієнтам розраховуватися за товари та послуги без готівкових коштів.

Банківські картки є універсальним міжнародним засобом платежу, тому що за допомогою них можна розраховуватися будь-де, незалежно від валюти, в якій відкрито банківський рахунок [1]. Вони допомагають власникам зменшити витрати та незручності, що пов’язані зі зберіганням, переміщенням, конвертацією грошей. Вони надають змогу швидко проводити розрахунки та отримати готівкові кошти з рахунку. Зростаюче використання банківських карт в торгово-сервісних організаціях стало потужним стимулом у реалізації товарів і послуг [2-3]. Завдяки різкому зростанню популярності онлайн-покупок, використання банківських карток досягло високого рівня.

Шахрайська транзакція – це неавторизована транзакція з дебетової або кредитної картки клієнта. Вона може виникнути з картки, яка була втрачена, викрадена, підроблена, не отримана або шахрайська за рішенням банку [4-7].

Відповідно до статті 190 Кримінального кодексу України шахрайство визначається як заволодіння чужим майном або придбання права на майно шляхом обману чи зловживання довірою. Шахрайство, вчинене у великих розмірах, або шляхом незаконних операцій з використанням електронно-обчислювальної техніки, карається позбавленням волі на строк від трьох до восьми років [8].

У 2016 році VISA і MasterCard ввели в Україні принцип нульової відповідальності, який функціоную також в усьому світі [9]. Він означає, що якщо власник банківської карти цих платіжних систем став жертвою шахрайства, а потім зміг це довести, то банки повинні компенсувати йому кошти на його рахунок.

Хоча споживачі можуть бути впевнені, що такі заходи, як політика щодо нульової відповідальності, захищають їх від несанкціонованих транзакцій за допомогою цифрових та електронних платежів, вони, зрештою, також мають свою роль. Знаючи, як відбувається шахрайство, це хороший спосіб вжити заходів для його запобігання.

За даними Української міжбанківської асоціації членів платіжних систем середня сума втрат клієнтів в Україні від платіжного шахрайства знизилася у 2017 році – 145 грн. у порівнянні з 345 грн у 2016 році [10]. Цього результату вдалося досягти завдяки вжитим банками заходів [11] – вдосконалення системи виявлення шахрайських операцій, встановлення лімітів на платежі в Інтернеті, додаткова автентифікація клієнтів тощо.

У зв’язку з використанням зловмисниками нових мініатюрних пристроїв для скіммінгу, що ускладнюють їх візуальне визначення на банкоматах, триває зростання кількості скіммінгових інцидентів: 113 в 2017 проти 71 випадку в 2016 році [10].

Суттєво зменшилася кількість випадків кеш-треппінгу (захоплення готівки в банкоматі): знизилася в 4 рази – з 817 (2016 рік) до 191 в 2017 році. Позитивний результат вдалося досягти в результаті оснащення банкоматів захисними пристроями та інформування населення в рамках Національної програми SafeCard.

Кількість сайтів, які отримують реквізити банківських карт обманним шляхом, стало також менше: в 2017 році було виявлено і закрито 108 фішингових веб-ресурсів [10]. Для порівняння у 2016 році – 174 фішингових сайтів. Проте це не означає, що їх стало менше. Почали створювати більш якісні сайти, які важче ідентифікувати [12].

Шахраї почали використовувати все частіше сайти для синхронного переказу грошей. Так у момент здійснення операції дані переправляються на легальний платіжний сервіс, при цьому номер картки одержувача змінюється на новий – номер карти шахрая (також можлива зміна і суми переказу) [13-14]. Раніше шахраї тільки збирали реквізити карт, намагаючись зняти кошти з рахунків клієнтів пізніше. А тепер вони схильні відразу вкрасти кошти. За статистичними даними, у 2017 році подібних сайтів серед шахрайських було більше 25%.

Також шахраї частіше стали використовувати SSL сертифікат на сайтах. Раніше він вважався ознакою надійності, а зараз кожен четвертий фішинговий веб-ресурс має https у своїй адресній строці. У 2016 році їх кількість становило всього 5% від загальної кількості шахрайських інформаційних систем [10].

Слід зауважити, що велика кількість шахрайських операцій відбувається безкарно, тому що жертви шахраїв не намагаються цьому запобігти. За одним випадком шахрайських операцій було зафіксовано шахрайські дії з 2600 банківськими картками, а в поліцію звернулися лише 36 власників банківських карток.

Кількість номерів телефонів в чорних списках міжбанківської системи обміну інформацією про інциденти платіжного шахрайства Exchange-online, з яких шахраї телефонували та відправляли SMS повідомлення потенційним жертвам, за 2017 рік зросла у 5 разів – 3430 номерів.

Фішингових сайти створюють в основному за наступними темами [15]:

- P2P;
- поповнення мобільного рахунку;
- заробіток за опитування;
- заробіток в Інтернеті;
- отримання кредиту;
- авіаквитки;
- білети на концерти.

Вчені виділяють дві категорії методів виявлення шахрайства: наглядова та безконтрольна класифікації [16-17].

Наглядова класифікація:

– необхідні приклади з минулого шахрайської та легітимної діяльності;

– метод ефективний для виявлення відомих типів шахрайства;

– неможливо виявити нові типи.

Методами цієї категорії є нейроні та байєсівські мережі. Окрім них, розповсюдженого застосування набув метод аналізу зміщення тренду.

Безконтрольна класифікація:

– потрібні лише приклади попередньої законної діяльності;

– висока ефективність у виявленні нових типів шахрайства;

– ефективні при відомих типах, що відхиляються від норми;

– нормальний стан може базуватися на порівнянні діяльності самого клієнта або інших клієнтів в попередні часи.

Проблеми визначення шахрайських операцій з банківськими картами полягає в наступних особливостях:

– великі набори даних транзакцій (кількість транзакцій у мільйонах);

– більшість змінних будуть несуттєвими;

– більшість операцій не є шахрайством (лише 0.1% операції є шахрайськими);

– затримка в навчанні;

– шахраї змінюють стратегію і технології.

Інформування споживачів про те, що вони можуть робити аби захистити себе, є найважливішим запобіжним заходом у розвитку протидії глобальному та регіональному шахрайству. Найголовнішим є розуміння різних видів шахрайства, пов’язаних з операціями з дебетовою та кредитною картками. Існує вісім основних видів [18-19]:

– загублені чи вкрадені картки – в цьому випадку слід негайно повідомити про це свій банк, щоб закрити рахунок і мінімізувати будь-які збитки;

– поглинання рахунку – такий випадок, коли власник картки мимоволі надає особисту інформацію (таку як домашня адреса, дівоче прізвище матері тощо) шахраю, який потім звертається до банку власника картки, повідомляє про втрачену картку та зміну адреси і отримує нову картку на ім’я жертви;

– підроблені карти – випадок, коли карту клонують, а потім використовують для здійснення покупок. За статистикою від 10 до 15% шахрайства виникають внаслідок використання таких карт. В останні роки цей показник суттєво зменшився через покращення функцій безпеки, встановлених для платіжних карток, таких як чіп EMV;

– ніколи не отримана картка – це випадок, коли вдається через пошту викрасти нову або замінену картку, яка ніколи не потрапляє до її законного власника [20-21];

– шахрайське застосування – випадок, коли шахрай використовує ім’я та особисті дані іншої людини для подання заявки на отримання кредитної картки;

– множинний відбиток – випадок, коли одна транзакція записується кілька разів на старомодні машини віддруку кредитних карт;

– продавці, які змовилися – торговці працюють з шахраями з метою введення в оману співробітників банку;

– шахрайство електронною поштою або телефонним замовленням – випадок, який зараз включає електронну комерцію, і є найбільшою категорією загального обману платіжних карт, що становить майже три чверті всіх випадків шахрайства. Сектор платежів невтомно працює над удосконаленням програм перевірки карт та програм безпеки, щоб запобігти шахрайству з транзакціями в режимі онлайн або через поштові та телефонні замовлення [22].

1.2 Аналіз автоматизованого процесу виявлення шахрайських операцій у ході здійснення он-лайн платежів

Індустрія платежів вже давно створює нові інновації для боротьби з шахраями, щоб створити безпечне середовище для фінансових операцій. Багато з цих досягнень безпеки: від голограми та панелі підпису з підтвердженням до кодів підтвердження карт та чипа EMV – стали вже галузевими стандартами [23].

Проте цього не достатньо. Споживча освіта залишається ключовою частиною стратегії захисту інформації про рахунок власника карток та допомога банкам та торговим підприємствам у запобіганні збитків через шахрайські дії з платіжними картками [24].

Прийнято вважати, що від інтернет-шахрайства з пластиковими картами страждають покупці – власники цих самих карт. Але насправді інтернет-магазинам дістається не менше [25]. Шахрайські операції загрожують продавцеві втратою грошей, клієнтів і репутації.

Шахрайство в онлайн-магазинах в першу чергу небезпечне для самих магазинів, або торгово-сервісних підприємств (ТСП), тому що саме через них здійснюють шахрайські операції і саме з них будуть питати, якщо власник картки заявить про незаконне списання грошових коштів.

Класична схема працює наступним чином: в результаті скіммінгу / фішингу або будь-яких інших протиправних дій власник банківської карти сам того не знаючи передає зловмисникам дані своєї карти, достатні для здійснення покупки в інтернет-магазині [26]. Зловмисник оформляє покупку і отримує товар або послугу. Власник картки, дізнавшись про несанкціонований списання, заявляє про зникнення грошей в банк, що випустив карту. У свою чергу банк ініціює перевірку.

Якщо факт шахрайства підтверджується, то відбувається повернення списаних коштів на рахунок клієнта, а вони списуються з ТСП. Якщо товар

вже був отриманий зловмисником, то ТПС отримує потрійний збиток: повертає гроші власнику картки, позбавляється товару, за який вже заплачено постачальнику, плюс може заробити штраф за те, що виконав шахрайську транзакцію. У найгіршому випадку отримує повну заборону приймати онлайн платежі.

Для виявлення шахрайських операцій та протидії їх виникнення використовують системи антифрод (з англ. anti-fraud «боротьба з шахрайством»). Антифрод – це система моніторингу та запобігання шахрайських операцій, яка в режимі реального часу перевіряє кожен платіж, звіряючи його через десятки, а часом сотні фільтрів. Механізми антифрода працюють таким чином, щоб простежити, чи немає в платежі чогось незвичайного, що виходить за межі сформованого стандарту [26-29].

Завдання системи є перевірка кожної транзакції, знаходження в ній підозрілих моментів і винесення рішення – відхилити платіж або підтвердити його [30]. У класичному вигляді система антифрода складається з декількох компонентів: це автоматичний моніторинг транзакцій, що включає в себе безліч параметрів фільтрів, механізми автентифікації власника картки і верифікації карти, а також моніторинг транзакцій в ручному режимі для крайніх випадків.

Система повинна захищати учасників процесу онлайн-платежів від можливих шахрайських операцій. Схематично взаємозв’язок учасників при здійснені платежів в мережі Інтернет в загальному вигляді можна подати як на рисунку 1.1 [26].

Суцільна та пунктирна лінії показують зв’язок компонентів у режимі реального часу та з затримкою у часі відповідно.

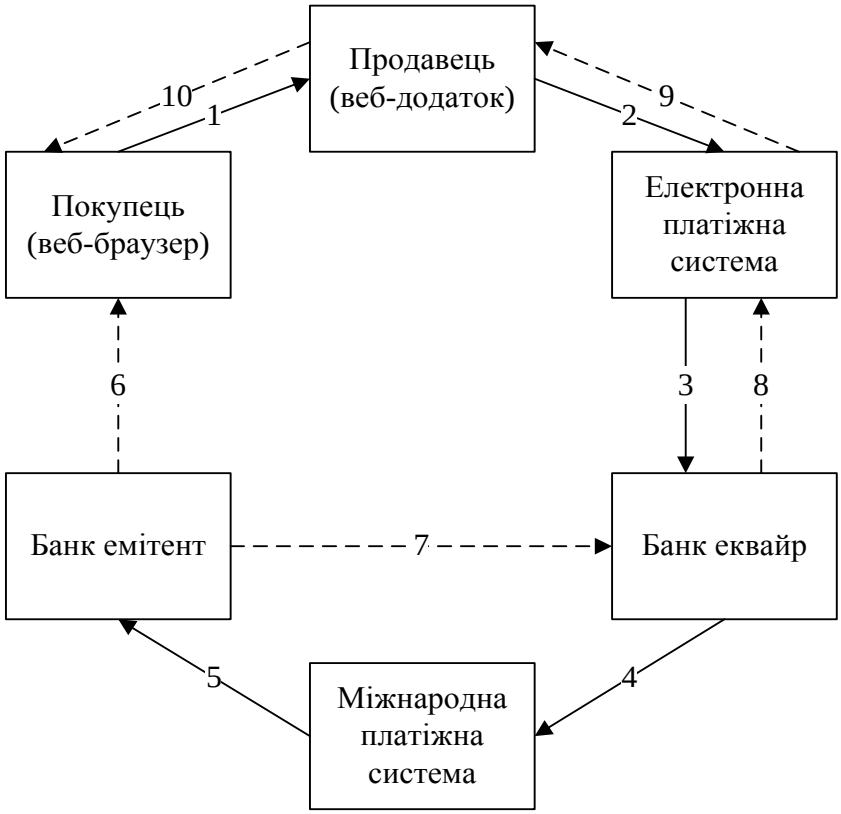

Рисунок 1.1 – Загальний вигляд процесу онлайн-платежів

В залежності від розробника системи фільтри антифрода можуть бути різними. Наведемо приклади фільтрів, які можна використовувати у системі виявлення шахрайських операцій [28, 31-34]:

Фільтри-валідатори. Як приклад це валідатор реквізитів банківської карти. Уже в процесі введення на платіжній формі номер карти перевіряється системою за алгоритмом Луна – так система засвідчує коректність введеного номеру картки.

Географічні фільтри. Наприклад, по країнам IP-адрес. Статистика показує, що в деяких країнах Африки високий рівень скіммінгу і компрометації карт, і як результат платежі, що здійснюються з цих країн, з високою часткою ймовірності виявляться шахрайськими.

Фільтри чорного списку. Якщо система отримує дані карти, по якій вже проходили платежі з поміткою як шахрайські, або власник карти заявив в банк-емітент про компрометацію її даних, така карта потрапляє в чорний

список. Тоді система знає, що по ній не можна пропускати транзакції, так як вони виявляться шахрайськими.

Фільтри відповідності (збігу) параметрів. Приклад: відповідність країни IP-адреси платника і країни емітента банківської карти. Якщо платіж здійснюється не з тієї країни, де була випущена карта, а власник картки не попередив в банк заздалегідь інформацію про свої подорожі, є ймовірність того, що реквізити карти були вкрадені і використовуються зловмисниками.

Фільтри лімітів авторизації. Наприклад, ліміт суми однієї транзакції, кількості спроб авторизації з однієї IP-адреси або з однієї банківської карти. Для захисту як платника, так і інших учасників процесу онлайн-оплати існують обмеження за кількістю і сумою платежів, що здійснюються протягом дня або іншого періоду. Для деяких типів бізнесу особливо великий платіж, якби опинився шахрайським, при поверненні може значно вдарити по прибутку.

Система антифрод є своєрідним бар'єром, через який повинна пройти операція. Наведемо спрощений вигляд процесу онлайн-платежу, врахувавши антифрод систему (рисунок 1.2).

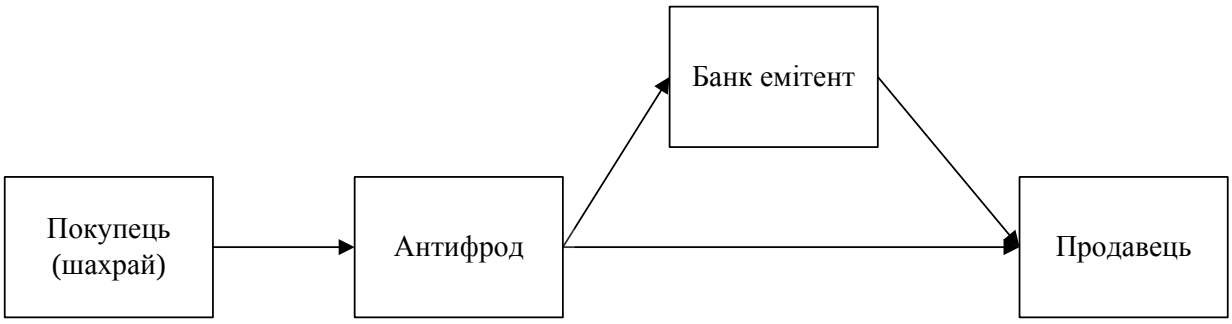

Рисунок 1.2 – Місце антифрод системи в шахрайському процесі онлайн платежу

Якщо операція проходить систему антифрод, то дані передаються в банк, що необхідно переказати кошти до магазину та до продавця, що необхідно відправити товар. В іншому випадку платіж не виконується і продавець не отримує замовлення.

Всього система може включати в себе сотні різних фільтрів, і чим більше сфера бізнесу схильна до шахрайських дій, тим більше фільтрів включається і тим точніше кожен з них налаштовується під конкретний інтернет-магазин або онлайн-сервіс [34].

Якщо подібна система відсутня або вимкнули її фільтри, то магазин почне пропускати шахрайські платежі значно більше, ніж якби фільтри антифрода працювали і перевіряли кожну транзакцію. За умови використання 3-D Secure, де покупець зобов'язаний підтвердити платіж за допомогою одноразового пароля, що приходить по СМС, інтернет-магазин може мінімізувати втрати [32]. Однак в разі масового проходження шахрайських операцій магазин все ж таки може бути відключений від платіжної системи. Достатньо аби кількість шахрайських транзакцій досягла 1-2% від кількості всіх платежів на сайті за певний період – після цього банк-еквайр вже може заблокувати проведення оплати.

У ситуації, коли 3-D Secure не використовується, ситуація може виявитися більш ніж гіршою: конверсія в успішні платежі може прагнути до 100%, але втрати від такого необдуманого кроку стануть катастрофічними для магазину. Однак, в реаліях сучасного ринку ситуацію з відключенням всіх механізмів захисту уявити складно. На таких умовах з магазином відмовляться працювати і банки-еквайри, і платіжні системи ще на етапі підключення.

При включенні всіх фільтрів відсоток прийнятих платежів може значно впасти. Деякі бізнеси такий захист може просто вбити [34]: наприклад, якщо ми говоримо про продаж авіаквитків, обмеження по країнам може негативно позначитися на продажах, адже покупець з картою українського банку може знаходитися в Іспанії і на німецькому сайті оплачувати авіаквиток. Відповідно, при включенні всіх фільтрів ми забезпечуємо 100% рівень безпеки, але значно знижуємо конверсію в успішні платежі. Існує розбіжність країни банку-емітента, сайту-продавця і країни, в якій

виконується онлайн-операція. В такому випадку система не пропустить платіж.

Серед існуючих систем виявлення шахрайства виділяють FraudlabsPro, Oculeus Anti-Fraud System, Alaris Anti Fraud System, Riskified, WayForPay Antifraud. Необхідно визначити їх переваги та недоліки (таблиця 1.1) [35-39].

Таблиця 1.1 – Порівняльна характеристика антифрод-систем

Критерій	FraudlabsPro	Oculeus	Alaris	Riskified	WayForPay
Легкість встановлення	+	-	-	+	+
Фільтрація	+	+	+	-	-
Інтегрованість	+	+	-	+	+
Геолокація	+	-	-	-	-
Можливість вносити зміни	-	-	-	+	-
Виведення звітів	+	+	+	-	+
СМС верифікація	+	+	+	-	-
Локалізація	-	-	-	-	+
Безкоштовність встановлення	+	-	+	+	+
Безкоштовність використання	-	-	-	-	-

Згідно з таблицею 1.1, існуючі системи виконують свою головну роль виявлення шахрайства, проте мають функціональні обмеження, що не підходять для роботи українським підприємствам електронної комерції. Найголовнішими проблемами є відсутність можливості модифікації системи, налаштування критеріїв виявлення шахрайства, української (російської) мови, плата за встановлення та використання. З огляду на це було вирішено розробити та реалізувати власну систему виявлення шахрайських операцій з банківськими картками.

Як було зазначено, система виявлення шахрайських операцій вимагає ретельного налаштування, щоб зберегти високий рівень безпеки і при цьому не втративши більшу частину прибутку.

1.3 Формування вимог до автоматизованого модулю виявлення шахрайських операцій

Антифрод-система є business critical системою, правильність і цілісність якої необхідні для успіху ринкової або ділової діяльності. Некоректність роботи системи буде вести до збільшення ризиків фінансових втрат для компанії. Звідси підвищені вимоги до надійності роботи, безпеки зберігання даних, відмовостійкості, масштабованості системи [40].

Ефективна система повинна мати наступні якості:

- розподіленість;
- відмовостійкість;
- висока масштабованість;
- надійність.

Для початку розглянемо вимоги до системи з точки зору зовнішнього світу, тобто інтернет-магазину. Він взаємодіє з антифрод-системою у відповідності з наступними вимогами до API:

функціональні:

- надати клієнту API для відправки даних про платіж;
- повернути результат чи є платіж шахрайським;
- надати клієнту API для коригування результатів проведення платежу.

нефункціональні:

- надати загальнодоступний протокол взаємодії з клієнтом;
- взаємодіяти з клієнтом тільки по захищених каналах зв’язку.

З точки зору внутрішньої логіки антифрод-системи виділимо всього одну суттєву бізнес-вимогу: виявити за даними про платіж чи буде транзакція шахрайською.

У кінцевому результаті розроблена інформаційна система має відповідати наступним вимогам:

– виконувати свою головну функцію;

– бути орієнтованим на користувача і виконувати його бізнес-задачі в сфері виявлення шахрайства;

– мати можливість вносити зміни у програмний код;

– мати фільтри виявлення шахрайства, які можна налаштовувати;

– бути економічно ефективним.

Автоматична система буде мати 2 інтерфейси для різних груп користувачів [41]. Для клієнтів магазину електронної комерції це буде веб-додаток з формою для заповнення. Друга група користувачів, це співробітники банку, які бачать інформацію про операції, які система визначила шахрайськими.

З цієї причини розділимо функціональні вимоги системи також на 2 групи. Як частина інформаційної системи електронної комерції, модуль повинен бути легко інтегрованим, мати зручну, інтуїтивно зрозумілу форму для заповнення клієнтом магазину, адаптивний дизайн, який можна легко змінити за допомогою таблиць стилів. Час відгуку системи має бути зведений до мінімуму. У випадку класифікації платежу як шахрайського, має бути можливість підтвердити його та змінити цей статус.

Вимоги до частини системи, яка використовується у банку, полягає у наступному: виводити інформацію про платіж, який був визначений шахрайським, його поточний статус, прізвище, ім’я та телефон власника картки, за допомогою якої виконувався даний платіж. Можливість фільтрувати та змінювати дані безпосередньо у цьому додатку.

Дотримання вищеназваних вимог дозволить розробити ефективний автоматизований модуль виявлення шахрайських операцій з банківськими картками, який буде використовуватися в електронній комерції для зменшення шахрайських платежів та чарджбеків.

РОЗДІЛ 2 РОЗРОБКА ПРОЕКТУ АВТОМАТИЗОВАНОГО МОДУЛЮ ВИЯВЛЕННЯ ШАХРАЙСЬКИХ ОПЕРАЦІЙ

2.1 Розробка моделей бізнес-процесу

Одним з кроків створення ефективної інформаційної системи є її попереднє моделювання. Відтворення моделі дозволяє отримати загальний вигляд даних інформаційної системи [42]. Цей загальний вигляд системи, яким користуються всі учасники (всі підсистеми), є механізмом, що дозволяє системно підійти до проекту.

Бізнес-процес відображає організаційну структуру системи і моделювання може дозволити організації належним чином керувати своїм робочим процесом [42]. Моделювання бізнес-процесів може систематично відображати потоки ділової активності, що дозволяє проводити відповідний аналіз та моделювання.

Моделювання бізнес-процесів визнаються як інструмент, який може допомогти організації ефективно працювати і легко знаходити проблемні зони. Більше того, він дозволяє перетворити або модернізувати бізнес-процеси. Таким чином, чим краще буде моделювання бізнес-процесів, тим більше можна покращити продуктивність та конкурентоспроможність організації.

Під час попереднього дослідження (перша ітерація) вибираються найважливіші дані з урахуванням обсягу та частоти процесів. Важливо визначити підмножину інформації, що дозволить добре сформулювати модель даних та всі підсистеми [43].

Система виявлення шахрайських операцій складається з наступних складових (підсистем):

– Fraud Predictor Service – сервіс виявлення шахрайських операцій за допомогою перевірки за різними фільтрами;

– Transactions Log – база даних транзакцій банківських карт;

– SMS API Service – сервіс верифікації за допомогою повідомлення на мобільний телефон.

Крім того система містить клієнтські веб-додатки як, наприклад, веб-додаток для банку для відображення транзакцій, котрі система визначила шахрайськими.

Взаємодію компонентів як єдиної системи продемонстровано на діаграмі послідовності на рисунку 2.1.

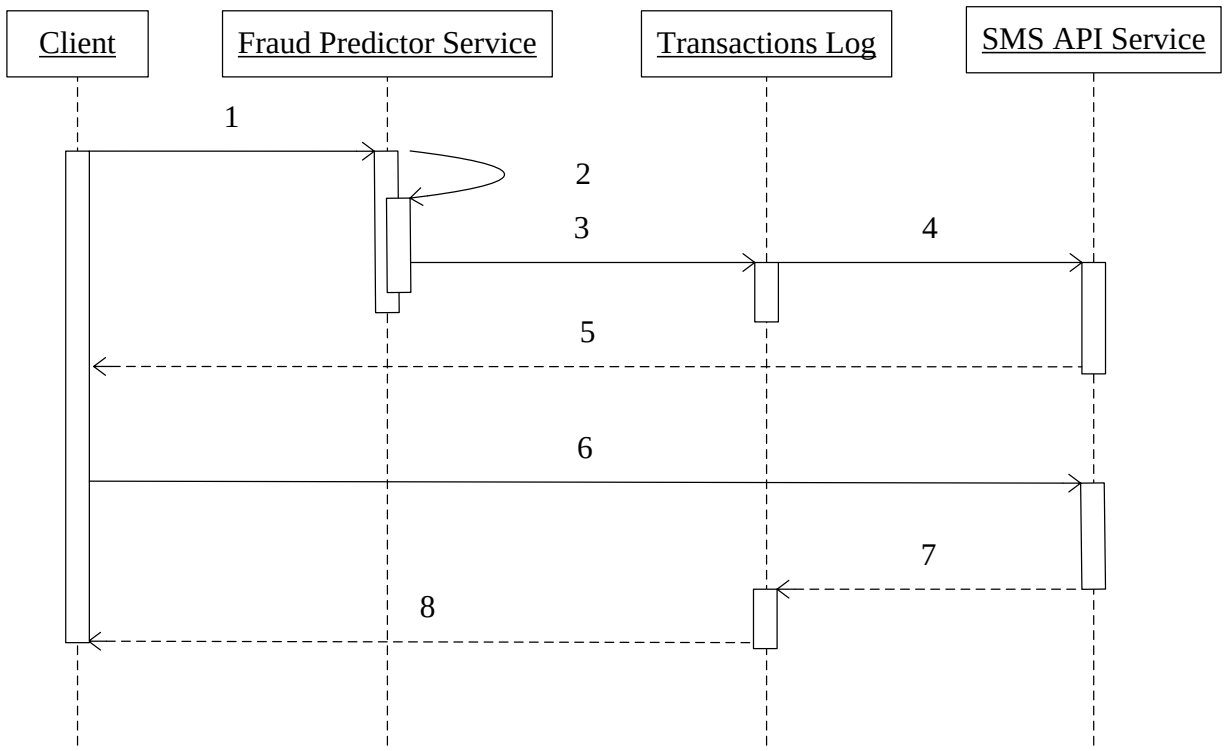

Рисунок 2.1 – Діаграма послідовності системи

Прокоментуємо подану діаграму поетапно:

– Крок 1. Відправка запиту з боку клієнта до системи.

– Крок 2. Робота сервісу виявлення шахрайських операцій та повернення результату – чи буде платіж шахрайським.

– Крок 3. Збереження даних.

– Крок 4. Виклик вікна додаткової верифікації.

– Крок 5. Повернення результату клієнту.

– Крок 6. Введення коду, отриманого у повідомлені.

– Крок 7. Зміна інформації про транзакцію.

– Крок 8. Повернення результату клієнту.

Кроки 4-8 відбуваються тільки у випадку шахрайського платежу.

Перед розробкою автоматизованої системи необхідно розглянути її з точки зору бізнес-процесів, побудувавши бізнес-моделі. Вони являють собою формалізований (графічний, табличний, текстовий, символьний) опис бізнес-процесів, що відображає реально існуючу або передбачувану діяльність [44].

Для моделювання та опису бізнес-процесів прийнято використовувати спеціалізовані системи управління бізнес процесами – BMP (Business Process Management) системи, які використовують наступні нотації моделювання:

– BPMN (Business Process Model and Notation) – нотація моделювання бізнес процесів, яка забезпечує високий рівень наочного зображення процесу. Вони розробляються як стандартні блок-схеми [45].

– BPEL (Business Process Execution Language) – це спеціальна XML-мова виконання бізнес-процесів. Вона подає окремий бізнес-процес у вигляді послідовності веб-сервісів.

– DFD (Data Flow Diagramming) – опис потоків даних. Відображення інформаційних потоків, які відбуваються протягом робіт. Також дану нотацію застосовують для опису документообігу.

– IDEF0 (Business Process Modeling) – методологія для опису бізнес-процесів, чиї моделі використовуються для опису робіт процесу. В нотації враховуються не тільки входи і виходи, а й управління та механізми, тобто дозволяє описувати керування процесами організації [44].

– IDEF3 – нотація, що зосереджена на описі потоків робіт (Work Flow Modelling). Стандарт IDEF3 наближений до стандартних блок-схем, але включає в себе орієнтованість на алгоритмічність методу побудови схем бізнес-процесів [44].

– XPDL (XML Process Definition Language) – формат обміну даними між BPM-системами. Використовується в основному як стандарт виконання експорту-імпорту описів бізнес-процесів [46].

Для опису бізнес-моделі нашої системи будемо використовувати нотації IDEF0 та IDEF3. В основі даної методології покладені чотири основні поняття [44]:

– функціональний блок;

– інтерфейсна дуга;

– декомпозиція;

– глосарій.

На рисунку 2.2 зображено контекстну діаграму процесу виявлення шахрайських операцій. Для її розуміння необхідно описати всі елементи, що присутні на діаграмі (таблиця 2.1).

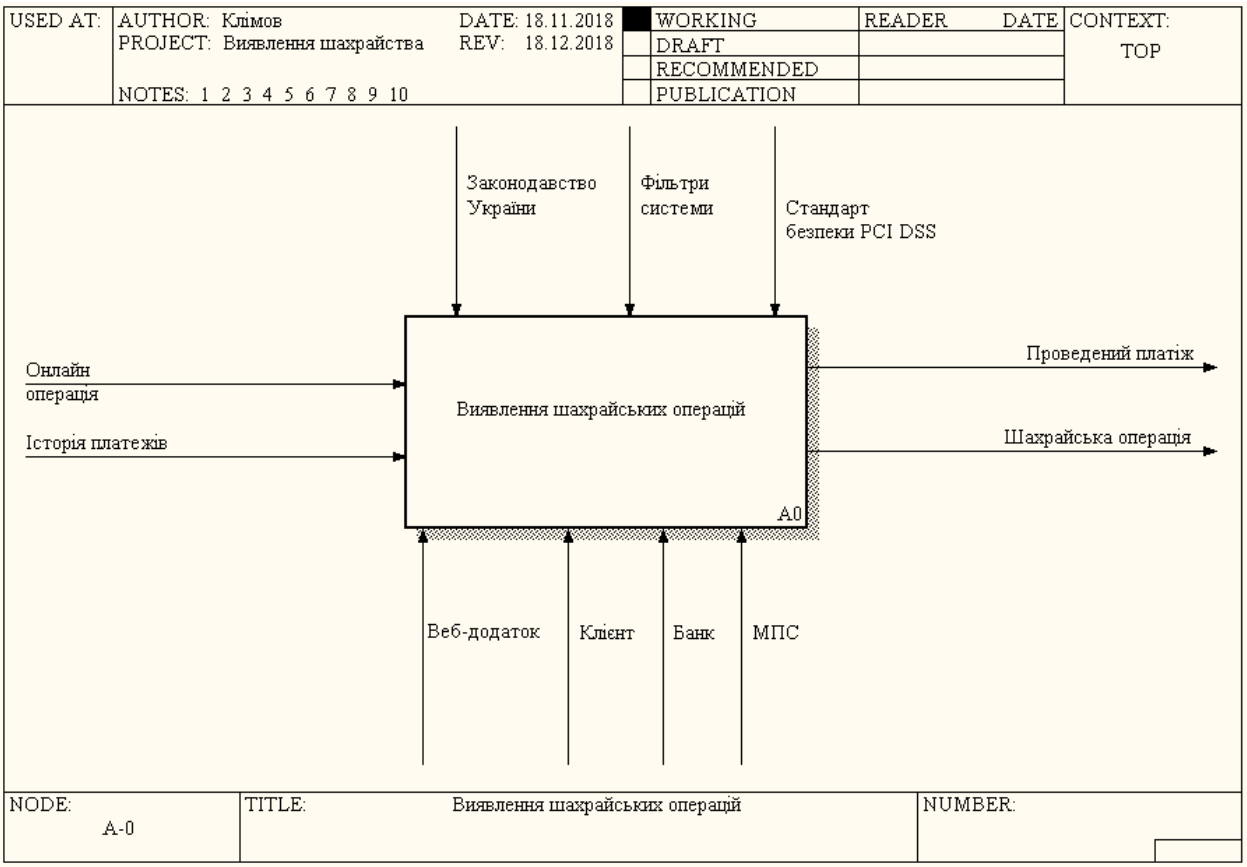

Рисунок 2.2 – Контекстна діаграма «Виявлення шахрайських операцій»

Таблиця 2.1 – Опис основних елементів контекстної діаграми

Назва стрілки	Опис	Тип
Онлайн операція	Операція купівлі товару в інтернет-магазині за допомогою банківської картки	Input
Історія платежів	Попередні операції в Інтернеті	Input
Законодавство України	Закони України, що регулюють процес проведення онлайн-платежів та взаємодію його учасників	Control
Фільтри системи	Критерії, яким повинні відповідати нешахрайські операції	Control
Стандарт безпеки PCI DSS	Стандарт безпеки даних індустрії банківських платіжних карток	Control
МПС	Міжнародна платіжна система	Mechanism
Веб-додаток	Форми для зв’язку з клієнтом	Mechanism
Банк	Банк, який випустив банківську картку клієнту	Mechanism
Клієнт	Особа, яка проводить платіж в мережі Інтернет	Mechanism
Проведений платіж	Успішно проведена транзакція клієнта	Output
Шахрайська операція	Виявлена шахрайська операція	Output

Контекстна діаграма не дає детального та повного розуміння суті процесу. Тому наступним кроком є декомпозиція контекстної діаграми, тобто розбиття на частини (підпроцеси), щоб глибше розібрати даний процес виявлення шахрайських операцій (рисунок 2.3).

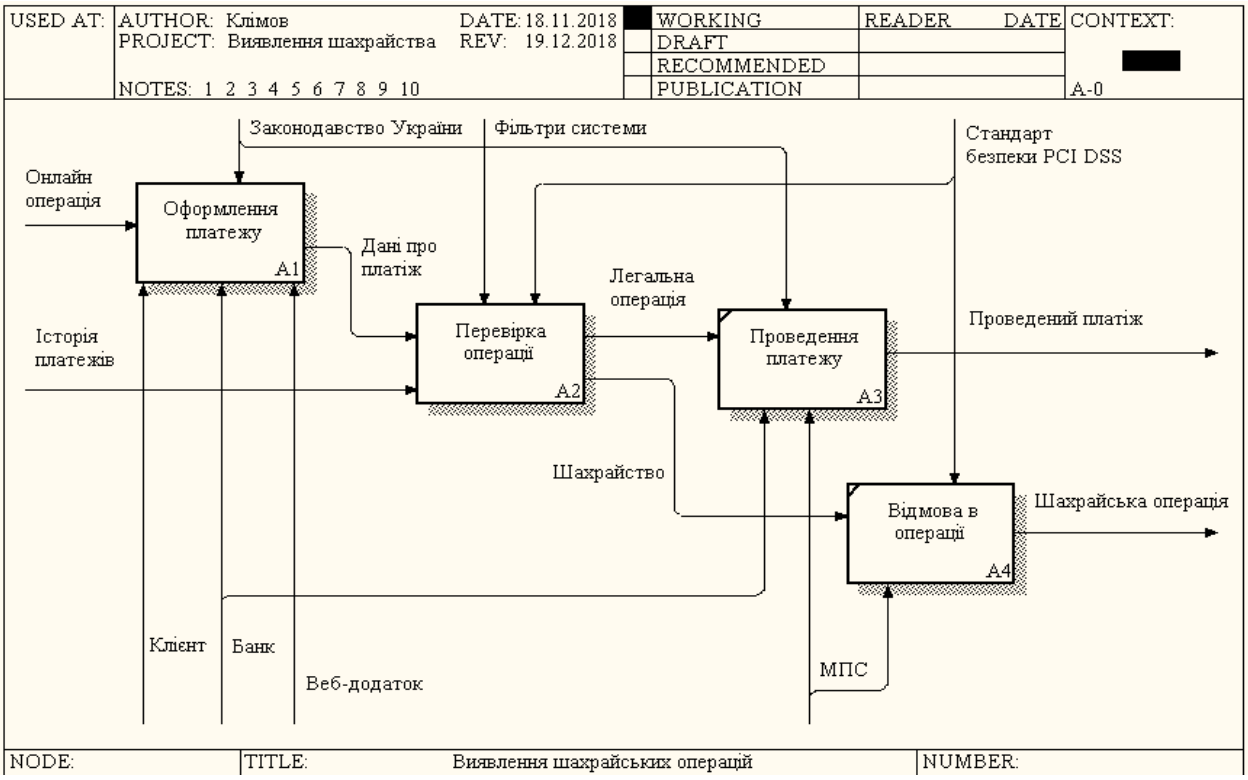

Рисунок 2.3 – Декомпозиція контекстної діаграми

Для поданого рисунка необхідно навести опис робіт (таблиця 2.2).

Таблиця 2.2 – Опис робіт діаграми-декомпозиції

Функціональний блок	Опис	Тип
Оформлення платежу	Заповнення форми купівлі товару в Інтернеті	WORKING
Перевірка операції	Моніторинг операції та аналіз її на можливість шахрайства	WORKING
Проведення платежу	Підтвердження транзакції купівлі	WORKING
Відмова в операції	Операцію визнано шахрайською, транзакція відхилена	WORKING

За аналогією до контекстної діаграми наведемо детальний опис зв’язків між роботами діаграми-декомпозиції контекстної діаграми, де буде вказано назва стрілки, її джерело, призначення та один із чотирьох можливих типів (таблиця 2.3).

Таблиця 2.3 – Опис зв’язків між роботами діаграми-декомпозиції

Назва стрілки	Джерело	Тип	Призначення	Тип
Онлайн операція	Контекстна діаграма		Оформлення платежу	Input
Історія платежів	Контекстна діаграма		Перевірка операції	Input
Законодавство України	Контекстна діаграма		Оформлення платежу, проведення платежу	Control
Фільтри системи	Контекстна діаграма		Перевірка операції	Control
Стандарт безпеки	Контекстна діаграма		Перевірка операції, відмова в операції	Control
Клієнт	Контекстна діаграма		Оформлення платежу	Mechanism
Банк	Контекстна діаграма		Оформлення платежу, проведення платежу	Mechanism
Веб-додаток	Контекстна діаграма		Оформлення платежу	Mechanism
МПС	Контекстна діаграма		Проведення платежу, відмова в операції	Mechanism
Дані про платіж	Оформлення платежу	Output	Перевірка операції	Input
Легальна операція	Перевірка операції	Output	Проведення платежу	Input
Шахрайство	Перевірка операції	Output	Відмова в операції	Input
Шахрайська операція	Відмова в операції	Output	{Border}	Output
Проведений платіж	Проведення платежу	Output	{Border}	Output

Для кращого розуміння процесів потрібно ще далі заглибитись та дослідити підпроцеси «оформлення платежу» та «перевірка операції». Результат декомпозиції першого підпроцесу наведено на рисунку 2.4.

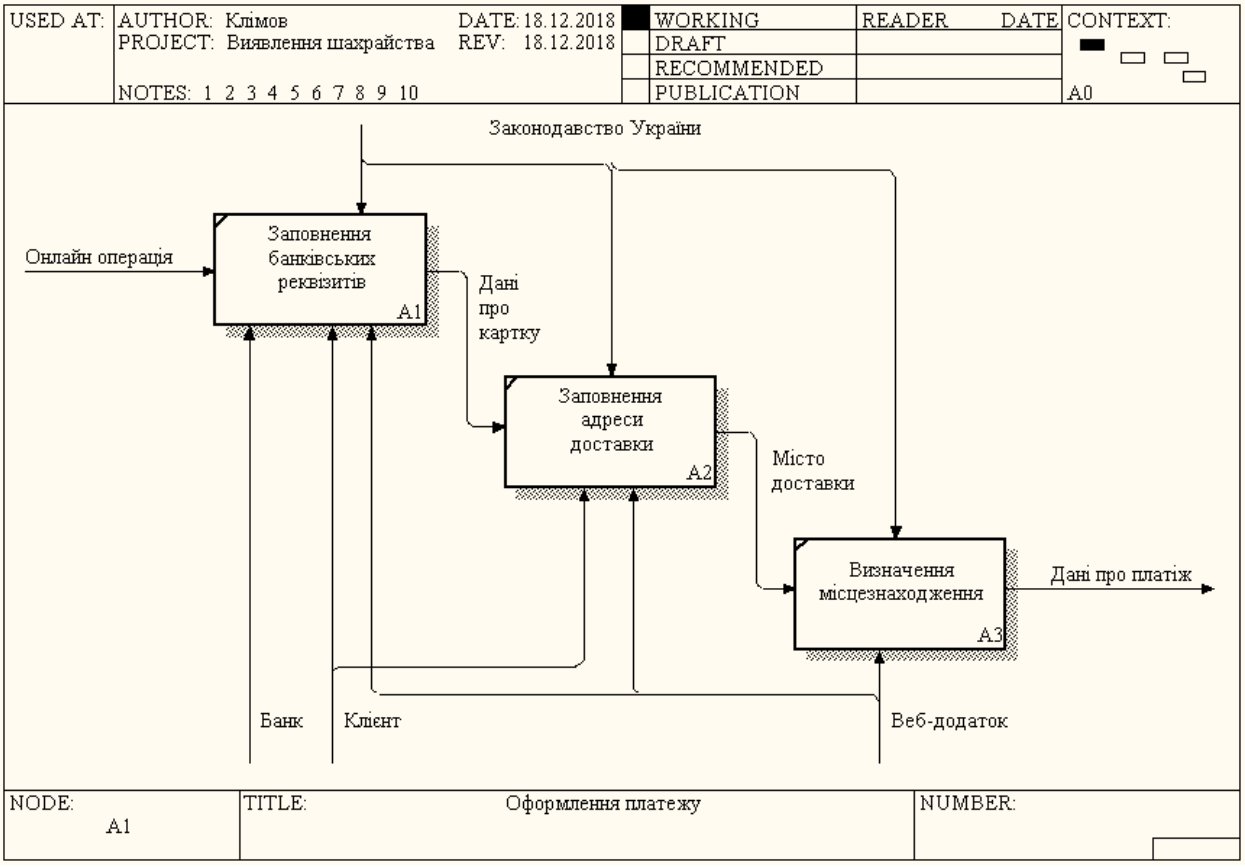

Рисунок 2.4 – Діаграма-декомпозиція процесу «оформлення платежу»

На рисунку 2.4 зображено 3 роботи, які складають процес оформлення платежу. Необхідно навести їх опис у вигляді таблиці 2.4. Також потрібно продемонструвати зв’язків між ними, описавши їх в таблиці 2.5. Зв’язки об’єднують не тільки роботи в цій діаграмі-декомпозиції, а й в батьківській.

Таблиця 2.4 – Опис робіт діаграми-декомпозиції

Функціональний блок	Опис	Тип
Заповнення банківських реквізитів	Процес введення клієнтом даних банківської картки	WORKING
Заповнення адреси доставки	Процес введення регіону та міста, куди відправляти товар	WORKING
Визначення місцезнаходження	Пошук міста, в якому знаходиться клієнт, за допомогою IP-адреси	WORKING

Таблиця 2.5 – Опис зв’язків між роботами діаграми-декомпозиції

Назва стрілки	Джерело	Тип	Призначення	Тип
Онлайн операція	Контекстна діаграма		Заповнення банківських реквізитів	Input
Законодавство України	Контекстна діаграма		Заповнення банківських реквізитів, заповнення адреси доставки, визначення місцезнаходження	Control
Банк	Контекстна діаграма		Заповнення банківських реквізитів	Mechanism
Клієнт	Контекстна діаграма		Заповнення банківських реквізитів, заповнення адреси доставки	Mechanism
Веб-додаток	Контекстна діаграма		Заповнення банківських реквізитів, заповнення адреси доставки, визначення місцезнаходження	Mechanism
Дані про картку	Заповнення банківських реквізитів	Output	Заповнення адреси доставки	Input
Місце доставки	Заповнення адреси доставки	Output	Визначення місцезнаходження	Input
Дані про платіж	Визначення місцезнаходження	Output	Перевірка операції, відмова в операції	Output

Для декомпозиції другого підпроцесу скористаємося нотацією IDEF3. Вона краще підходить для опису процесів на глибоку рівні декомпозиції, тому що зображує послідовність виконання процесів як деякий алгоритм.

Аналогічно нотації IDEF0, в нотації IDEF3 основною одиницею є діаграма. На діаграмі ключову роль становлять одиниці роботи (UnitOfWork), які описують алгоритм процесу. Істотною відмінністю даної нотації є наявність перехресть. Це можуть бути перехрестя злиття (синхронна або асинхронна кон’юнкція) та перехрестя розгалуження (синхронна, асинхронна або виключна диз’юнкція) [47-48]. Особливістю побудови є те, що одиниця роботи може мати лише одну стрілку на вхід та одну на вихід.

IDEF3 діаграму перевірки операцій наведено на рисунку 2.5. Детальний опис поданої діаграми наведемо в наступній таблиці 2.6.

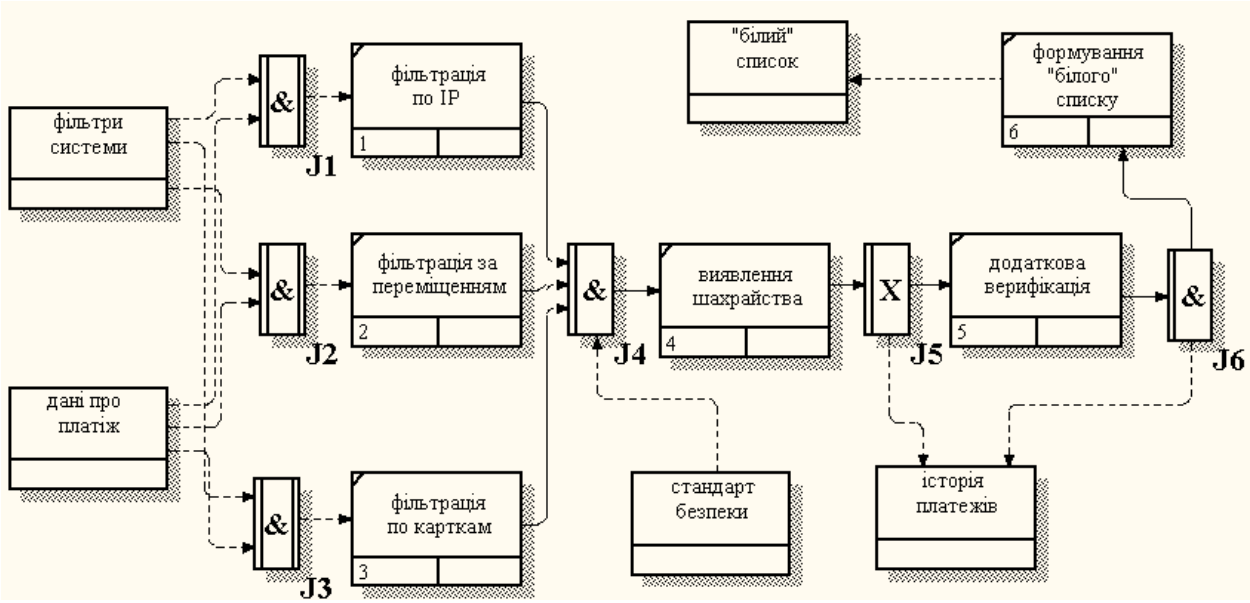

Рисунок 2.5 – IDEF3 діаграма системи виявлення шахрайства

В нотації IDEF3 продемонстровано 6 робіт зв’язаних між собою перехрестями: асинхронної, синхронної кон’юнкції та виключної диз’юнкції. Роботи між собою поєднуються стрілками пріоритету, а із зовнішніми об’єктами – стрілками відношення.

Таблиця 2.6 – Опис робіт діаграми

Функціональний блок	Опис	Тип
Фільтрація по IP	Процес порівняння поточного місця знаходження та адреси доставки	WORKING
Фільтрація за переміщенням	Процес аналізу швидкості переміщення клієнта	WORKING
Фільтрування по картках	Розрахунок унікальних банківських карт	WORKING
Виявлення шахрайства	Узагальнення результатів роботи фільтрів	WORKING
Додаткова верифікації	Підтвердження достовірності особи	WORKING
Формування «білого» списку	Процес верифікації банківських карт та IP-адрес	WORKING
Дані про платіж	Інформація про місце знаходження клієнта, адреса замовлення, минулі платежі	DATABASE
«Білий» список	Карти, платежі за якими не потребують підтвердження	DATABASE
Фільтри системи	Фільтри, які використовуються для виявлення шахрайства	DATABASE
Історія платежів	Список всіх транзакцій по окремій картці	DATABASE
Стандарт безпеки	Вимоги забезпечення безпеки даних банківських карт	DATABASE

2.2 Вибір архітектури та технології для розробки модулю

Враховуючи, що система являє собою модуль автоматизованої системи для онлайн-магазину інтегрованого із банком та платіжною системою потрібно використовувати таку архітектуру і технології, які будуть надійними та безпечними.

Найбільш поширеною технологією для організацій електронної комерції є трирівнева клієнт-серверною архітектура [49-51]. Вона складається з таких компонент: клієнтський додаток (також називають термінал), додаток з'єднаний з сервером, до якого підключена база даних.

Клієнтський рівень – це інтерфейсний додаток верхнього рівня, іншими словами графічне зображення з яким працює кінцевий користувач. На цьому рівні не повинно бути прямих зв'язків з базою даних (обумовлено вимогами безпеки), не повинно бути основної бізнес-логіки (принцип масштабованості) і не потрібно зберігати стан програми [52].

Серверна частина знаходиться на другому рівні, на якому знаходиться основна частина програмної логіки. До неї не входять тільки інформація, що відображається на терміналі, тригери та збережені процедури, які розміщені на третьому рівні [53].

На сервері бази даних (третій рівень) відбувається зберігання даних. Він може бути реляційною або об'єктно-орієнтованою системою керування базою даних [54-55]. Якщо на третьому рівні представлені база даних разом зі збереженими процедурами й тригерами, то на другому рівні створюється програмне забезпечення, яке зв'язує клієнтську частину з прикладною логікою бази даних.

Архітектура системи зображена на рисунку 2.6.

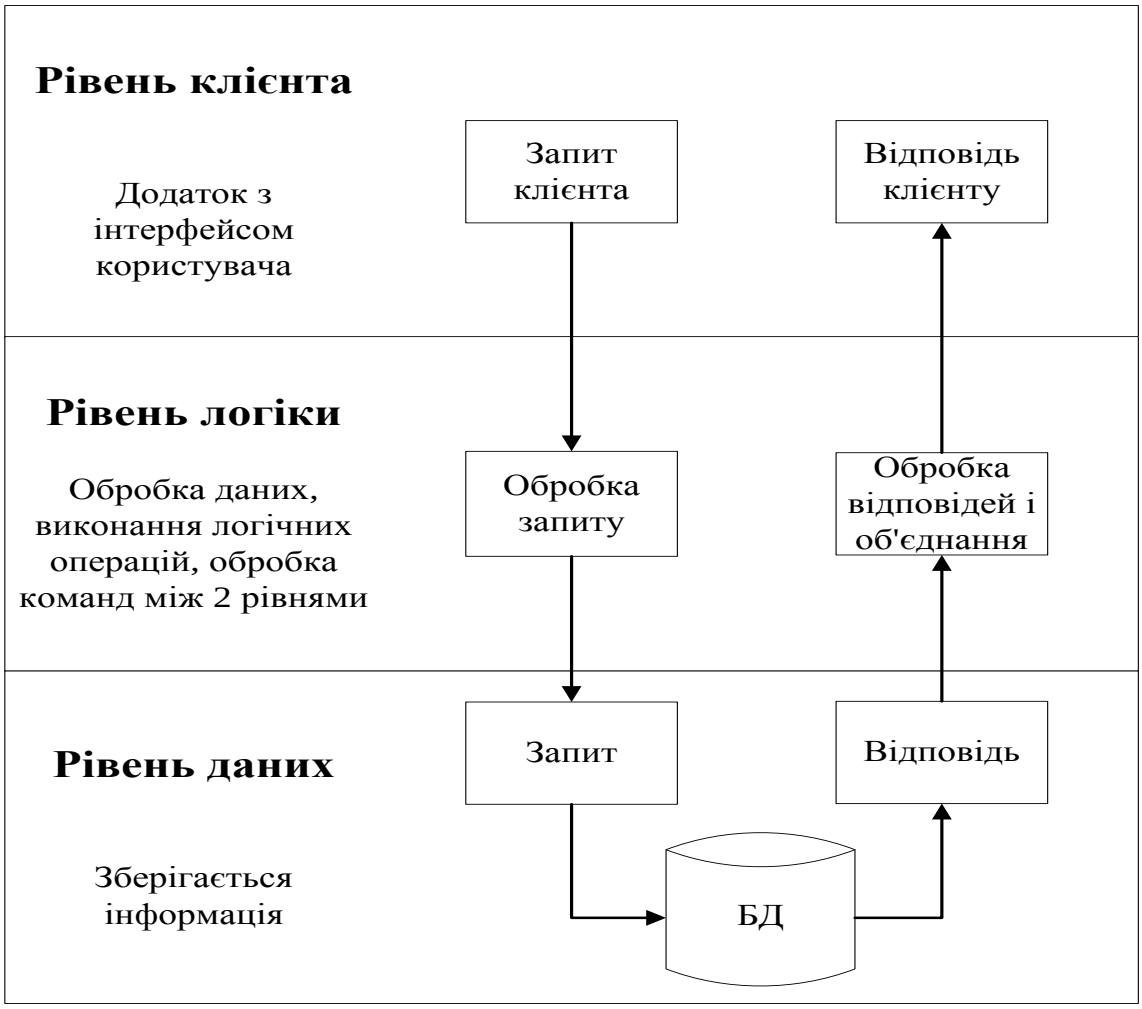

Рисунок 2.6 – Схема архітектурної структури модуля виявлення шахрайських операцій

Переваги даної архітектури:

– масштабованість;

– конфігурованість – ізольованість рівнів один від одного дозволяє швидко і простими засобами змінити налаштування системи при виникненні збоїв або при плановому обслуговуванні на одному з рівнів [56];

– високий рівень безпеки;

– висока надійність;

– низькі вимоги до продуктивності і технічних характеристик терміналів, як наслідок зниження їхньої вартості.

Для реалізації прототипу даної системи використовувалися популярні технології створення веб-додатків [57-62]. Клієнтська частина реалізована з

використанням HTML5, CSS3, JavaScript, JQuery та безкоштовним фреймворком Bootstrap, прикладний сервер – мовою програмування PHP з використанням Ajax-запитів. Для зберігання даних і роботи з ним було обрану систему керування базою даних MySQL.

При написані логіки роботи додатку використовувалося процедурне програмування на PHP. Найпоширеніші функціональні завдання, які виконує PHP є обробка даних форми, генерування динамічних сторінок, створювати/зчитувати cookies [63].

Виділяють три основні галузі застосування PHP [64]:

1. Створення коду для виконання програми на стороні сервера. Це є найголовнішим способом використання PHP. Для виконання цієї функції є необхідними три умови: веб-сервер, інтерпретатор мови PHP (наприклад CGI або серверний модуль) та веб-браузер. Для перегляду результатів виконання PHP-скриптів, необхідно щоб був встановлений веб-сервер і PHP. Тоді можна побачити в браузері HTML-сторінку, яку згенерував сервер.

2. Створення коду для виконання в командній строчці. Можливо створити такий PHP-скрипт, що здатен виконуватися без сервера або веб-браузера. В цьому випадку необхідний парсер PHP. Такий спосіб використовують, коли скрипти повинні виконуватися постійно. Крім того, ці скрипти можуть використовуватися в задачах обробки текстів.

3. Створення додатків для виконання у вікні клієнта. Є інші мови програмування, які краще підходять для створення користувальницького інтерфейсу, проте і PHP передбачає таку можливість. Існує відповідний додаток PHP-GTK для створення подібного програмного забезпечення. Крім цього, він також використовується для створення крос-платформних додатків. PHP-GTK – це спеціальне розширенням PHP, яке необхідно окремо додатково встановлювати.

До ключових переваг PHP відносять підтримку великого кола різних баз даних. Створити код для з’єднання з базою даних можна за допомогою розширення, яке є специфічним для окремої бази даних (наприклад MySQL)

або використати абстракції від бази даних (найпоширенішим є PDO) [65]. Ще одна можливість під'єднатися до бази даних, є Відкритий Стандарт З'єднання Баз Даних (ODBC) [66-67]. В інших випадках можна використовувати cURL або сокети.

MySQL – це розповсюджена безкоштовна система управління базами даних (СУБД), яка дуже часто використовується разом із PHP.

MySQL за структурою є клієнт-серверною системою. Вона містить багатопоточний SQL-сервер, який виконує підтримку різних ЕОМ баз даних та різних клієнтських додатків і бібліотек, засобів адміністрування і різноманітних прикладних програмних інтерфейсів (API) [68-69].

Ajax – це концепція програмування, за допомогою якої зміст сторінки може бути змінено в веб-браузері, без повної її перезавантаження. Ajax забезпечує асинхронне передавання даних між веб-клієнтом та веб-сервером. Асинхронний, оскільки на відміну від класичного зв'язку браузер-сервер, на Ajax, XML і JavaScript надсилають запити на веб-сервер [70]. Сервер може змінити відображений вміст користувачу, не перезавантажуючи сторінку. Це дозволяє веб-додаткам швидше реагувати на дії користувача та мінімізувати трафік. Завдяки цій концепції більше не потрібно, щоб статичні, незмінні дані постійно перезавантажувалися.

Bootstrap – це фреймворк для створення інтерфейсу веб-сайтів. Шаблони HTML та CSS призначені для представлення різних елементів веб-сайту [71]. До них входять форми, кнопки, таблиці, навігація та розмітка сітки сторінки. Крім того, модулі JavaScript дозволяють включати в себе взаємодію (наприклад, слайд-шоу зображень, вкладки та діалоги). Крім того, Bootstrap пропонує всі необхідні умови для розробки адаптивних веб-дизайнів, які потім можуть бути оптимально відображені на смартфонах або планшетах [72].

2.3 Підсистеми забезпечення функціональної частини модулю

2.3.1 Функціональна структура задачі

Головною функцією автоматизованого модулю є аналіз платіжних операцій з картками для виявлення шахрайський операцій. Крім цього він повинен мати можливість для виконання інших функцій. Всі можливості можна умовно поділити на три частини: для клієнтів електронної комерції, для онлайн-магазину та для банку.

Для клієнта система повинна надавати можливість:

- ввести банківські реквізити;
- ввести адресу, на яку відправити товар;
- підтвердити особистість за допомогою коду із СМС.

Існування даної системи для справжнього власника банківської картки дасть впевненість, що його кошти не зможуть вивести через магазин, в якому функціонує подібна система. А для шахраїв навпаки, буде відсутня можливість провести шахрайську операцію.

Онлайн-магазин отримує можливість виконувати тільки ті замовлення, які не будуть шахрайськими. Всі замовлення, які були класифіковані як шахрайські операції не будуть надходити до внутрішньої системи магазину і відволікати співробітників від їх роботи.

Функціональні можливості для банку будуть наступні:

- переглядати підозрілі операції з картками клієнтів свого банку;
- фільтрувати та аналізувати отримані дані;
- вносити зміни в дані підозрілих транзакцій.

Результат роботи системи буде доступний співробітнику банку для перегляду та аналізу. Отримані дані про шахрайські операції можна використовувати для виявлення та попередження можливих шахрайських операцій.

2.3.2 Технічне забезпечення

Для функціонування системи потрібно мати сервер додатків та сервер бази даних відповідної конфігурації, які наведені в таблицях 2.7 та 2.8.

Таблиця 2.7 - Рекомендована конфігурація сервера додатків

Компонент	Характеристика
Процесор	4-ядерний Intel Xeon-D 1521 @ 2.4 ГГц
Жорсткий диск	120Гб (SSD), 2x2 ТБ (HDD)
Оперативна пам’ять	32 ГБ
Швидкість LAN	10 Гбіт/с
Операційна система	Windows Server 2012 (x64)

Таблиця 2.8 - Рекомендована конфігурація сервера бази даних

Компонент	Характеристика
Процесор	4-ядерний Intel Xeon-D 1521 @ 2.4 ГГц
Жорсткий диск	120Гб (SSD), 2x2 ТБ (HDD)
Оперативна пам’ять	32 ГБ
Швидкість LAN	10 Гбіт/с
Операційна система	Windows Server 2012 (x64)

Для використання системи клієнтом, йому потрібно мати доступ в мережу Інтернет та термінал, зі встановленою операційною системою та браузером.

Технічні вимоги для роботи з системою для співробітника банку такі самі, як і для клієнта: термінал з доступом до Інтернету.

2.3.3 Програмне забезпечення

Для реалізації прототипу автоматизованого модуля було обрано портативний локальний сервер Open Server. Для налагодження скриптів для різного оточення Open Server пропонує на вибір відразу два види HTTP серверів, різні версії PHP і СУБД модулів і можливість швидкого переключення між ними. За допомогою Open Server можна запустити / зупинити сервер або відкрити потрібний домен.

Open Server являє собою WAMP комплекс:

– Windows – операційна система, для роботи в якій призначений даний локальний сервер;

– Apache – web-сервер, який запускається при старті програми Open Server;

– MySQL – дуже популярна система управління базами даних;

– PHP – інтерпретатор серверної мови програмування.

Під ним мається на увазі комплекс програм, які дають можливість працювати безпосередньо з сайтами без хостингу. Цей локальний додаток підходить для розробки, тестування, оновлення сторінок в мережі, коли потрібно ввести спочатку зміни, перевірити їх, а вже потім залити сайт на хостинг.

Особливості ПЗ Open Server:

– докладний перегляд логів всіх компонентів в реальному часі;

– вибір HTTP, СУБД і PHP модулів в будь-якому поєднанні;

– підтримка SSL і доменів;

– створення локального піддомена;

– підтримка доменних покажчиків, та їх налаштування;

– доступ до доменів і швидкий доступ до шаблонів конфігурації;

– багатомовний інтерфейс (Російська, Українська, Білоруська, Англійська мови).

2.3.4 Організаційне забезпечення

Впровадження автоматизованої системи не впливає на організаційну структуру організації електронної комерції. Посади та обов’язки працівників магазину залишаться такими самими.

Використання даного модулю призведе до змін в структурі банка. Для аналізу підозрілих транзакцій та їх обробки необхідний співробітник, який буде виконувати дані задачі. Якщо у складі банка немає особи, яка може додатково виконувати ці функції це призведе до додаткових витрат.

Проте, автоматизована система мінімізує необхідний час для виконання поданих функцій, тому планується, що за це буду відповідати менеджер по роботі з клієнтами.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОТОТИПУ МОДУЛЮ ВИЯВЛЕННЯ ШАХРАЙСЬКИХ ОПЕРАЦІЙ З БАНКІВСЬКИМИ КАРТКАМИ

3.1 Структура та особливості реалізації інформаційного забезпечення

Реалізація автоматизованого модулю передбачає також створення інформаційного забезпечення. В даному випадку воно буде у вигляді реляційної бази даних. База даних буде зберігати тільки необхідну інформацію, яка пов’язана із перевіркою платежу на шахрайство. Інформаційне забезпечення стосовно перевірки банківської картки на вірність терміну діє та CVV2 буде знаходитися у відповідного банку.

Усю інформацію, з якою працює автоматизований модуль можна виокремити у три групи:

- інформація, яку вводить клієнт;
- інформація, яка зберігається у системі (історія транзакцій);
- інформація, що виводиться співробітнику банка.

Для зберігання поданої інформації необхідно створити наступні сутності:

- «clients» – інформація про клієнтів;
- «cards» – інформація про банківські карти;
- «transactions» – інформація про всі транзакції;
- «frauds» – інформація про транзакції, які позначили шахрайськими;
- «location» – довідник місцезнаходження від IP-адреси;
- «location_ua» – довідник місцезнаходження в Україні від IP-адреси.

Сутності зв’язані між собою на відношеннях. Створення зв’язку між таблицями передбачає використання первинних та зовнішніх ключів. Бувають 3 види відношень: «один до одного», «один до багатьох» та «багато до багатьох» [73]. Тепер необхідно для кожної сутності створити таблицю, в якій кожен стовпчик буде відповідати за певний атрибут. Визначити в

таблиці первинний ключ, та за необхідності зовнішній ключ і відповідно зв’язки між ними.

В результаті створення бази даних, було отримані наступні таблиці з відповідними структурами (таблиця 3.1 – 3.6).

Таблиця 3.1 – Структура таблиці «clients»

№	Назва атрибута	Тип даних	Обмеження	Призначення атрибута
1	clientID	int(11)	AUTO_INCREMENT	Первинний ключ
2	fname	varchar(100)	NOT NULL	Ім’я клієнта
3	sname	varchar(100)	NOT NULL	Прізвище клієнта
4	patronymic	varchar(100)	NOT NULL	По-батькові клієнта
5	telephone	varchar(12)	NOT NULL	Номер телефона

Таблиця 3.2 – Структура таблиці «cards»

№	Назва атрибута	Тип даних	Обмеження	Призначення атрибута
1	cardID	varchar(16)	NOT NULL	Первинний ключ, номер банківської картки
2	clientID	int(11)	FOREIGN KEY, NOT NULL	Зовнішній ключ

Таблиця 3.3 – Структура таблиці «transactions»

№	Назва атрибута	Тип даних	Обмеження	Призначення атрибута
1	transactionID	int(11)	AUTO_INCREMENT	Первинний ключ
2	cardID	varchar(16)	FOREIGN KEY, NOT NULL	Зовнішній ключ, номер банківської картки
3	time	datetime	NOT NULL, CURRENT_ TIMESTAMP	Дата та час транзакції
4	region	varchar(128)	NOT NULL	Регіон доставки
5	ort	varchar(128)	NOT NULL	Місто доставки
6	ip	int(10)	NOT NULL, UNSIGNED	IP-адреса транзакції
7	fraud	boolean	NOT NULL, DEFAULT=0	Виявлено шахрайство

Таблиця 3.4 – Структура таблиці «frauds»

№	Назва атрибута	Тип даних	Обмеження	Призначення атрибута
1	fraudID	int(11)	AUTO_INCREMENT	Первинний ключ
2	transactionID	int(11)	FOREIGN KEY, NOT NULL	Зовнішній ключ
3	code	int(11)	NOT NULL	Код підтвердження клієнта
4	reason	varchar(128)	NOT NULL	Причина виявлення шахрайства

Таблиця 3.5 – Структура таблиці «location»

№	Назва атрибута	Тип даних	Обмеження	Призначення атрибута
1	ip_from	int(10)	UNSIGNED	Початок діапазону IP-адреси
2	ip_to	int(10)	UNSIGNED	Кінець діапазону IP-адреси
3	country_code	char(2)	-	Код країни
4	country_name	varchar(64)	-	Назва країни
5	region_name	varchar(128)	-	Назва регіону
6	city_name	varchar(128)	-	Назва міста
7	latitude	double	-	Географічна широта
8	longitude	double	-	Географічна довгота
9	zip_code	varchar(30)	-	Поштовий індекс

Таблиця 3.6 – Структура таблиці «location_ua»

№	Назва атрибута	Тип даних	Обмеження	Призначення атрибута
1	ip_from	int(10)	UNSIGNED	Початок діапазону IP-адреси
2	ip_to	int(10)	NOT NULL	Кінець діапазону IP-адреси
3	region_name	varchar(128)	NOT NULL	Назва регіону
4	city_name	varchar(128)	NOT NULL	Назва міста
5	latitude	double	NOT NULL	Географічна широта
	longitude	double		Географічна довгота

Таблиці створюються таким чином, щоб сутності перебували у 3 нормальній формі. В таблицях всі атрибути знаходяться у функціональній залежності від первинного ключа і не містять транзитивних залежностей [74].

Відношення між таблицями були встановлені наступні:

- «clients» – «cards» – відношення один до багатьох;
- «cards» – «transactions» – відношення один до багатьох;
- «transactions» – «frauds» – відношення один до одного.

Схематичне зображення всіх сутностей з атрибутами та зв’язків між ними прийнято подавати у вигляді схеми база даних або моделі сутність-зв’язок. Програмне забезпечення Open Server забезпечує таку можливість у спеціальному вікні «Дизайн» (рисунок 3.1).

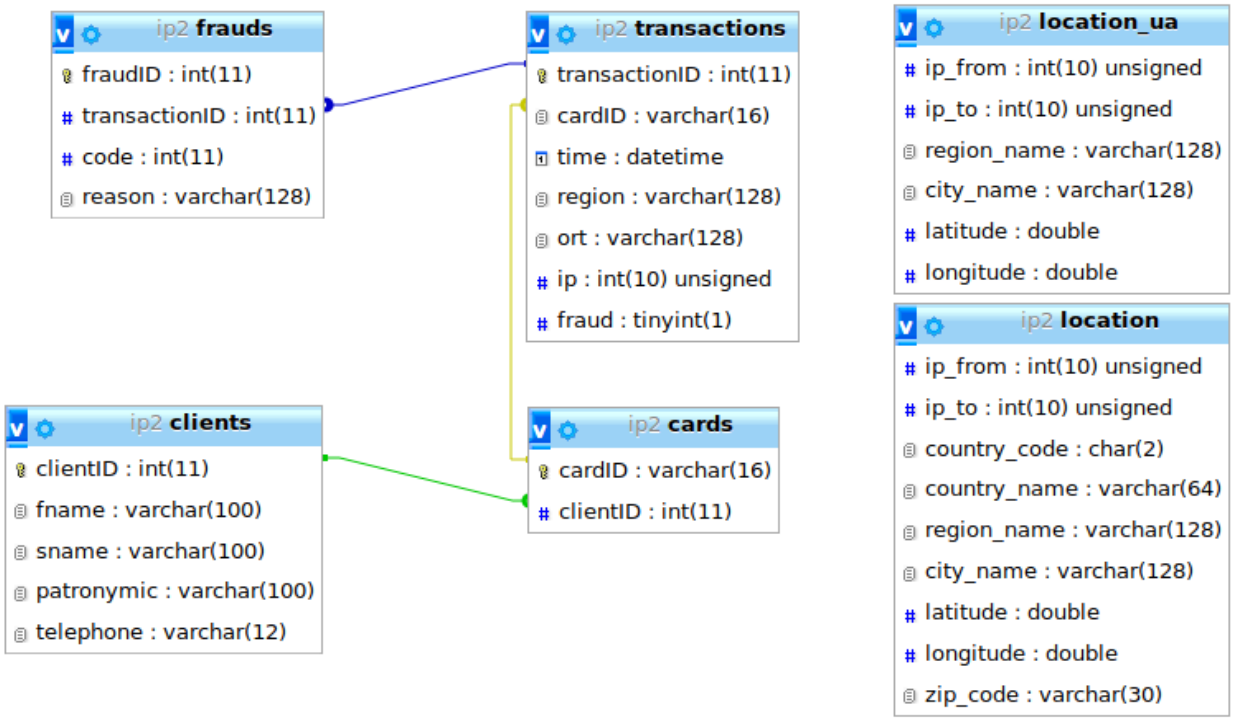

Рисунок 3.1 – Схема бази даних автоматизованого модуля виявлення шахрайських операцій з картками

Зв’язок між сутностями location та location_ua на рівні бази даних не передбачений, тому що вони являють собою довідники місцезнаходження без первинного ключа. В них немає атрибуту, з яким можна поєднати сутність transactions, так як шукана ip-адреса повинна знаходитись в діапазоні, а не дорівнювати певному значенню. Сценарій створення бази даних, усіх таблиць з атрибутами та зв’язків між ними наведено в Додатку Б.

3.2 Структура та особливості реалізації алгоритмічного забезпечення

Логіка будь-якого додатку полягає в його функціональних можливостях та алгоритмічному забезпеченні. Головна ідея цього модулю – це виокремлення шахрайських операцій та робота з ними. Враховуючи це, найголовнішим є аналіз онлайн-платежу за різними параметрами і винесення результат за кожним компонентом системи. Внутрішня система аналізу складається з трьох компонент (фільтрів), перевірку через які повинен пройти платіж. На кожному етапі система повертає результат, чи є операція шахрайською. Роботу даних фільтрів можна зобразити у вигляді блок-схем (рисунок 3.2-3.5).

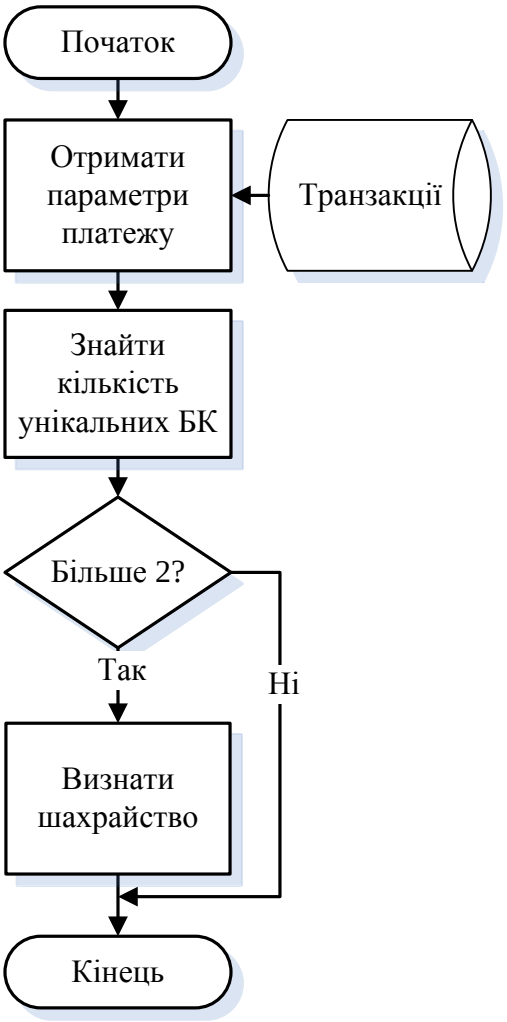

Рисунок 3.2 – Блок-схема алгоритму фільтрування за кількістю карт

Логіка алгоритму, продемонстрованого на рисунку 3.2, полягає у підрахунку кількості банківських карт. При виконані операції з певної IP-адреси, програма визначає зі скількох інших банківських карт виконувалися онлайн-операції за останню добу. Якщо унікальних карт буде більше двох, то операція вважається шахрайською.

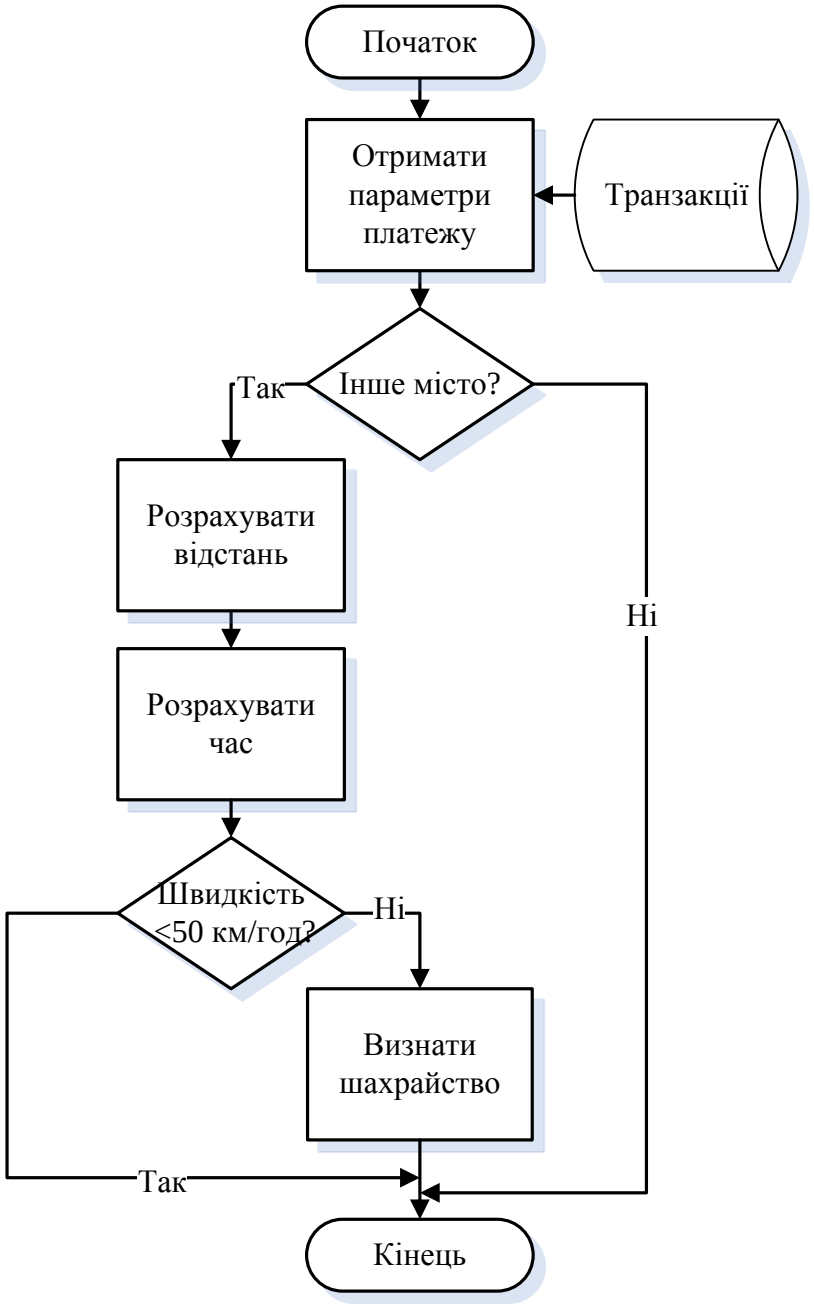

Рисунок 3.3 – Блок-схема алгоритму фільтрування за швидкістю переміщення клієнта

Рисунок 3.3 демонструє алгоритм, за яким відбувається аналіз переміщення клієнта. Розраховується швидкість, з якою клієнт подолав відстань за час з моменту останньої операції і порівнюється з критичним значенням у 50 км/год. Якщо швидкість клієнта була більшою, то це є підозрілим і операція вважається шахрайською.

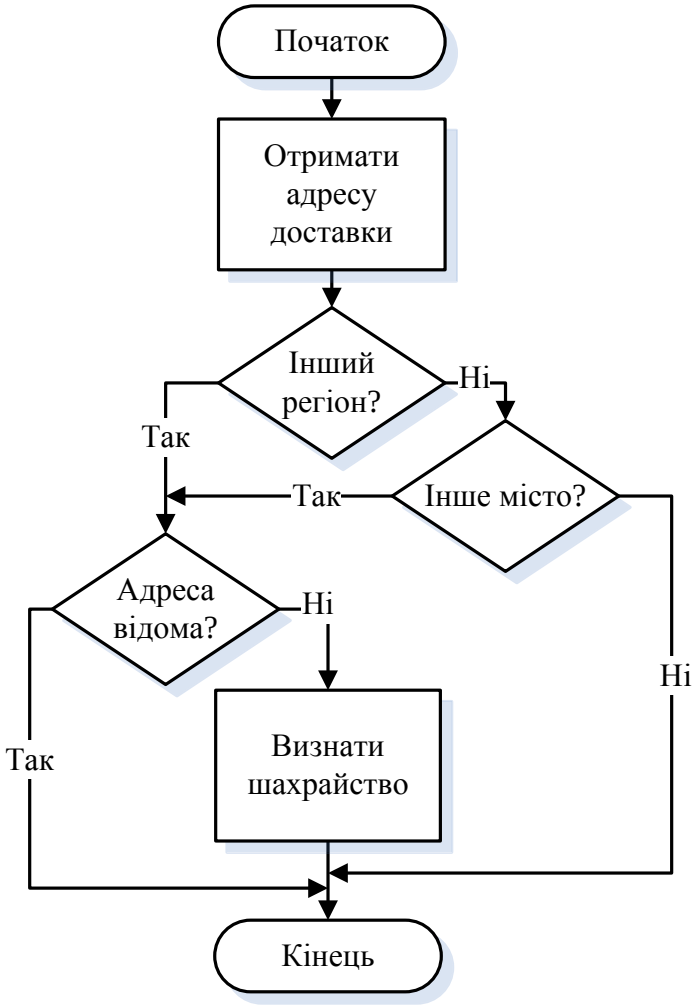

Рисунок 3.4 – Блок-схема алгоритму фільтрування за місцем знаходження та доставки

Робота даного фільтру полягає у порівнянні поточного регіону та міста, яке визначається за IP-адресом та місто, в яке замовлено доставку товару. У випадку різних значень додатково перевіряється, чи куплялися товари раніше на ту адресу. Якщо дана адреса вже була збережена у транзакціях клієнта, то операція не вважається шахрайською.

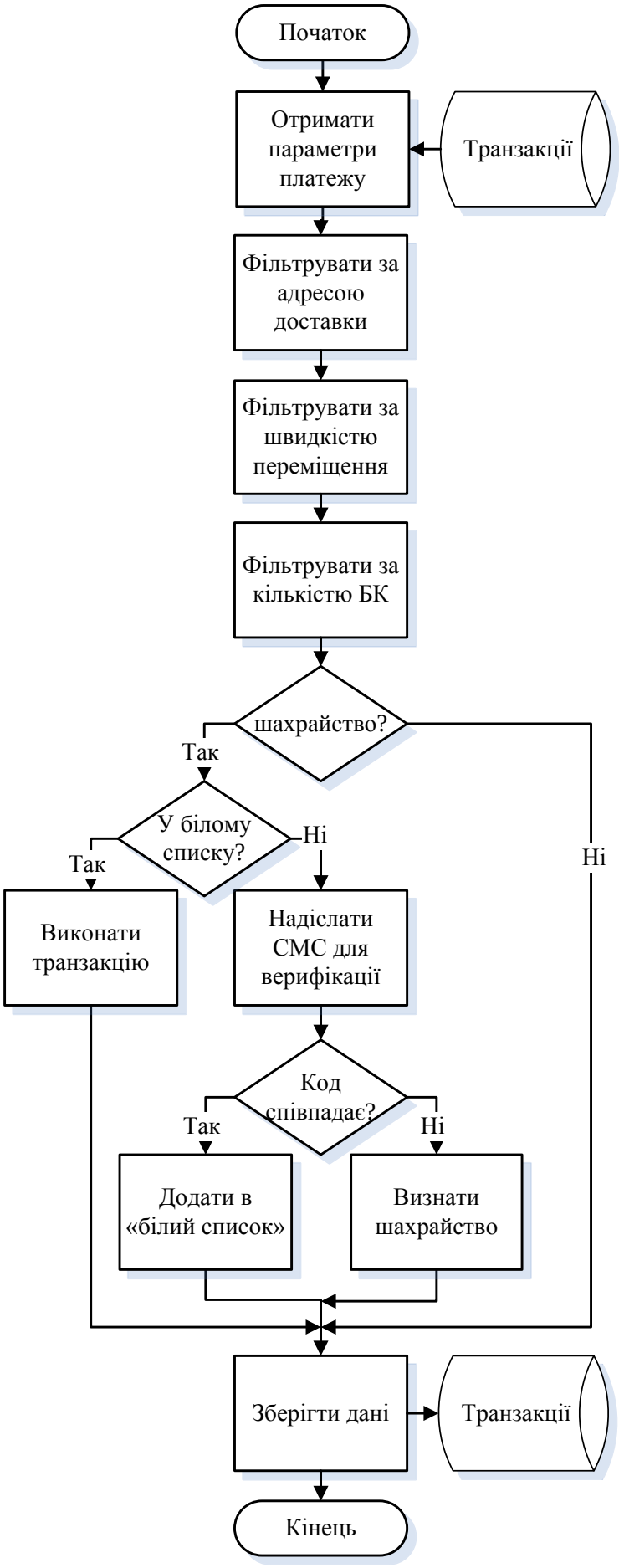

Рисунок 3.5 – Загальний вигляд алгоритму модулю виявлення шахрайства

На рисунку 3.5 зображено весь алгоритм за яким виконується процес виявлення шахрайських операцій – фільтрування платежу за 3 критеріями, верифікація у випадку необхідності та зберігання результатів в базі даних.

Автоматична система починається з форми онлайн-платежу, яку заповнює клієнт. Після цього методом POST дані відправляються до системи з фільтрами для виявлення шахрайських операцій. Кожен фільтр зберігає причину класифікації платежу як шахрайського, якщо вона є.

Для наочності наведемо частину програмного коду, який відповідає за аналіз часу та місцем знаходження клієнта між платежами:

Лістинг 3.1 – Фільтрація платежу за швидкістю переміщення клієнта

```
$time_dif = (strtotime("now")-strtotime($res['time']))/3600;
$result = mysqli_fetch_assoc(mysqli_query($link,$sql));
$lat1 = $res['latitude'];
$long1 = $res['longitude'];
$lat2 = $result['latitude'];
$long2 = $result['longitude'];
$dist = calculateTheDistance($lat1, $long1, $lat2, $long2)/1000;
if($time_dif < $dist/50) {
    $fraudrisk=1;
    $fraud[] = "ошибка во времени";
}
```

Перший рядок розраховує скільки годин пройшло з моменту останнього платежу. Другий рядок повертає географічні координати поточного місцезнаходження. У рядках 3-4 записується у змінні координати місцезнаходження у момент останнього платежу, у рядках 4-6 – поточного місця. У 7 рядку викликається власна функція calculateTheDistance, яка розраховує відстань між містами у кілометрах. У 8 рядку перевіряється, чи достатньо було часу для проходження розрахованої відстані при швидкості у 50 кілометрів за годину. Якщо часу недостатньо, то записується, що платіж шахрайський (рядок 10) і його причину (рядок 11).

Як зазначено на рисунку 3.5, після виконання аналізу за допомогою 3 фільтрів, повертається результат про те, чи є операція шахрайською. Якщо система класифікує її такою, то перевіряється достовірність клієнта –

звіряються дані платежу з «білим списком» та відправляється клієнту СМС з кодом. Після введення отриманого коду у форму, платіж підтверджується і клієнт повертається до початкової сторінки оформлення платежу. Для зменшення надмірного відправлення СМС клієнтам з метою додаткової верифікації створюється «білий список». Він являє собою перелік унікальних банківських карт та IP-адрес, операції за якими спочатку були виявлені як шахрайські, але потім пройшли верифікацію. Він формується динамічно запитом до бази даних. Тому при повторному проведені платежу протягом 3 годин не потрібно буде заново підтверджувати особистість.

Після виконання алгоритму інформація зберігається в базі даних. Операції які позначені, або були позначені як шахрайські виводяться в додаток, який використовує співробітник банку.

Програмний код алгоритмічного забезпечення наведено в Додатку В. У лістингу В.1 продемонстровано програмний код, який виконує процес аналізу операції. В лістингу В.2 записана функція, яка розраховує відстань між містами. Код з лістингу В.3 використовується для збереження результатів.

3.3 Контрольний приклад створеного прототипу

Автоматизований модуль буде мати 2 частини, призначені для різних користувачів. Перша частина створена для клієнтів електронної комерції, які хочуть здійснити платіж за допомогою кредитної картки. У своєму браузері клієнт буде бачити форму, в якій йому необхідно заповнити дані про банківську картку та адресу доставки товару. Всі поля, окрім квартири є обов’язковими для заповнення. Поля область та місто доставки є вибірковими, при цьому назва міст динамічно змінюються зі зміною області.

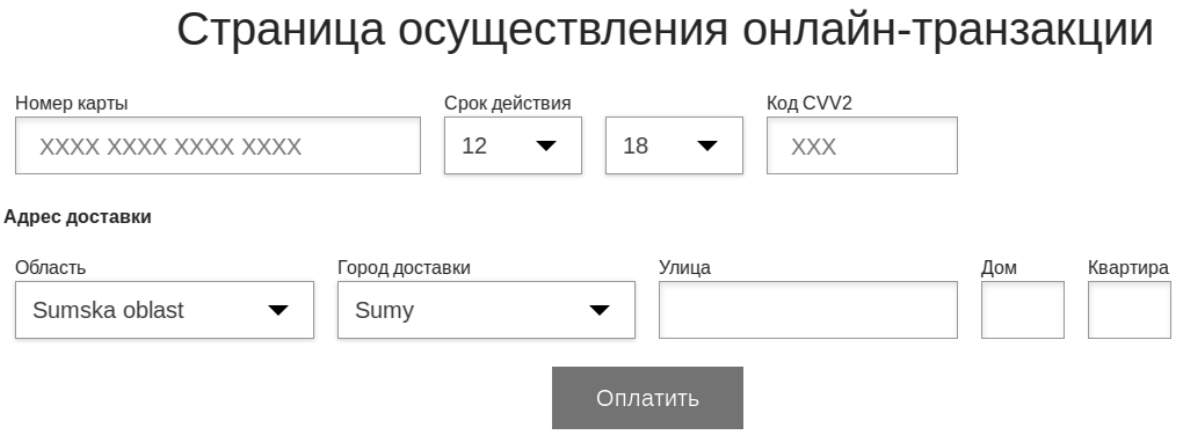

Рисунок 3.6 – Вікно здійснення онлайн-платежу

Після натискання кнопки виконується алгоритмічна частина додатку, в якій перевіряється чи є даний платіж шахрайським. Якщо у системи немає зауважень до цього платежу, то транзакція передається на виконання у платіжну систему, а клієнт повертається на початкову сторінку магазину електронної комерції.

У випадку виявлення шахрайства виконання транзакції призупиняється і виконується запит до SMS API Service. На мобільний телефон клієнта приходить повідомлення із кодом, який необхідно ввести у форму (рисунок 3.7). Після введення вірного коду платіж перестає вважатися шахрайським, транзакція виконуються і клієнт повертається на початкову сторінку. Код запиту до API для верифікації наведено в лістингу 3.2.

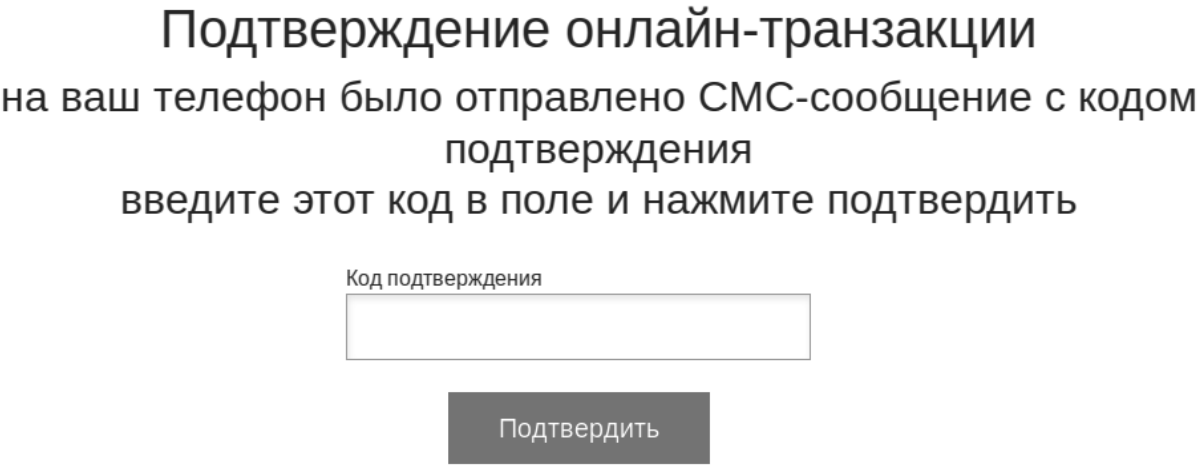

Рисунок 3.7 – Вікно підтвердження платежу

Лістинг 3.2 – Програмний код відправки коду підтвердження

```
$sql = "SELECT telephone FROM user
     INNER JOIN cards c ON c.userID = user.userID
     WHERE c.cardID = " . $cardID;
mysqli_query($link, $sql);
$result = mysqli_fetch_assoc(mysqli_query($link,$sql));
$apiKey = urlencode('cqrSX9lns-nVyN2k3Bfk8ihMXZYGsHSZMvKplNuP');
$numbers = array($result['telephone']);
$numbers = implode(',', $numbers);
$sender = urlencode('UABS');
$message = 'Confirm payment. Do not give this code to anyone: '
. $code;
$data = array('apikey' => $apiKey, 'numbers' => $numbers,
"sender" => $sender, "message" => $message, "unicode" => true);
$ch = curl_init('https://api.txtlocal.com/send/');
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$response = curl_exec($ch);
curl_close($ch);
```

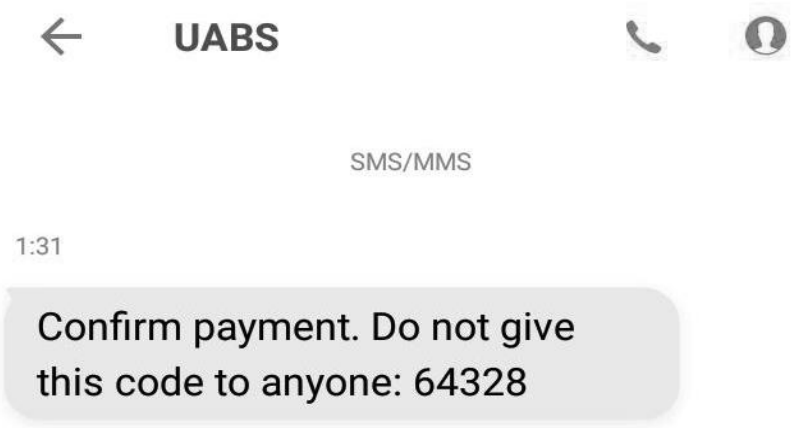

Рисунок 3.8 – СМС з кодом підтвердження

На цьому робота додатку з клієнтом магазину електронної комерції завершується. Друга частина додатку призначена для роботи співробітника банку. Він отримує перелік всіх платежів за участі свого банку, які були визначені як шахрайські операції. Співробітник побачить прізвище та ім'я клієнта, картковий рахунок, телефон, дату проведення платежу, причину визначення його як шахрайського та поточний статус. Отриману інформацію можна фільтрувати по стовпцях. Крім того значення стовпців дати, номера телефону та підтвердження платежу може змінювати відразу в цьому вікні.

Результат работы модуля
операции, которые вызывают подозрения

#	Клиент	Карта	Дата и время	Причина отмены платежа	Операция мошенническая	Телефон
		5168	дд . мм . гггг	---	--	
1	Павлусик Андрей	5168739112345678	2018-10-31	новый город доставки	Нет	380669272071
2	Климов Сергей	5168757399128671	2018-11-19	ошибка во времени	Да	380957684065
3	Климов Сергей	5168757399128671	2018-11-20	разные регионы, ошибка во времени	Нет	380957684065
4	Павлусик Андрей	5168739112345678	2018-11-20	много карт по 1 IP	Да	380669272071
5	Павлусик Андрей	5168739112345678	2018-11-20	разные регионы	Нет	380669272071

Рисунок 3.9 – Вікно виведення шахрайських операцій

Програмний код створення кожної сторінки автоматизованого модуля наведено в Додатку Г. В лістингу Г.4 наведено код JavaScript, який використовується для роботи з даними у вікні виведених результатів.

Важливим для роботи є програмний код, що реалізує зміну інформації в таблиці веб-додатку у режимі реального часу (лістинг Г.5).

3.4 Оцінка очікуваного ефекту від впровадження системи автоматизації

Розробка автоматизованого модуля повинна бути корисною для продавця онлайн-магазину, який буде їм користуватися. Корисність буде виражатися в оцінці економічного ефекту від впровадження поданої технології. Ефективність автоматизованої системи визначається співставленням результатів від функціонування системи і витрат усіх видів ресурсів, необхідних для її створення і розвитку. Якщо впровадження та експлуатація системи буде коштувати більше, ніж кошти які вдасться заощадити, то дану систему не слід використовувати у такому магазині.

Розробка автоматизованої системи створює соціальний та економічний позитивний ефект для організації, яка її буде використовувати. Джерелом соціального ефекту є збільшення задоволеності клієнтів магазину електронної комерції, підвищення їх довіри до магазину та банку. Збільшення прибутків є джерелом економічного ефекту.

Ефект будемо вимірювати за допомогою ефективності, тобто відношення отриманого результату до затрат. Тому факторами економічної ефективності виступають наступні:

- скорочення кількості втрачених товарів;
- скорочення кількості штрафів;
- збільшення лояльності клієнтів до магазину.

Факторами соціальної ефективності системи є:

- більша надійність та захищеність онлайн-платежів;
- збільшення лояльності клієнтів до магазину та банку;
- покращення взаємовідносин між банком та магазином.

Економічна ефективність буде виражатись у вигляді річної економії підприємства. Для розрахунку річної економії необхідно розрахувати капітальні витрати на автоматизацію, витрати підприємства на обробку інформації до автоматизації та після. Вхідними показниками для оцінки річної економії є витрати на фонд оплати праці, експлуатаційні витрати та доходи від продажів.

Значення економії при автоматизації визначається за формулою 3.1

$$S = B_б - B_п, \tag{3.1}$$

де $B_б$ – приведені до одного року витрати на функціонування інтернет-магазину до впровадження автоматизації;

$B_п$ – приведені до одного року витрати на функціонування інтернет-магазину після впровадження автоматизації.

$B_б$ будуть складати річні витрати підприємства на заробітну плату програміста-адміністратора, втрати від повернення платежів та можливі штрафні санкції платіжної системи внаслідок шахрайських операцій.

Місячна заробітна плата одного співробітника становить 6500 грн. У рік він буде отримувати 78000 грн. Штраф за чарджбек становить від 20 до 100 доларів США. У середньому чарджбек через шахрайство відбувається 2 рази на 1000 транзакцій. У рік таких випадків буде 51. Через це магазин втратить приблизно 109140 грн. у рік – 23460 грн. за повернення платежів (при середньому чекові 460 грн.) та 85680 грн. штрафів. При збільшені клієнтів та транзакцій розмір втрат також буде збільшуватися.

Після впровадження системи кількість шахрайських замовлень до магазину буде потрапляти менше, проте це не сильно вплине на кількість роботи працівника та його заробітну плату. Найголовніше це те, що використання розробленого модулю дозволить знизити кількість шахрайських операцій на 15% до 44 випадків на рік. В цьому випадку втрати внаслідок шахрайства зменшиться до 94160 грн. у рік (20240 грн. повернення та 73920 грн. штрафів). Крім того, лояльність постійних клієнтів збільшить дохід на 3% (35190 грн. прибутку при 10% нормі прибутку). Значення економії після впровадження системи буде становити:

$$S = 78000 + 109140 - (78000 + 94160) + 35120 = 50100 \text{ (грн.)} \qquad (3.2)$$

Тепер необхідно розрахувати вартість капітальних витрат, які включають в себе витрати на програмне та апаратне забезпечення, проектування, програмування, впровадження та налаштування системи.

Усі онлайн-магазини вже мають необхідне апаратне забезпечення, яке задовольняє вимоги розробленого модулю, програмне забезпечення PHP та MySQL також вже є (надається продавцю хостингом). Можливо потрібно буде покращити тарифний план для отримання більше вільного місця, додаткової бази даних та пришвидшення виконання запитів. Вартість

покращеного тарифного плану буде становити на 700 грн. більше на рік. Додатково необхідно встановити сервер бази даних. Його вартість відноситься до капітальних витрат (23970 грн.). Витрати, які пов'язані зі створенням та інтегруванням системи виражаються у заробітній платі програміста за договором субпідряду. Вартість такої роботи буде становити 14000 грн. (місячна заробітна плата програміста рівня junior згідно з даними DOU.ua [75]. Додатково прийдеться ще платити за сервіс відправки SMS повідомлень. Вартість відправки одного повідомлення по Україні на сервісі textlocal.com становить 72 копійки. За рік витрати будуть становити 7200 грн. Отже, загальний обсяг капітальних витрат буде становити 700+14000+7200+23970=45870 грн.

Річний економічний ефект розраховується за формулою 3.3

$$E_y = S - C \cdot r_n, \tag{3.3}$$

де S – річна економія в результаті впровадження АІС;

C – капітальні витрати на автоматизацію;

r_n – нормативний коефіцієнт окупності капітальних вкладень (в галузі ІТ він становить 0,33).

$$E_y = 50100 - 45870 \cdot 0{,}33 = 34963 \text{ (грн.)} \tag{3.4}$$

Коефіцієнт економічної ефективності капітальних витрат визначається за формулою 3.5. Термін окупності розраховується обернено до неї (3.6).

$$R_{ce} = \frac{S}{C} = \frac{50100}{45870} = 1{,}092 \tag{3.5}$$

$$P_p = \frac{C}{S} = \frac{45870}{50100} = 0{,}916 \text{ (років)} \tag{3.6}$$

Згідно з розрахунками, автоматизований модуль повністю окупиться за рік. Слід нагадати, що автоматизований модуль окрім економічного ефекту, він виконую свою найголовнішу функцію, яку важко виміряти у грошових одиницях. Виявлення шахрайських операцій з банківськими операціями допомагає продавцю онлайн-магазина зберегти свою репутацію в очах клієнтів, банка еквайра та платіжної системи.

ВИСНОВКИ

Під час виконання кваліфікаційної магістерської роботи було зібрано інформацію та зроблено аналіз про види шахрайських операцій з банківськими картками, методи їх виявлення та протидії. Було виявлено та сформульовано основні вимоги до системи, яка повинна автоматизовано захищати учасників онлайн-платежів в Інтернеті.

З метою виконання поставлених вимог та ефективного функціонування системи було обрано відповідні технології та інформаційні рішення, які добре інтегруються у систему електронної комерції.

В ході написання дипломної роботи було обрано архітектуру «клієнт-сервер» та технології, за допомогою яких був побудований прототип автоматизованої системи.

Під час моделювання системи весь процес виявлення шахрайських операцій було розділено на підпроцеси – бізнес-моделі. Їх детальний аналіз для наочності було зображено в нотаціях моделювання IDEF0 та IDEF3. Їх декомпозиція дозволила краще зрозуміти алгоритм виконання системи.

В роботі було відображено бізнес-схеми алгоритмів виконання автоматичного виявлення шахрайських операцій з банківськими картками. Подані блох-схеми детально демонструють логіку, за якою відбувається аналіз онлайн-платежу на можливість шахрайських дій.

Для роботи автоматизованого модулю спроектовано та реалізовано базу даних, яку включає сутності для зберігання та аналізу онлайн-платежів. Дані з бази даних використовуються для роботи співробітником банку з шахрайськими операціями.

З метою наочності було наведено контрольний приклад системи із зображеннями інтерфейсу додатку.

Для відображення доцільності розробленої системи було розраховано очікувану економічну ефективність від її впровадження. Очікуваний річний

економічний ефект складає 34963 грн з коефіцієнтом ефективності 1,092. Очікуваний термін окупності фінансових вкладень складає менше року. Автоматизована система також має соціальний ефект: надійність платежів захищає репутацію інтернет-магазину та банку.

Автоматизований модуль може використовуватися на практиці в електронній комерції для зменшення кількості шахрайських замовлень. Модуль повинен бути інтегрований в інформаційну систему Інтернет-магазину та з'єднуватися з відповідним банком-еквайром.

До рекомендацій стосовно покращення автоматичної системи можна віднести її ускладнення новими функціональними можливостями. Для зменшення шахрайських операцій можна додати ще фільтри, які будуть перевіряти платежі. Проте це може призвести до зменшення конверсії. Тому важливим є налаштуванням системи під окремий вид електронної комерції. Якщо продається товар з низькою націнкою та великою собівартістю, то для погашення його втрати через шахрайство потрібно буде продати велику кількість товару. В цьому випадку необхідно максимально зменшити можливість шахрайських операцій. Якщо навпаки продається товар чи послуга, в ціну якої закладено більше 80% прибутків, то потрібно максимально збільшувати конверсію магазину. Серед можливих засобів фільтрації платежів я рекомендую реалізувати наступні:

– фільтрація за операційної системою та приладом, з якого відбувається платіж;

– фільтрація за сумою платежу (вартість покупки складає більше 90% заощаджень на рахунку);

– фільтрація по випадкам нестачі коштів;

– фільтрація за товарами.

Для удосконалення системи додатково рекомендується створити можливість для співробітника банку формувати звіти, зберігати та імпортувати їх.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ветрова В. Д. Национальная платежная система как инструмент финансовой независимости страны: причины создания, проблемы и перспективы развития, 2014. Изд. 3 (18). С. 17–19.

2. Annual report: True Cost of Fraud. 2016. URL: https://www.lexisnexis.com/risk/downloads/assets/true-cost-fraud-2016.pdf (last accessed: 13.10.2018).

3. Credit Card & Debit Card Fraud Statistics, 2017. URL: https://wallethub.com/edu/credit-debit-card-fraud-statistics/25725/ (last accessed: 12.11.2018).

4. Fraud Trends 2016. Latest perspectives on international eCommerce fraud. 2016. 19 p. URL: http://www.worldpay.com/sites/default/files/Fraud-trends-2016.PDF (last accessed: 13.10.2018).

5. Preventing money laundering and bank fraud in the banking industry. URL: https://www.aciworldwide.com/-/media/files/collateral/case-studies/preventing-money-laundering-and-bank-fraud-in-the-banking-industry-cs-us.pdf (last accessed: 12.10.2018).

6. Актуальні питання діяльності правоохоронних органів у сфері протидії кіберзлочинності : Матеріали Міжнародної науково-практичної конференції (м. Харків, 12 лист. 2014 р.). Харків, 2014. 200 с.

7. Клочко А. М. Злочини у сфері банківської діяльності. Суми, 2014. № 1(10). С. 68-71.

8. Кримінальний кодекс України URL: http://zakon.rada.gov.ua/laws/show/2341-14 (дата звернення: 21.11.2018).

9. Кібербезпека в Україні: правові та організаційні питання: Матеріали Всеукраїнської науково-практичної конференції (21 жовтня 2016, м. Одеса). Одеса, 2016. 235 с.

10. Статистика платіжного шахрайства – підсумки 2017 року URL: https://ema.com.ua/cyberfraud-ema-statistics-results-2017/ (дата звернення: 20.11.2018).

11. Постанова НБУ № 95 «Про затвердження Положення про організацію заходів із забезпечення інформаційної безпеки в банківській системі України» від 28.09.2017. 2017. URL: http://zakon3.rada.gov.ua/laws/show/v009-5500-17 (дата звернення: 05.12.2018).

12. Луговець В. В. Кібернетичні загрози державному сектору економіки. Моделювання та прогнозування економічних процесів: Матеріали X науково-практичної конференції (м. Київ, 7-9 грудня 2016). Київ, 2016. С. 69-72. URL: http://mses.kpi.ua/konfer/37.pdf (дата звернення: 14.11.2018).

13. Baesens B. Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection. 2015. URL: http://www.dataminingapps.com/wp-content/uploads/2015/08/68614_excerpt-1.pdf (last accessed: 22.11.2018).

14. Wrobel-Konior S. Payment Trends in 2017: Fighting Fraud Using Machines. URL: https://www.business2community.com/cybersecurity/ payment-trends-2017-fighting-fraud-using-machines-01690864 (last accessed: 20.09.2018).

15. Ryan C. Hybrid Risk: The truth behind first party fraud. *The official site of the company «Experian».* 2015. URL: http://www.experian.com/blogs/insights/2015/10/hybrid-risk-the-truth-behind-first-party-fraud/ (last accessed: 02.12.2018).

16. Коваленко О. Г., Зубова М. Н. Правовые риски, связанные с пластиковыми картами. *Экономика и управление*. 2013. № 1 (12). С. 56–58.

17. Мельник С. І. Організаційно-правове забезпечення системи економічної безпеки банку, 2011. №1. С. 199–206.

18. Изотов Д.С., Быкова Н.Н. Виды мошенничества с банковскими картами. *Вестник НГИЭИ.* 2015, № 3 (46). С.81-83.

19. Фінагеєв В. О., Чернявський С. С., Татаров О. Ю. Виявлення та розслідування злочинів, пов’язаних з використанням засобів доступу до банківських рахунків. Київ, 2013. 79 с.

20. Яровенко Г. М. Розробка інформаційної моделі виявлення ознак шахрайств у банках. *Економічна наука.* 2018.№ 14. С. 23-28. URL: http://www.investplan.com.ua/pdf/14_2018/7.pdf (дата звернення: 23.11.2018).

21. Яровенко Г. М., Ковач В.О. Моделювання портретів потенційних шахрая та жертви банківських шахрайств. *Ефективна економіка*. 2018. URL: http://www.economy.nayka.com.ua/pdf/10_2018/63.pdf (дата звернення: 25.11.2018).

22. Яровенко Г. М., Сковронська А. І., Бояджян М. М. Моделювання виявлення ознак кіберзагроз в банках із використанням інтелектуального аналізу. *Ефективна економіка*. 2018. URL: http://www.economy.nayka.com.ua/pdf/7_2018/39.pdf (дата звернення: 23.11.2018).

23. Card-Not-Present Fraud in a Post-EMV Environment: Combating the Fraud Spike. 2014. URL: https://www.emc.com/collateral/white-papers/card-not-present-fraud-post-emv-env-wp.pdf (last accessed: 14.11.2018).

24. Проблеми кібербезпеки інформаційнотелекомунікаційних систем: Збірник матеріалів доповодідей та тез (05-06 квітня 2018, м. Київ). Київ, 2018. 506с.

25. Глебов О.А. Актуальные методы противодействия мошенничеству. *Банковское дело*. 2014, №9. С.74-76.

26. Клімов С. В. Система виявлення шахрайських операцій з банківськими картками. *Проблеми та перспективи розвитку фінансово-кредитної системи України* : збірник матеріалів ІІІ Всеукраїнської науково-практичної on-line конференції (м. Суми, 22–23 листопада 2018 р). Суми, 2018. С. 303-307.

27. Fraud Detection in Debit Card Transactions, 2015. URL: https://www.researchgate.net/publication/301908348_Fraud_Detection_in_Debit_Card_Transactions (last accessed: 21.11.2018).

28. Prasad, V. Method and System for Detecting Fraud in Credit Card Transaction. *International Journal of Innovative Research in Computer and Communication Engineering.* 2013.

29. Фролов Д. Б., Ревенков П. В. кибербезопасность в условиях применения систем электронного банкинга. *Деньги и кредит*. 2016, № 6. С. 9-12.

30. Dheepa V., Dhanapal R. Analysis of Credit Card Fraud Detection Methods. *International Journal of Recent Trends in Engineering*. 2009. pp. 126-128.

31. Rana P. and Baria J. A Survey on Fraud Detection Techniques in Ecommerce. *International Journal of Computer Applications.* 2015. Pp. 5-7.

32. Batani J. An Adaptive and Real-Time Fraud Detection Algorithm in Online Transactions. *International Journal of Computer Science and Business Informatics*. 2017. 12 p.

33. Rama K., Uma D. Fraud Detection of Credit Card Payment System by Genetic Algorithm. International Journal of Scientific & Engineering Research, issue 3, 2012. pp. 1-6.

34. Usman A., Shah M. Critical Success Factors for Preventing eBanking. *Journal of Internet Banking and Commerce*. 2013. Issue 18(2). Pp. 1 –13.

35. FraudLabs Pro: protect your business from online fraud URL: https://www.fraudlabspro.com/features (last accessed: 14.09.2018).

36. Oculeus Anti-Fraud System URL: https://www.oculeus.com/ systemantifraud.html (last accessed: 16.09.2018).

37. Alaris Anti Fraud System. URL: http://alarislabs.com/solutions/anti-fraud (last accessed: 16.09.2018).

38. Fraud Detection Solution for eCommerce. URL : https://www.riskified. com/solution/fraud-detection (last accessed: 16.09.2018).

39. WayForPay Antifraud. URL: https://wayforpay.com/uk/antifraud. (last accessed: 16.09.2018).

40. Анісімов В. В. Проектування інформаційних систем: конспект лекцій. Хабаровск, 2009. URL: https://sites.google.com/site/anisimovkhv /publication/umr/pris (дата звернення: 18.12.2018).

41. Поморова О.В., Говорущенко Т.О. Проектування інтерфейсів користувача: Навчальний посібник. Хмельницький, 2011. 206 с.

42. Гадецька З. М., Холопова М. О. Моделювання бізнес-процесів діяльності підприємства. *Ефективна економіка*. 2016. №5. URL: http://www.economy.nayka.com.ua/?op=1&z=4950.

43. Заїка А. В., Філенко М. І., Остапченко А. С. Моделювання архітектурних рішень підтримки мультисайтовості для організації інформаційних систем. *Вісник КрНУ імені Михайла Остроградського*. 2015. №3. С. 54–59. URL: http://www.kdu.edu.ua/PUBL/statti/2015_3_54-3-2015.pdf (дата звернення: 20.11.2018).

44. AllFusion Process Modeler: Getting Started. URL: https://support content.ca.com/cadocs/0/e002711e.pdf (last accessed: 20.11.2018).

45. Business Process Model and Notation (BPMN) version 2.0. *The official site of the company «Object Management Group»*. 2011. URL: http://www.omg.org/spec/BPMN/2.0 (last accessed: 09.12.2018).

46. Скотт Б. Проектування веб-інтерфейсів. СПб, 2010. 352 с.

47. Кнут Д. Теоретичні основи технології програмування URL: http://www.bourabai.kz/alg/technology.htm (дата звернення: 01.12.2018).

48. Фаулер М. Основи UML. 3-тє вид. Санкт-Петербург, 2004. 192 с. URL: https://studfiles.net/preview/6354103/ (дата звернення: 11.12.2018).

49. Вахнюк, С.В. Технологія створення програмних та інтелектуальних систем : навчальний посібник. Суми, 2011. 254 с

50. Пасічник О. Г., Пасічник О. В., Стеценко І. В. Основи веб-дизайну: Навчальний посібник. К, 2009. 336 с. URL: http://school1k24.at.ua/ 10CLASS_WEB/OsnovyWebDis.pdf (дата звернення: 18.12.2018).

51. Ривкінд Й. Я., Лисенко Т. І., Чернікова Л. А. Інформатика. 11 клас. Академічний рівень, профільний рівень : Підручн. Харків, 2011. 304 с.

52. Колісніченко Д. PHP і MySQL. Розробка Веб-додатків. Санкт-Петербург, 2015. 593 с..

53. Гебель, Є. С. Програмування та основи алгоротмізації.: конспект лекцій. Омськ, 2012. 111 с. URL: https://studfiles.net/preview/2702812/ (дата звернення: 18.12.2018).

54. Яременко Н.С. Технологія створення інформаційних систем : навчальні презентації. Суми, 2016.

55. Шмітт К. CSS. Рецепти програмування. 2-ге вид. СПб, 2009. 592 с.

56. Ніксон Р. Створюємо динамічні веб-сайти за допомогою PHP, MySQL, JavaScript, CSS та HTML5. *Пітер*. Санкт-Петербург, 2016. 767 с.

57. Faulkner S., Eicholz A., Leithead T., Danilo A., Moon S. HTML 5.2 W3C Recommendation, 2017. URL: https://www.w3.org/TR/html/ (last accessed: 09.11.2018).

58. Довідник CSS. URL: https://webref.ru/css/ (дата звернення: 02.12.2018).

59. Довідник JavaScript. URL: https://javascript.ru/manual (дата звернення: 05.12.2018).

60. Дронов В. А. HTML 5, CSS 3 и Web 2.0. Розробка сучасних Веб-сайтів. Санкт-Петербург, 2011. 416 с.

61. Лабберс П., Олберс Б., Салім Ф. HTML5 для професіоналів: потужний інструмент для розробки сучасних веб-додатків.Москва, 2011.272с.

62. Уроки з HTML та CSS. URL: https://webref.ru/layout/learn-html-css/ (дата звернення: 02.12.2018).

63. PHP 5 Tutorial. URL: https://www.w3schools.com/php/ (last accessed: 16.11.2018).

64. Дарі К., Брінзаре Б., Черчез-Тоза Ф., Бусіка М. AJAX и PHP. Розробка динамічних веб-додатків. Санкт-Петербург, 2009. 336 с.

65. PHP lernen leicht gemacht. URL: https://www.php-einfach.de/ (last: accessed 14.11.2018).

66. PHP Manual. URL : https://secure.php.net/manual/en/intro-whatcando. php (last accessed: 14.11.2018).

67. PHP: Hypertext Preprocessor. URL: http://php.net/manual/en/ langref.php (last accessed: 06.11.2018).

68. Date C. An Introduction to Database Systems, 8th edition. 2012. 247 p. URL: https://www.pdfdrive.com/an-introduction-to-database-systems-8e-by-c-j-datepdf-e31093920.html (last accessed: 13.10.2018).

69. MySQL Documentation. Oracle Corporation and/or its affiliates. URL: https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html (last accessed: 06.11.2018).

70. Сучасний підручник JavaScript. URL: http://learn.javascript.ru/ (дата звернення: 05.12.2018).

71. Початок роботи з Bootstrap. URL: http://mybootstrap.ru/get-started (дата звернення: 05.12.2018).

72. Bootstrap Introduction. URL: https://getbootstrap.com/docs/4.1/getting-started/introduction/ (last accessed: 12.12.2018).

73. GRUBER M. Understanding SQL. Moskow, 1993. 291 p. URL: http://specfx.narod.ru/books/SQL_M_Gruber.pdf (last accessed: 02.09.2018).

74. Теорія нормальних форм. URL: http://www.mstu.edu.ru/education/materials/zelenkov/ch_4_2.html (дата звернення: 06.09.2018).

75. Заробітна плата розробників URL: https://jobs.dou.ua/salaries/ (дата звернення: 25.11.2018).

ДОДАТКИ

Додаток А

(Обов’язковий)

SUMMARY

Klimov S. V. Prototype development of the automated module for detecting fraud transactions with bank cards. – Masterslevel Qualification Thesis. Sumy State University, Sumy, 2018.

The master’s thesis focuses on the detecting fraud transactions with bank cards. The analysis of the existing automated systems for detecting fraud transactions and the description of the main business processes, information and algorithmic support are carried out. The prototype of the module has been implemented. The expected economic effect of the implementation is calculated.

Keywords: automation, automated system, fraud, fraud detection, fraudulent operation, transactions, bank card, e-commerce, anti-fraud.

АНОТАЦІЯ

Клімов С. В. Розробка прототипу автоматизованого модулю процесу виявлення шахрайських операцій з банківськими картками. – Кваліфікаційна магістерська робота. Сумський державний університет, Суми, 2018 р.

У роботі досліджено процес виявлення шахрайських операцій з банківськими картками, проаналізовано існуючі автоматизовані системи виявлення шахрайських операцій. Здійснено опис основних бізнес-процесів системи, інформаційного та алгоритмічного забезпечення системи. Реалізовано прототип модулю та розраховано очікуваний економічний ефект від впровадження.

Ключові слова: автоматизація, автоматизована система, шахрайство, виявлення шахрайства, шахрайська операція, банківська операція, банківська картка, електронна комерція, антифрод.

Додаток Б

(довідковий)

ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Лістинг Б.1 – Створення бази даних, ключових таблиць та зв’язків між ними

```
CREATE DATABASE ip2;
CREATE TABLE `cards` (
  `cardID` varchar(16) COLLATE utf8_bin NOT NULL,
  `clientID` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
CREATE TABLE `clients` (
  `clientID` int(11) NOT NULL,
  `fname` varchar(100) COLLATE utf8_bin NOT NULL,
  `sname` varchar(100) COLLATE utf8_bin NOT NULL,
  `patronymic` varchar(100) COLLATE utf8_bin NOT NULL,
  `telephone` varchar(12) COLLATE utf8_bin NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
CREATE TABLE `frauds` (
  `fraudID` int(11) NOT NULL,
  `transactionID` int(11) NOT NULL,
  `code` int(11) NOT NULL,
  `reason` varchar(128) COLLATE utf8_bin NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
CREATE TABLE `location` (
  `ip_from` int(10) UNSIGNED DEFAULT NULL,
  `ip_to` int(10) UNSIGNED DEFAULT NULL,
  `country_code` char(2) COLLATE utf8_bin DEFAULT NULL,
  `country_name` varchar(64) COLLATE utf8_bin DEFAULT NULL,
  `region_name` varchar(128) COLLATE utf8_bin DEFAULT NULL,
  `city_name` varchar(128) COLLATE utf8_bin DEFAULT NULL,
  `latitude` double DEFAULT NULL,
  `longitude` double DEFAULT NULL,
  `zip_code` varchar(30) COLLATE utf8_bin DEFAULT NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
CREATE TABLE `location_ua` (
  `ip_from` int(10) UNSIGNED DEFAULT NULL,
  `ip_to` int(10) UNSIGNED DEFAULT NULL,
  `region_name` varchar(128) COLLATE utf8_bin DEFAULT NULL,
  `city_name` varchar(128) COLLATE utf8_bin DEFAULT NULL,
  `latitude` double DEFAULT NULL,
  `longitude` double DEFAULT NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
CREATE TABLE `transactions` (
  `transactionID` int(11) NOT NULL,
  `cardID` varchar(16) COLLATE utf8_bin NOT NULL,
  `time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `region` varchar(128) COLLATE utf8_bin NOT NULL,
  `ort` varchar(128) COLLATE utf8_bin NOT NULL,
  `ip` int(10) UNSIGNED NOT NULL,
```

```
  `fraud` tinyint(1) NOT NULL DEFAULT '0'
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
ALTER TABLE `cards`
  ADD PRIMARY KEY (`cardID`),
  ADD KEY `userID` (`clientID`);
ALTER TABLE `clients`
  ADD PRIMARY KEY (`clientID`);
ALTER TABLE `frauds`
  ADD PRIMARY KEY (`fraudID`),
  ADD KEY `transactionID` (`transactionID`);
ALTER TABLE `location`
  ADD KEY `idx_ip_from` (`ip_from`),
  ADD KEY `idx_ip_to` (`ip_to`),
  ADD KEY `idx_ip_from_to` (`ip_from`,`ip_to`);
ALTER TABLE `location_ua`
  ADD KEY `idx_ip_from` (`ip_from`),
  ADD KEY `idx_ip_to` (`ip_to`),
  ADD KEY `idx_ip_from_to` (`ip_from`,`ip_to`);
ALTER TABLE `transactions`
  ADD PRIMARY KEY (`transactionID`),
  ADD KEY `cardID` (`cardID`);
ALTER TABLE `clients`
  MODIFY `clientID` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=3;
ALTER TABLE `frauds`
  MODIFY `fraudID` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=7;
ALTER TABLE `transactions`
  MODIFY `transactionID` int(11) NOT NULL AUTO_INCREMENT,
AUTO_INCREMENT=11;
ALTER TABLE `cards`
  ADD CONSTRAINT `cards_cl ` FOREIGN KEY (`clientID`) REFERENCES
`clients` (`clientID`) ON DELETE CASCADE ON UPDATE CASCADE;
ALTER TABLE `frauds`
  ADD CONSTRAINT `frauds_ibfk_1` FOREIGN KEY (`transactionID`)
REFERENCES `transactions` (`transactionID`) ON DELETE CASCADE ON
UPDATE CASCADE;
ALTER TABLE `transactions`
  ADD CONSTRAINT `trans_card` FOREIGN KEY (`cardID`) REFERENCES
`cards` (`cardID`) ON DELETE CASCADE ON UPDATE CASCADE;
```

Лістинг Б.2 – Підключення бази даних

```
<?php
define('DB_NAME', 'ip2');
define('DB_USER', 'root');
define('DB_PASSWORD', '123qsc');
define('DB_HOST', 'localhost');
define('DB_CHARSET', 'utf8');
$link = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME)
or die('ошибка подключения БД');
mysqli_set_charset($link, "utf8");
?>
```

Додаток В

(довідковий)

АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Лістинг В.1 – Програмний код фільтрування операції з банківською картою

```
$bregion = $_POST['region_name'];
$bcity = $_POST['city_name'];
$fraudrisk = 0; $fraud = [];
$cardID = str_replace(" ","",$_POST['cardID']);
$sql = "SELECT `ort` FROM transactions WHERE cardID = '"
.$cardID . "' AND fraud=0";
$arr = mysqli_query($link,$sql);
while ($result = mysqli_fetch_assoc($arr)) {
    $array[] = $result['ort'];
}
/* фильтр 1 местоположение по IP и адрес доставки*/
if(!empty($array)) {
    if ($region == $bregion) {
        if($city != $bcity) {
            if (!in_array($bcity, $array)) {
                $fraud[] = "новый город доставки";
                $fraudrisk++;
            }
        }
    } else {
        if (!in_array($bcity, $array)) {
                $fraud[] = "разные регионы";
                $fraudrisk++;
        }
    }
}
/* фильтр 2 время и расстояние между заказами*/
if(!empty($array)) {
    $sql = "SELECT `time`, `ort`, `ip`, `latitude`, `longitude`,
`city_name` FROM transactions, location WHERE cardID = '" .
$cardID . "' AND `ip` <= ip_to ORDER BY `time` DESC LIMIT 1";
    $arr = mysqli_query($link,$sql);
    $res = mysqli_fetch_assoc($arr);
    if($res['city_name'] != $city) { // город текущий и город
последней транзакции
        $time_dif = (strtotime("now")-
strtotime($res['time']))/3600; // разница в часах
        $sql = "SELECT `latitude`, `longitude` FROM location
WHERE city_name = '" . $city . "'";
        $result = mysqli_fetch_assoc(mysqli_query($link,$sql));
// координаты текущего города
        $lat1 = $res['latitude'];
        $long1 = $res['longitude'];
        $lat2 = $result['latitude'];
```

```
        $long2 = $result['longitude'];
        $dist = calculateTheDistance($lat1, $long1, $lat2,
$long2) / 1000; // расстояние в км
        if($time_dif < $dist/50) { // скорость 50 км/ч
            $fraudrisk=1;
            $fraud[] = "ошибка во времени";
        }
    }
}
/* фильтр 3 несколько карт по 1 IP*/
if(!empty($array)) {
    $sql = "SELECT DISTINCT cardID as `Cards` FROM
`transactions` WHERE `time` > DATE_SUB(NOW(), INTERVAL 1 DAY)
AND `ip` = '" . $ipnum . "'";
    $result = mysqli_query($link,$sql);
    $cards = [];
    foreach ($result as $res) {
        $cards[] = $res['Cards'];
    }
    $cards[] = $cardID;
    $cards = count(array_unique($cards));
    if ($cards > 2) {
        $fraudrisk=2;
        $fraud[] = "много карт по 1 IP";
    }
}
```

Лістинг В.2 – Розрахунок відстані між містами

```
define('EARTH_RADIUS', 6372795);
function calculateTheDistance ($φA, $λA, $φB, $λB) {
    // перевести координаты в радианы
    $lat1 = $φA * M_PI / 180;
    $lat2 = $φB * M_PI / 180;
    $long1 = $λA * M_PI / 180;
    $long2 = $λB * M_PI / 180;
    // косинусы и синусы широт и разницы долгот
    $cl1 = cos($lat1);
    $cl2 = cos($lat2);
    $sl1 = sin($lat1);
    $sl2 = sin($lat2);
    $delta = $long2 - $long1;
    $cdelta = cos($delta);
    $sdelta = sin($delta);
    // вычисления длины большого круга
    $y = sqrt(pow($cl2 * $sdelta, 2) + pow($cl1 * $sl2 - $sl1 *
$cl2 * $cdelta, 2));
    $x = $sl1 * $sl2 + $cl1 * $cl2 * $cdelta;
    $ad = atan2($y, $x);
    $dist = $ad * EARTH_RADIUS;
    return $dist;
}
```

Лістинг В.3 – Збереження результатів в базі даних

```
if ($fraudrisk > 0) {
    $sql = "SELECT tr.cardID, tr.ip FROM frauds f, transactions
tr WHERE f.transactionID = tr.transactionID AND tr.fraud = 0
            AND tr.time > DATE_SUB(NOW(), INTERVAL 3 HOUR)";
    $frauds = mysqli_query($link, $sql);
    while ($result = mysqli_fetch_assoc($frauds)) {
        $WL_cards[] = $result['cardID'];
        $WL_ip[] = $result['ip'];
    }
    if(in_array($cardID, $WL_cards) || in_array($ipnum, $WL_ip))
    {
        $sql = 'INSERT INTO `transactions`(`cardID`, `region`,
`ort`, `ip`, `fraud`) VALUES ("' . $cardID . '", "' . $bregion .
'", "' .
        $bcity . '", "' . $ipnum . '", "0")';
        echo $sql;
        mysqli_query($link, $sql);
    } else {
        $fraud = implode(" ",$fraud);
        $sql = 'INSERT INTO `transactions`(`cardID`, `region`,
`ort`, `ip`, `fraud`) VALUES ("' . $cardID . '", "' . $bregion .
'", "' .
        $bcity . '", "' . $ipnum . '", "1")';
        mysqli_query($link, $sql);
        $transactionID = mysqli_insert_id($link);
        $code = mt_rand(10000, 99999);
        $sql = "INSERT INTO `frauds`(`transactionID`, `code`,
`reason`) VALUES ('" . $transactionID . "', '" . $code . "', '"
.$fraud . "')";
        mysqli_query($link, $sql);
        include_once('send.php');
        mysqli_close($link);
        echo '<form method="post" accept-charset="UTF-
8"action="send_form.php" name="send_form">
                <input type="hidden" name="transaction" value="'
. $transactionID . '">
            </form>';
        echo '<script type="text/javascript">
        document.forms["send_form"].submit();
        </script>';
    }
}
```

Додаток Г

(довідковий)

КЛІЄНТСЬКИЙ ВЕБ-ДОДАТОК

Лістинг Г.1 – Створення вікна здійснення онлайн-платежу

```
<?php include_once('ip.php'); ?>
<!DOCTYPE html>
<html lang="en">
<head>
     <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
     <title>Anti-fraud system</title>
     <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/boo
tstrap.min.css" integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO
" crossorigin="anonymous">
     <link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
     <div class="container">
          <h1>Страница осуществления онлайн-транзакции</h1>
               <form method="post" accept-charset="UTF-8"
name="myform" action="analys.php" id="form">
                    <div class="row">
                         <div class="form-group col-lg-4 col-md-6
col-sm-8 col-12">
                              <label class="form-row">
                                   <span class="folm_label">Номер
карты</span>
                                  <input type="text" class="form-
control form__input" name="cardID" pattern="[0-9]{4}\s[0-
9]{4}\s[0-9]{4}\s[0-9]{4}"  required placeholder="XXXX XXXX XXXX
XXXX">
                              </label>
                        </div>
                        <div class="form-group col-lg-3 col-md-4
col-sm-6 col-8">
               <label class="form-row">
               <span class="folm_label">Срок действия</span>
                    <div class="grid grid-gutter">
                         <div class="item-gutter">
                              <span class="form__input
form_input-selectable">
                                   <select id="MM" name="MM"
class="form-select">
                                   <option value="01">01</option>
                                   <option value="02">02</option>
```

```
                                        <option value="03">03</option>
                                        <option value="04">04</option>
                                        <option value="05">05</option>
                                        <option value="06">06</option>
                                        <option value="07">07</option>
                                        <option value="08">08</option>
                                        <option value="09">09</option>
                                        <option value="10">10</option>
                                        <option value="11">11</option>
                                        <option value="12">12</option>
                                </select>
                                </span>
                        </div>
                        <div class="item-gutter">
                        <span class="form__input form_input-selectable">
                        <select id="YY" name="YY" class="form-select">
                                        <option value="18">18</option>
                                        <option value="19">19</option>
                                        <option value="20">20</option>
                                        <option value="21">21</option>
                                        <option value="22">22</option>
                                        <option value="23">23</option>
                                        <option value="24">24</option>
                                        <option value="25">25</option>
                        </select>
                        </span>
                        </div>
                </div>
                </label>
        </div>
        <div class="form-group col-md-2 col-sm-3 col-4">
                <label class="form-row">
                        <span class="folm_label">Код CVV2</span>
                       <input type="password" class="form-control
form__input" name="CVV2" maxlength="3" required
placeholder="XXX">
                </label>
         </div>
                        </div>
                        <div class="row"><b>Адрес доставки</b></div>
                        <div class="row">
                        <div class="form-group col-lg-3 col-sm-6 col-12">
                             <label class="form-row">
                                  <span class="folm_label">Область</span>
                                      <span class="form__input form_input-
selectable">
                                       <select id="region_name"
name="region_name" class="form-select">
                                        <?php
                                        $sql = "SELECT DISTINCT
region_name FROM location_ua WHERE `region_name` <> '-'";
                                       $array = mysqli_query($link, $sql);
```

```
                                    while ($result =
mysqli_fetch_assoc($array)) {
     if($result['region_name']== $region){
                                        echo '<option value="' .
$result['region_name'] . '" selected>' . $result['region_name']
. '</option>';
                                            } else {
               echo '<option value="' . $result['region_name'] .
'">' . $result['region_name'] . '</option>';
                                                }
                                            }
                                            ?>
                                        </select>
                                    </span>
                                </label>
                            </div>
                <div class="form-group col-lg-3 col-sm-6 col-12">
                   <label class="form-row">
                   <span class="folm_label">Город доставки</span>
                  <span class="form__input form_input-selectable" >
                   <select id="city_name" name="city_name"
class="form-select">
                        <?php $sql = "SELECT DISTINCT city_name FROM
location_ua WHERE `region_name` = '" . $region . "'";
                        $array = mysqli_query($link, $sql);
                        while ($result = mysqli_fetch_assoc($array))
{
     if($result['city_name']== $city){
                        echo '<option value="' . $result['city_name']
. '" selected>' . $result['city_name'] . '</option>';
                                        } else {
                        echo '<option value="' . $result['city_name']
. '">' . $result['city_name'] . '</option>';
                                    }
                                }
                                    mysqli_close($link);?>
                                    </select>
                                </span>
                            </label>
                        </div>
                <div class="form-group col-lg-3 col-sm-6 col-12">
                        <label class="form-row">
                        <span class="folm_label">Улица</span>
                       <input type="text" class="form-control
form__input" name="street" required >
                        </label>  </div>
                  <div class="form-group col-lg-1 col-sm-3 col-6">
                        <label class="form-row">
                             <span class="folm_label">Дом</span>
                           <input type="text" class="form-control
form__input" name="street" required >
                        </label>
                          </div>
```

```
                        <div class="form-group col-lg-1 col-6">
                                <label class="form-row">
                            <span class="folm_label">Квартира</span>
                    <input type="text" class="form-control
form__input" name="street" >
                                </label>
                        </div>
                </div>
                <div class="center">
                <input type="submit" name="form" class="button"
value="Оплатить">
                </div>
            </form>
     </div>
<script type="text/javascript" src="js/jquery-
3.3.1.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/
1.14.3/umd/popper.min.js" integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49
" crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3
/js/bootstrap.min.js" integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy
" crossorigin="anonymous"></script>
<script type="text/javascript">
      $(document).ready(function(){
        $("#region_name").change(function() {
            var region = {region:$("#region_name").val()};
            $.ajax({
                type:'POST',
                url:'ajax.php',
                data:region,
                success:function(data){
                    $('#city_name').html(data)
                }
            });
        });
        var field = $('#region_name').find('option');
    });
     var cc = myform.cardID;
     for (var i in ['input', 'change', 'blur', 'keyup']) {
     cc.addEventListener('input', formatCardCode, false);
     }
     function formatCardCode() {
         var cardCode = this.value.replace(/[^\d]/g,
'').substring(0,16);
         cardCode = cardCode != '' ?
cardCode.match(/.{1,4}/g).join(' ') : '';
         this.value = cardCode;
     }
</script>
</body>
</html>
```

Лістинг Г.2 – Створення вікна підтвердження платежу

```
<?php
include_once('db.php');
if(!empty($_POST['code'])) {
     $sql = "SELECT code FROM `frauds` WHERE transactionID = " .
$_POST['transaction'] ;
    mysqli_query($link, $sql);
    $result = mysqli_fetch_assoc(mysqli_query($link,$sql));
    if($result['code'] == $_POST['code']) {
     $sql = "UPDATE transactions SET fraud=0 WHERE transactionID
= " . $_POST['transaction'];
     mysqli_query($link, $sql);
     header("Location: /");
    } else {
     $error = "Вы ввели неверный код попробуйте снова";
    }
}
mysqli_close($link);
?>
<!DOCTYPE html>
<html lang="en">
<head>
     <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
     <title>Anti-fraud system</title>
     <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/boo
tstrap.min.css" integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO
" crossorigin="anonymous">
     <link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
     <div class="container">
          <h1>Подтверждение онлайн-транзакции</h1>
          <?php if (!empty($error)) {
               echo "<h2 class='error'>" . $error . "</h2>";
          } else {
               echo '<h2>на ваш телефон было отправлено СМС-
сообщение с кодом подтверждения<br> введите этот код в поле и
нажмите подтвердить</h2>';
          }
          ?>
          <form method="post" accept-charset="UTF-8"
name="myform" action="send_form.php" id="form">
               <div class="row">
          <div class="form-group col-lg-4 col-sm-3 col-12"></div>
               <div class="form-group col-lg-4 col-sm-6 col-12">
                         <label class="form-row">
                              <span class="folm_label">Код
подтверждения</span>
```

```
                                <input type="text" class="form-
control form__input" name="code" maxlength="5" required >
                                <input type="hidden"
name="transaction" value="<?= $_POST[‘transaction’]; ?>">
                            </label>
                </div>
                <div class="form-group col-lg-4 col-sm-3 col-12
center"></div>
            </div>
            <div class="center">
              <input type="submit" name="form" class="button"
value="Подтвердить">
          </div>
        </form>
    </div>
</body>
</html>
```

Лістинг Г.3 – Створення вікна виведення шахрайських операцій

```
<?php include_once(‘db.php’);
$sql = "SELECT Tr.`transactionID`, Concat(Cl.sname, ‘ ‘,
Cl.fname) as ‘Client’, Tr.`cardID`, Cl.telephone, Tr.`time`,
Tr.`fraud`, Fr.`reason`
FROM `frauds` Fr
INNER JOIN `transactions` Tr ON Tr.`transactionID` =
Fr.`transactionID`
INNER JOIN `cards` C ON C.cardID = Tr.cardID
INNER JOIN `clients` Cl ON Cl.clientID = C.clientID";
$array = mysqli_query($link,$sql);
 ?>
<!DOCTYPE html>
<html lang="en">
<head>
     <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
     <title>Anti-fraud system</title>
     <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/boo
tstrap.min.css" integrity="sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO
" crossorigin="anonymous">
</head>
<body>
     <div class="container">
          <h2 style="text-align: center;">Результат работы
модуля</h2>
          <h2 style="text-align: center;">операции, которые
вызывают подозрения</h2>
          <table class="table table-striped">
               <thead>
```

```
                    <tr>
                      <th scope="col">#</th>
                      <th scope="col">Клиент</th>
                      <th scope="col">Карта</th>
                      <th scope="col">Дата и время</th>
                      <th scope="col">Причина отмены платежа</th>
                      <th scope="col">Операция мошенническая</th>
                      <th scope="col">Телефон</th>
                    </tr>

                <tr>
                     <td></td>
                     <td>
                       <input id="client" class="form-control">
                    </td>
                    <td>
                        <input id="card" class="form-control">
                    </td>
                    <td>
                      <input type="date" name="date" id="time"
class="form-control">
                    </td>
                    <td>
                      <select id="reason" class="form-control">
                           <option value="">---</option>
                           <option value="новый город
доставки">новый город доставки</option>
                           <option value="ошибка во времени">ошибка
во времени</option>
                           <option value="разные регионы">разные
регионы</option>
                           <option value="много карт по 1 IP">много
карт по 1 IP</option>
                      </select>
                    </td>
                    <td>
                      <select id="fraud" class="form-control">
                           <option value="">--</option>
                           <option value="Нет">Нет</option>
                           <option value="Да">Да</option>
                      </select>
                    </td>

                    <td>
                      <input id="telephone" class="form-control">
                    </td>
                </tr>
                </thead>
                <tbody id="target">
               <?php $i=0; while ($result =
mysqli_fetch_assoc($array)) { $i++;
                echo "<tr>
                 <td>" . $i . "</td>
```

```
                <td>" . $result['Client'] . "</td>
                <td>" . $result['cardID'] . "</td>
                <td class='edit time " .$result['transactionID']
."'>" . substr($result['time'], 0, 10) . "</td>
                <td>" . $result['reason'] . "</td>
                <td class='edit fraud " .$result['transactionID']
."'>";
             echo ($result['fraud']==1) ? "Да" : "Нет";
             echo "</td>
               <td class='edit telephone "
.$result['transactionID'] ."'>" . $result['telephone'] . "</td>
              </tr>";
              } ?>
              </tbody>
          </table>
     </div>
<script type="text/javascript" src="js/jquery-
3.3.1.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd
/popper.min.js" integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49
" crossorigin="anonymous"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/boots
trap.min.js" integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy
" crossorigin="anonymous"></script>
<script type="text/javascript"
src="js/filterTable.v1.0.src.js"></script>
<script type="text/javascript" src="js/bank.js"></script>
</body>
</html>
```

Лістинг Г.4 – Програмний код фільтрації інформації у веб-додатку

```
var filterTable = function (HTMLTBodyRef, aFilters) {
    var rows = HTMLTBodyRef.getElementsByTagName("TR"),
        filters = {}, n,
        walkThrough = function (rows) {
            var tr, i, f;
            for (i = 0; i < rows.length; i += 1) {
                tr = rows.item(i);
                for(f in filters) {
                    if (filters.hasOwnProperty(f)) {
                        if (false ===
filters[f].validate(tr.children[f].innerText) ) {
                            tr.style.display = "none"; break;
                        } else {
                            tr.style.display = "";
                        }
                    }
```

```
                }
            }
        };
    for(n in aFilters) {
        if (aFilters.hasOwnProperty(n)) {
            if (aFilters[n] instanceof filterTable.Filter) {
                filters[n] = aFilters[n];
            } else {
                filters[n] = new
filterTable.Filter(aFilters[n]);
            }
            filters[n]._setAction("onchange", function ()
{walkThrough(rows);});
        }
    }
}
```

```
filterTable.Filter = function (HTMLElementRef, callback, eventName) {
  /* Если ф-цию вызвали не как конструктор фиксим этот момент: */
  if (!(this instanceof arguments.callee)) {
    return new arguments.callee(HTMLElementRef, callback, eventName);
  }
```

```
    /* Выравниваем пришедший аргумент к массиву */
    this.filters = {}.toString.call(HTMLElementRef) == "[object
Array]" ? HTMLElementRef : [HTMLElementRef];

    /**
     * Шаблонный метод вызывается для каждой строки таблицы, для
соответствующей
     * ячейки. Номер ячейки задается в объекте-конфигурации
фильтров ф-ции
     * filterTable (См. параметр 2 ф-ции tableFilter )
     * @param String cellValue - строковое значение ячейки
     * @returns {boolean}
     */
    this.validate = function (cellValue) {
        for (var i = 0; i < this.filters.length; i += 1) {
            if ( false === this.__validate(cellValue,
this.filters[i], i) ) {
                return false;
            }
        }
    }
    this.__validate = function (cellValue, filter, i) {
        /* Если фильтр был создан явно и явно указана функция
валидации: */
        if (typeof callback !== "undefined") {
            return callback(cellValue, this.filters, i);
        }
        /* Если в фильтр напихали пробелов или другой непечатной
фигни - удаляем: */
        filter.value = filter.value.replace(/^\s+$/g, "");
```

```
        /* "Фильтр содержит значение и оно совпало со значением
ячейки" */
        return !filter.value || filter.value == cellValue;
    }
    this._setAction = function (anEventName, callback) {
        for (var i = 0; i < this.filters.length; i += 1) {
            this.filters[i][eventName||anEventName] = callback;
        }
    }
};
```

Лістинг Г.5 – Маніпулювання даними про шахрайські платежі у реальному часі

```
$(document).ready(function(){
        $("#region_name").change(function(){
            var region = {region:$("#region_name").val()};
            $.ajax({
                type:'POST',
                url:'ajax.php',
                data:region,
                success:function(data){
                    $('#city_name').html(data)
                }
            });
        });
        var field = $('#region_name').find('option');
        filterTable( document.getElementById("target"), {
   1: new filterTable.Filter(document.getElementById("client"),
                function (value, filters, i) {
                   return value.indexOf(filters[i].value) === 0;
                },
                "onkeyup"
            ),
   2: new filterTable.Filter(document.getElementById("card"),
                function (value, filters, i) {
                   return value.indexOf(filters[i].value) === 0;
                },
                "onkeyup"
            ),
   3: document.getElementById("time"),
   4: document.getElementById("reason"),
   5: document.getElementById("fraud"),
   6:       new       filterTable.Filter(document.getElementById
("telephone"),
                function (value, filters, i) {
                   return value.indexOf(filters[i].value) === 0;
                },
                "onkeyup"
            )
        });
        $('td.edit').click(function(){
          $('.ajax').html($('.ajax input').val());
```

```
          $('.ajax').removeClass('ajax');
               $(this).addClass('ajax');
               $(this).html('<input     id="editbox"     size="'+
$(this).text().length+'" type="text" value="' + $(this).text() +
'" />');
               $('#editbox').focus();
        });
        $('td.edit').keydown(function(event){
          arr = $(this).attr('class').split( " " );
          console.log(arr);
          if(event.which == 13) {
                    $.ajax({
                         type: "POST",
                         url:"ajax.php",
                         data:                  "value="+$('.ajax
input').val()+"&id="+arr[2]+"&field="+arr[1],
                         success: function(data){

                              $('.ajax').html($('.ajax
input').val());
                              $('.ajax').removeClass('ajax');
                         }
                    });
               }
          });
          $(document).on('blur', '#editbox', function(){
               $('.ajax').html($('.ajax input').val());
               $('.ajax').removeClass('ajax');
          });
    });
```