DEVELOPMENT OF AN AI-POWERED PAYMENT GATEWAY SYSTEM WITH FRAUD DETECTION AND AUTOMATED RECEIPT GENERATION

BY

EZE CHARITY MMESOMA
GOU/U22/CSC/786

DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS
FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY
GODFREY OKOYE UNIVERSITY, ENUGU STATE.

JUNE, 2026.

TITLE PAGE

DEVELOPMENT OF AN AI-POWERED PAYMENT GATEWAY SYSTEM WITH FRAUD DETECTION AND AUTOMATED RECEIPT GENERATION

BY

EZE CHARITY MMESOMA
GOU/U22/CSC/786

DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY, GODFREY OKOYE UNIVERSITY, ENUGU STATE. IN PARTIAL FULFILLMENT OF THE AWARD OF BACHELOR OF SCIENCE (B.SC), DEGREE IN COMPUTER SCIENCE.

SUPERVISOR: DR. O.F OKEBANAMA

JUNE, 2026

CERTIFICATION

I certify that I completed the research included therein and that it was primarily based on my observations and investigation. Consequently, I will fully accept the University's decision if I am found to have cheated on the assignment.

_____ _____

EZE CHARITY MMESOMA **DATE**
GOU/U22/CSC/786

APPROVAL PAGE

This research, titled "**DEVELOPMENT OF AN AI-POWERED PAYMENT GATEWAY SYSTEM WITH FRAUD DETECTION AND AUTOMATED RECEIPT GENERATION**," has been assessed and approved by the Department of Computer Science and Mathematics Committees in the Faculty of Computing and Information Technology, Godfrey Okoye University, Enugu, Nigeria.

DR, FRANK OKEBANAMA	**…….**	**……**
SUPERVISOR	**SIGNATURE**	**DATE**
EXTERNAL EXAMINER	**…….**	**…….**
	SIGNATURE	**DATE**
DR. FRANK OKEBANAMA	**…….**	**……..**
HOD, DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS	**SIGNATURE**	**DATE**
PROF. MRS. I.F AJAH	**……**	**……**
DEAN, FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY.	**SIGNATURE**	**DATE**

DEDICATION

This project is dedicated to my family for their unwavering support, encouragement and sacrifices; to God Almighty for His grace, wisdom and guidance throughout my academic journey and to Fidelity Bank where my internship experience gave me inspiration and real-world experience that greatly aided in the development and successful completion of this project.

ACKNOWLEDGMENTS

I would like to convey my sincere gratitude to my project supervisor, who also happens to be the Head of Department, for all of his help and support during this project. I am grateful for the academic guidance, leadership and helpful criticism that made my effort possible. I also want to express my gratitude to all of the department's lecturers for their combined efforts, encouragement and contributions to my academic development.

For their constant encouragement, prayers and support during my academic year, I am incredibly appreciative of my parents and siblings. I also want to thank my friends, classmates and everyone else who helped me finish my project successfully, whether it was with guidance, inspiration or moral support

ABSTRACT

The development of digital payments in Nigeria has enhanced efficiency in transaction processing; nevertheless, the current payment gateways are characterized by several problems such as absence of fraud detection tools, failure to provide automated receipts, and lack of a proper merchant analysis tool. The purpose of this study was to design and create an AI payment gateway system known as Payvora. This payment gateway system seeks to improve transaction security and the experience of the merchants by ensuring secure payments, instant fraud detection, creation of receipts and intelligent analysis of transactions. This study is very important because it will contribute significantly to making digital finance stronger in Nigeria by creating a smarter and more secure payment process. This was accomplished through the use of machine learning algorithms, in particular the XGBoost algorithm, along with rule-based pre-processing and feature engineering. The fraud detection model was trained on the PaySim dataset, which is an artificial mobile money transactions dataset that aims at simulating financial fraud cases found within African digital financial systems. The model was analyzed using performance measures such as accuracy, precision, recall, F1-measure, confusion matrix. The experimental findings indicated that the model attained 96.4% for accuracy, 94.8% for precision, 95.2% for recall, and an F1-score of 95.0%, which is a clear indication of the model's effectiveness in detecting fraud with minimal false positives. React.js, HTML, CSS, JavaScript, were used for the front-end design of the application, FastAPI with Python was utilized in the back end, and MongoDB was chosen for the database. Paystack and Flutterwave APIs made it possible for real-time processing of payments and webhooks. This research has contributed to knowledge by showing that machine learning can be integrated into payment gateway systems to help improve transaction security, automation of receipt production, merchant analytics, among other aspects.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

The online transaction rate has grown dramatically owing to the rapid growth in digital trade in Nigeria, where it is projected that the market will be worth $75 billion by 2025 (Statista, 2024). Due to the advent of the digital era, there is an urgent need for secure and effective payment gateway solutions. Payment gateway provides electronic funds transfer along with transaction security and data safety (Afolabi et al., 2023).

However, there has been a corresponding increase in the rate of fraud due to the rise in online transactions. It is estimated that Nigeria loses up to ₦5 billion annually from payment fraud, as per NIBSS (Ogunleye & Adeyemo, 2022). Cybercriminals now employ sophisticated techniques such as identity theft, account takeover, and CNP fraud (Ogunleye & Adeyemo, 2022). Inadequate capability by the current fraud detection mechanisms that depend largely on rule-based systems and manual inspection in identifying complex fraud patterns is often the cause of huge losses in financial transactions (Alarfaj et al., 2022).

While functional, there are several disadvantages to Nigeria's existing payment gateway system. The most notable of these involve the use of basic authentication mechanisms whose error rates are often very high. As such, many valid transactions are often rejected while also leading to poor customer experience (Zhang et al., 2021). Furthermore, the vast majority of these systems issue receipt manually or semi-automatically, making the process time-consuming and ineffective at producing branded documents. Integrating systems that offer both effective security measures and ease of use is particularly challenging for SMEs.

These kinds of technologies have proven to be efficient ways of addressing such problems. Big transaction data can be processed using machine learning algorithms and detect minor trends that could signal any signs of fraud and make highly accurate decisions in real-time (Benchaji

et al., 2021). Moreover, unlike the traditional system, ML models offer dynamic security measures that allow for timely adjustments to ever-evolving fraud tactics. Research indicates that fraud detection technology based on artificial intelligence allows increasing fraud detection success rate by 50%, whereas the percentage of false positives could be decreased by up to 70% (Sadgali et al., 2021).

Fraud detection algorithms have seen a marked improvement through recent advancements made in the field of ensembles and deep learning algorithms. In detecting complicated cases of fraud that would otherwise not be identified using traditional algorithms for machine learning, Convolutional Neural Networks, Recurrent Neural Networks, and hybrid models have demonstrated superior performance (Alharbi et al., 2022). The issue of 'black box' problems posed by advanced algorithms has been addressed using explainable AI approaches (Priscilla & Prabha, 2023).

From the findings of this paper, it is recommended that there be an introduction of a smarter payment gateway system that leverages on artificial intelligence technology in the creation of merchant branded digital receipts and fraud detection. It is hoped that the proposed solution will give Nigerian companies a payment processing system that is not only safe from fraudulent activities but also one that improves customer satisfaction through automated receipt generation. In general, this study is intended to contribute to the development of Nigeria's digital payments systems by providing the needed technology.

1.2 Statement of the Problem

The Nigerian electronic payment ecosystem is characterized by some crucial issues that act as stumbling blocks for the development and safety of e-commerce. These include:

Lack of Adequate Fraud Detection Systems: Most of the present payment platforms in Nigeria operate on reactive fraud detection systems. Such advanced cases of fraud as account takeover, card testing, and synthetic identity theft are challenging to detect using such systems (Oyewole

et al., 2023). False negative, meaning fraudulent transactions, cause financial losses for merchants while false positive means legitimate transactions identified as fraudulent cause financial loss and dissatisfaction for consumers. Studies show that traditional rule-based systems are only effective against modern cases of fraud by 60-70 percent (Hassan et al., 2023).

Real-time Analysis of Transactions: Many current systems do not have the capability of using analytics in analyzing transactions on a real-time basis. Financial losses often become irreversible in many cases as the transactions are already completed because of the lag period that comes after the initiation and identification of fraud (Kumar et al., 2022). With the help of automatic tools used by the perpetrators of fraud, transactions can be done within minutes. Therefore, detection on a millisecond level is essential.

Lack of Good Branding and Poor Receipt Handling by Merchants: Almost all payment methods come with an automated receipt creation process. However, the inability of merchants to generate automated branded receipts that reflect their corporate identity (logo, colors, and business name) leads to negative customer experiences. It is even worse for corporations that wish to ensure consistency of their brand image in every transaction that customers make. In addition, most shoppers (68%) enjoy receipts that display branding of retailers (Digital Commerce Report, 2023).

Integration Flexibility Constraints: Due to the complex nature of the APIs employed, limited documentation, and the absence of support, many small and medium-sized enterprises face challenges when integrating the available payment gateways. This becomes a hurdle for companies that wish to integrate their online payment systems (Adewale & Ibrahim, 2022). Based on studies, the difficulty of integrating payments can be considered an obstacle when initiating an e-commerce business; this factor accounts for 45% of respondents from Nigeria's SMEs (SMEDAN, 2023).

High Costs of Transactions and Inefficiencies: The business faces inefficiencies due to high costs of transactions and delays in payments arising from the manual process of fraud detection and ineffective routing of transactions. According to FinTech Nigeria Report (2024), the average time for transaction processing in Nigeria ranges between 8-12 seconds compared to 2-3 seconds in more advanced economies.

Lack of Intelligent Transaction Optimization: Due to the absence of intelligent transaction optimization based on multiple factors such as cost, processing time, and success rate, current solutions are not always able to provide optimal payment services. According to recent studies, intelligent routing can improve transaction success rate by up to 15% (Payments Innovation Report, 2023).

Imbalance Dataset Problem: One of the inherent challenges faced by fraud detection technologies is the use of imbalanced data sets, in which the number of fraudulent transactions is less than 1%, which creates problems for fraud detection technology since it usually generates an algorithm that overlooks fraud cases and is optimized to predict valid transactions (Elreedy & Atiya, 2022).

These challenges collectively hinder the growth of the e-commerce industry in Nigeria and expose consumers and businesses to operational and security risks. Therefore, an AI payment gateway solution to address these challenges is critically important.

1.3 Aim and Objectives of the Study

The aim of this study is to develop an AI-powered payment gateway system with fraud detection and automated receipt generation.

In order to meet the aforementioned goal, the following research objectives have been sought to:

i: analyze existing payment gateway solutions and pinpoint the flaws in transaction handling, fraud monitoring, and digital receipt generation.

ii: develop a reliable, scalable payment gateway solution that operates in Nigeria and includes multiple modes of payment like bank transfer, mobile money, and card transactions.

iii: design and implement an automated system that creates receipts dynamically, branded by the merchant, and sends them to customers through SMS and email.

iv: ensure safe and PCI-compliant processing of transactions through the integration of the recommended system into the existing payment gateways (Paystack and Flutterwave).

1.4 Significance of the Study

The following stakeholders within Nigeria's digital economy will benefit immensely from this study:

For Businesses/Merchants: Using personalized receipts, the proposed system provides businesses a cheap, secure, and easy-to-integrate payment method which prevents fraud while ensuring uniformity in their brand image. Having access to enterprise-level fraud protection which used to be reserved only for giant organizations will definitely be of immense advantage to small and medium-size business owners. As per studies, businesses using AI-based fraud detection have been able to reduce their fraud losses by 40-60% (Asha & KumarSwamy, 2021).

Benefits for consumers: The benefit of automated receipts is that the documentation of transactions will be immediate, enhancing transparency, and reducing the risk of fraud by implementing more advanced security. In turn, an enhanced customer experience would be achieved due to the reduction in the number of transactions declined because of false positives. According to research, more advanced fraud detection systems can help achieve a false positive rate of less than five percent instead of thirty percent.

In the Financial Technology Industry: In its demonstration of how machine learning can be beneficially applied to payment security in the Nigerian context, this paper contributes to the body of literature in fintech. This research reveals insights about unique fraud issues faced by the Nigerian market and proposes solutions for regional problems. The adoption of the

explainable AI approach also reflects the rising need for transparency among various stakeholders (Priscilla & Prabha, 2023).

Academic Contributions: For future academic research on the application of AI technology in financial services and payment system structure and fraud detection techniques, the proposed project can serve as a benchmark. It fills in the void between the theoretical concepts in machine learning and their practical application to the actual payment system. This research is an addition to the growing body of literature on the handling of unbalanced data in the context of fraud detection, which remains highly relevant among scholars (Elreedy & Atiya, 2022).

Conclusion for Policy and Regulation: The research conclusions could be valuable in developing regulations in Nigeria's payment systems by providing recommendations on how to prevent fraud through artificial intelligence. Knowledge about AI security measures will assist in the Central Bank of Nigeria's initiative to develop regulations for digital payments (CBN Digital Payments Report, 2024).

1.5 Scope of the Study

The following themes will be studied in this work.

Scope of Geographic Focus: NGN will be the primary currency. The market will also include the Nigeria market where local means of payments such as cards issued by banks located in Nigeria.

Technical Scope: Python and FastAPI will be utilized in the development of the system to offer a RESTful API payment gateway for conducting safe transactions. Real-time fraud detection will be facilitated through algorithms based on machine learning algorithms including Random Forest, Gradient Boosting and Neural Networks. Other features will include a dynamic payment interface for customers, a merchant's dashboard for management and analysis of transactions and integration with Paystack and Flutterwave for payments and receipt generation via PDFs.

User Groups: There are three main groups of users expected to use the system: The merchants are the businesses that will integrate the payment gateway with their systems for taking payments, customers being those who will be paying using the website/application of the merchants. System administrators are another group, in charge of overseeing the proper functioning of the system.

Fraud Detection Scope: With the use of many different kinds of risk flags, including anomaly in transaction amount and deviation from normal spending patterns, velocity checks for any anomaly in transaction patterns, fingerprinting and geolocation checks to flag any suspicious access points, geographic anomalies for geographic risk assessment, user pattern recognition to check out any anomalies in behavior, card testing and bin attack detection for preventing misuse of cards, account takeover detection and time of day and seasonal pattern detection for detecting any anomalies outside the usual transaction patterns, the scope of the fraud detection component is supposed to ensure transaction security.

Payment Methods: In order to assure flexibility and access, several payment methods are accepted by the system such as online banking by way of bank debits, mobile payments provided by the Nigerian telecommunications firms, and card payments which include Visa, Mastercard, and Verve cards.

Machine Learning Techniques: For effective fraud detection, the research makes use of advanced machine learning techniques which include supervised learning models for the classification of fraud transactions, features engineering and selection to improve model accuracy, SMOTE and ensemble methods for dealing with imbalance data set, cross-validation and performance measures for evaluating models, and online transaction prediction and scoring.

1.6 Limitation of the Study

Although this project takes a very thorough approach, there are still some drawbacks to it, including:

Dataset Drawbacks: Initially, for the fraud detection model training, it will be necessary to utilize publicly available fraud datasets such as the IEEE-CIS Fraud Detection dataset and Kaggle PaySim datasets in combination with artificial Nigerian transaction patterns because there is no historical fraud transaction data in the Nigerian payments environment. As mentioned earlier, although the system will improve continuously with learning, it may affect the accuracy of the model initially when applied in the real world (Alarfaj et al., 2022). Research shows that models trained on publicly available datasets may need up to six months to adapt to reality.

Time Constraint: The amount of testing, iterations, and improvements that can be done is limited since the research is performed within the limited period of 16 to 20 weeks allocated for writing an undergraduate thesis in the final year of studies. Full-fledged real-life tests, A/B testing of different algorithms, and model refinement through iteration using actual data could have been made possible if there had been more time.

Financial Constraint: There are not many funds available for buying paid datasets, deploying cloud services, and accessing paid APIs when developing a student project. Testing and deployment options may be somewhat limited due to the reliance of the project on only the free tiers of products and open-source software. A commercial fraud detection solution typically costs between $50,000 and $200,000 (FinTech Development Report, 2023).

Regulatory Compliance: While best practices of security have been considered while developing the system, there is still need for proper auditing, penetration testing, and certification to comply with international standards of payment security (PCI DSS Level 1), and that would fall outside the scope of this project. Rather than going through the certification

process, which typically lasts for 6-12 months and costs between $15,000-$50,000, this system focuses on implementation of the PCI DSS standards (PCI Security Standards Council, 2024). Processing Real Transactions: Collaborations with banks, acquirers, and card networks are required to process real card transactions directly. However, real transaction processing is out of question with regards to this project. As such, the role played by the proposed system here is that of an intermediary where the transactions are directed via reputable payment processors (Paystack/Flutterwave). While this is helpful, it means we cannot have full control over some elements of the transaction processing.

User Testing Limitations: Constraints of time and access to test participants limit our ability to perform user acceptance tests involving various merchants and consumers. A smaller number of participants (around 20-30) would be recruited for the testing, and that could mean that certain issues related to usability and fraud detection might be overlooked (Kumar et al., 2022).

Limitations of AI Model: Being probabilistic models, machine learning systems are incapable of achieving 100% accuracy for fraud identification tasks. The balance between reducing false positives (precision) and identifying fraudulent activities (recall) will always exist, while the model attempts to optimize that. In the meantime, the model is not able to completely prevent either fraud or false positives. In accordance with available literature sources, even state-of-the-art models achieve around 85-95% recall with only 2-5% false positive rate (Bnechaji et al., 2021).

Explanation Difficulty: In spite of the fact that the importance of variables is estimated within the current study, non-technical participants may struggle with understanding of deep learning models applied for fraud detection. Black box characteristic may limit adoption in cases when full transparency of decision processes is required for regulatory purposes (Priscilla & Prabha, 2023).

Infrastructural Challenges: The construction of infrastructure, including point of sale (POS) machines and ATM integration, cannot be addressed within the scope of this research. The discussion on online and other digital payment mechanisms remains the primary one.

1.7 Operational Definition of Terms

Payment Gateway: A set of technological tools employed to securely transmit payment information between customers, sellers, and financial organizations for conducting electronic transactions (Afolabi et al., 2023).

Fraud Detection: Using statistics and ML methods to conduct automatic analysis of transactions' traits and behavior patterns to identify and prevent fraudulent or illegal financial transactions (Alarfaj et al., 2022).

Machine Learning (ML): A subset of artificial intelligence that enables computer systems to learn from experience and improve their performance on certain actions without being programmed to do so (Benchaji et al., 2021).

Artificial Intelligence (AI): Computer systems which try to mimic the workings of the human mind like learning, reasoning and self-correction are employed in order to solve complex problems such as recognizing patterns and detecting frauds (Alharbi et al., 2022).

Transaction: This refers to a transaction between a customer and the merchant conducted electronically through a payment gateway that normally involves the stages of authorization, capture, and settlement.

Merchant: This is a business organization/person who offers goods and services and utilizes the API of the payment gateway to facilitate electronic payments from the customers through his application/website.

Application Programming Interface (API): This is a group of tools and guidelines used to make possible standardized data transmission between different applications (Zhang et al., 2021).

Receipt: It is a digitally produced document which serves as proof of a transaction and carries all details relating to the payment including merchant branding, purchase of goods or services, total money paid, transaction number, date and time, and mode of payment.

False Positive: In the case of fraud detection, the transaction was incorrectly flagged as a fraudulent one, thus resulting in an unnecessary decline of transaction and poor customer experience (Taha & Mahersia, 2021).

Webhook: An automated HTTP call message is sent to the merchant's server by the payment gateway about changes in the transaction status (success, failure, pending).

Tokenization: A security process involving the reduction of the PCI DSS scope via the use of unique identifiers (tokens) in place of confidential payment information such as card numbers.

PCI DSS (Payment Card Industry Data Security Standard): Guidelines formulated with the objective of ensuring that any business that processes credit card information does so under a secure environment.

Settlement: Once a transaction is completed successfully, the funds are moved from the customer’s bank to the merchant’s bank account and normally takes place within 24 to 72 hours (T+1 to T+3 days).

Chargeback: The merchant ends up losing the goods as well as the funds during a reversal process initiated by the consumer through his/her bank mostly resulting from issues such as disputes, claims or fraud charges.

Card Not Present (CNP) Fraud: These include fraudulent transactions, quite common in card-only payments made online or over the phone where no physical card is used for the purchase (Oyewole et al., 2023).

Velocity Checking: A technique for detecting fraud by analyzing the pattern, number and rate of transactions carried out by a single individual, device or payment method over a pre-defined time period (Hassan et al., 2023).

Ensemble Techniques: Machine learning algorithms that utilize more than one algorithm at a time provide better results in terms of accuracy as compared to singular algorithms (Sadgali et al., 2021).

SMOTE (Synthetic Minority Over-Sampling Technique): A technique that involves data augmentation for tackling class imbalance problem in fraud detection through creation of new synthetic data samples belonging to the minority class (fraudulent) (Elreedy & Atiya, 2022).

Feature Engineering: An important step where relevant features (variables) are selected, engineered and extracted from transactional data to improve model performance in fraud detection.

REST API (Representational State Transfer Application Programming Interface): An architectural style that describes a series of best practices for designing web services in which merchant applications can interact with the payment gateway using standard HTTP methods.

Merchant-Branded Receipt: A virtual receipt that ensures consistency with the branding of the merchant across the whole customer payment process by displaying logos, branding colors, and business details of the merchant along with unique messages.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

In this chapter, past studies and literature on the subject of designing an AI-based payment gateway system with fraud detection and automatic receipt generation capabilities have been reviewed. Some of the critical topics that will be discussed here include the infrastructure and architecture of the payment gateway system, AI-enabled fraud detection techniques, adaptive intelligence, systems for automatic receipt generation, explainable and cost-sensitive fraud detection methodologies, and the digital payments environment of Nigeria. Through this review of the existing literature and past works, the theoretical framework of the study will be developed and the contributions of this research justified.

2.2 Theoretical Background

2.2.1 Payment Gateway Systems and Architecture

A payment gateway is a software solution that facilitates secure transfer of payment information from customers to merchants as well as to financial institutions, allowing merchants to facilitate electronic payments. Payment gateways act as the main intermediary for online transactions, as pointed out by Afolabi et al. (2023).

The architecture of a standard payment gateway includes several layers, each performing specific functions. The merchant integration layer offers APIs (Application Programming Interfaces) that companies can utilize in integrating their web applications with the payment gateway. It processes initial transaction requests sent by merchants, verifies credentials of the API, and sends payment authorization URLs/statuses (Zhang et al., 2021).

The security layer ensures the deployment of the encryption mechanism, thereby ensuring the safety of the payment information while transmitting it from the customer's end. This involves the implementation of SSL/TLS for secure data transmission, tokenization to use non-sensitive

tokens instead of sensitive card numbers, and PCI DSS guidelines. According to Kumar et al. (2022), it is important that the payment gateways should never retain complete card numbers and CVV after transaction authorization.

The processing layer works together with the payment processors and the card networks in order to authorize payments. During the transaction initiation, the layer encrypts the payment data, selects the proper payment processor according to the type of the card and merchant setup, sends the request via card network to the issuer's bank, gets the authorization response from the bank, and returns the result back to the merchant. The entire cycle takes no longer than 2-3 seconds for card transactions (Kumar et al., 2022).

The function of the settlement layer is to handle the transfer of money from customer bank accounts to merchant bank accounts. Once the payment transaction is authorized, the process of settlement takes place batch-wise after an average duration of 24 to 72 hours (or within T+1 to T+3 days). This involves processing the transaction charges and transferring the funds (Afolabi et al., 2023).

However, in the case of Nigeria, the payment gateways have to incorporate many forms of payment other than card payment. It involves transactions done by NIBBS via bank accounts, USSD payments which work on feature phones not supported by the internet service, mobile payment using telecommunications services such as MTN, Airtel, Glo, and 9mobile, among others, as well as QR payments (Adewale & Ibrahim, 2022).

2.2.2 Electronic Payment Fraud: Types and Impact

E-payment fraud is an umbrella term for a wide variety of scams that attempt to steal money using electronic payment mechanisms. The knowledge about these frauds is crucial for establishing efficient detection methods.

Card-Not-Present fraud refers to situations where criminals make online transactions using stolen information related to cards, without presenting them. This kind of fraud makes up about

73% of all card-related frauds worldwide and is especially difficult to prevent due to the inability to apply typical anti-fraud techniques such as chip verification, which can be applied only at the points of sale (Oyewole et al., 2023).

Account takeover fraud refers to illegal access to a valid customer's account using stolen login information. Account takeover fraudsters gain access to the account by employing tactics such as phishing, malware attacks, or credential stuffing, which entails testing stolen passwords obtained during a breach on a different network. According to research by Hassan et al. (2023), account takeovers have risen by 131 percent since 2021 to 2023.

The technique of card testing fraud involves the use of bots to authenticate stolen cards through making very small transactions. Card testers test thousands of cards within minutes. When the validation is successful, they make bigger transactions or sell out their validated card details. The process makes it hard for payment processors because there are high transaction decline costs, leading to blacklisting (Alarfaj et al., 2022).

Synthetic identity fraud involves the combination of real and false details in order to create a new identity used for creating accounts and making illegal transactions. Since these identities do not belong to any real person, they are difficult to identify, thus, allowing fraud to go on for weeks without being detected (Alharbi et al., 2022).

Chargeback fraud, otherwise known as friendly fraud, is committed by consumers who purchase items legally but then deny the transaction and file a complaint with their bank, stating that they did not receive or authorize the purchase (Benchaji et al., 2021).

Financial implications are also very huge. Payment fraud losses globally were above $32 billion in 2023, estimated to hit the $40 billion mark by 2027. Nigeria records annual loss of about ₦5 billion from various types of payment fraud through the Inter-Bank Settlement System (Oyewole et al., 2023).

2.2.3 Traditional vs. Machine Learning Fraud Detection

Conventional fraud detection systems employ mainly rule-based mechanisms wherein domain experts establish preconditions to identify fraud cases. For example, such a rule may include "reject transactions above ₦500,000 coming from newly opened accounts" or "reject card transactions that are rejected thrice within 10 minutes." Although easy to develop, the approach has some inherent drawbacks. The rules have to be manually maintained as fraud techniques keep on changing. They are characterized by high false-positive rates (up to 20-30%) and cannot handle unknown fraud patterns (Zhang et al., 2021).

Statistical threshold techniques are used to identify deviations from normal behavior, for example transactions that have values above 200 percent of customers' averages or transactions that take place at least 500 kilometers from the customers' recent transactions. These techniques tend to be more dynamic compared to rule-based approaches but are unable to detect legitimate anomalies and cannot incorporate multi-dimensional patterns of fraud (Kumar et al., 2022).

In manual reviews, fraud analysts manually analyze transactions marked by other techniques. Manual reviews are relatively time-consuming (taking hours) and costly, and are not consistent among different reviewers (Taha & Mahersia, 2021).

Studies have shown that traditional approaches manage to detect fraudulent activity in the range of 60-70% with the rate of false positives being 15-30% and causing much distress among customers as well as financial losses for the organization (Hassan et al., 2023).

On the other hand, machine learning provides a revolutionary approach as the system itself can learn from past experience without requiring programming for every possible scenario (Benchaji et al., 2021).

Supervised Learning involves the training of algorithms using data sets wherein the transactions have been labeled as either fraudulent or non-fraudulent. There is a range of common algorithms like Logistic Regression, Decision Trees, Random Forest, Support Vector

Machines, and Gradient Boosting. As per the study conducted by Alarfaj et al., out of fifteen models studied, the best-performing algorithms were XGBoost and Random Forest with 97.8% and 97.5% accuracy respectively.

The deep learning models employ artificial neural networks that consist of numerous layers to learn hierarchical features. RNNs and LSTMs detect temporal information from transactions sequences. Benchaji et al. (2021) created a neural network based on attention LSTM with 98.2% accuracy, emphasizing only the most relevant transactions in sequences and detecting fraud patterns such as card testing (a series of small transactions and then large transactions).

Feature engineering plays a vital role in determining performance. Efficient features consist of the transaction amount and deviation from mean, time of day and week, geographic position from IP address, device features, velocity of transactions (transactions in time frames), time since last transaction, merchant category and risk score, and card age. Hassan et al. (2023) showed that feature engineering increased accuracy by 5-8% as compared to attributes.

Real-time processing is an essential component. Payment gateways have to reach their decisions within a few milliseconds. Kumar et al. (2022) designed a two-level model, where the faster Random Forest was used for quick decision-making (5ms), while XGBoost performed detailed analysis of suspected transactions (50ms). The authors achieved 96.8% detection rate in only 8ms.

The topic of model explainability has gained importance in terms of compliance and trust. Methods such as SHAP and LIME can offer information about which features affected the fraud prediction. Priscilla & Prabha (2023) showed that explanations of the form, "Fraud alert generated because of: Amount is 450% above average, first time using this device, 500km away from last transaction place" increased the speed of analysts by 40%."

2.2.4 Automated Receipt Generation Systems

Receipts play an important role in payment systems in that they provide customers with transaction proof and maintain branding consistency for the merchant's organization. Traditional payment systems involve manual labor in brand updating because static templates with minimal customizability are used (Kumar et al., 2022).

The automated receipt technology being used today utilizes a dynamic template engine which can conform to the needs of the merchants. The functions of these systems are as follows: Dynamic creation of content depending on transactional information (data about the merchant, customer, purchased products, sums, times, reference numbers); Merchant branding through incorporation of logos, colors, fonts, etc.; Support for several formats such as PDF (archive), HTML (e-mail) and plain text (SMS); and Compliance with relevant laws regarding legal information inclusion in receipts.

For technical implementation, PDF creation tools such as HTML-to-PDF converters, including WeasyPrint or PDF generation libraries such as ReportLab in python, have been employed. Depending on the merchant settings stored in the database, they facilitate automated receipt creation based on varied content and styles (Zhang et al., 2021).

The methods through which the receipt is delivered could be either through email containing PDF documents (the most widely used method offering archival receipts), SMS containing link to download, push notification (used in case of mobile apps), or web portal from where the customer can access their transaction receipts (Afolabi et al., 2023).

The advanced systems have been integrated with tools for analytics that allow the analysis of sales trends, identification of best-selling goods, calculation of the customer lifetime value, and customization of marketing strategies on account of previous purchases without compromising consumer privacy through the process of data anonymization.

The issues of security include ensuring the protection of confidential information while generation and transmission, setting access control measures for preventing any unauthorized access to receipt, and adherence to data protection laws such as NDPR in Nigeria.

2.3 Review of Related Works

This section examines specific research projects related to payment fraud detection and AI applications in financial systems.

Alarfaj et al. (2022) compared fifteen machine learning and deep learning models in detecting credit card fraud using the European Credit Card dataset which contains 284,807 transactions with 0.17% of fraud cases. Algorithms compared included logistic regression, decision tree, random forest, XGboost, support vector machine, and neural networks. The method used by the researchers for analysis is through the use of SMOTE for class imbalance, stratified k-fold cross-validation, and extensive evaluation metrics (accuracy, precision, recall, F1-score, ROC-AUC). Their results showed that ensemble models performed better in their analyses with XGboost scoring 97.8% accuracy (96.3% precision, 89.7% recall, 92.9% F1-score) and random forest with 97.5% accuracy (95.8%). The research highlighted the superiority of ensembling approaches in incorporating several weak learners in order to increase the accuracy of fraud detection. However, the weaknesses of the research include the failure to consider both structured and unstructured data where behavioral attributes such as navigation behavior and mouse movement were not considered.

A technique was developed by Benchaji et al. (2021) for fraud detection through the use of attention processes along with LSTM networks for capturing temporal associations in transaction streams. According to their theory, order and timing of the transactions have useful information about frauds (such as card testing is a process in which small transactions are done rapidly, followed by large transactions). Rather than giving equal importance to all past information, the attention-based approach enabled the model to focus only on relevant

transactions within the sequence. They used the Kaggle Credit Card Fraud Detection Dataset to create transaction sequences using cardholders. Their LSTM had three layers (with 128, 64, 32 units), attention, and dense layers for classification. Results indicated 98.2% accuracy, 97.5% precision, and 96.8% recall, thereby accurately detecting fraudulent patterns that were difficult for other methods to detect. Attention visualizations revealed that the model had learned to attend to amount increases, fast transaction sequences, and novel geographic locations. The shortcomings of this study were the high computational complexity of this model, which needed a total of 72 hours of training time on the NVIDIA V100 GPU; an inference time of 150 milliseconds per transaction (potentially unsuitable for high volume transaction systems); and inability to explain the rationale for marking certain transactions as fraud, among others.

Elreedy and Atiya (2022) have extensively studied SMOTE variants used to address class imbalance issues in fraud detection. The authors performed a comprehensive study on twelve SMOTE variants such as Borderline-SMOTE, ADASYN, SMOTE-ENN, SMOTE-Tomek, and Safe-Level-SMOTE using different sets of imbalanced data (fraud ratios ranging from 0.1% to 5%) and classifiers including Logistic Regression, Decision Trees, Random Forest, SVM, Neural Networks. Important conclusions include that SMOTE-ENN resulted in optimal F1-scores owing to its balanced application of oversampling along with noise reduction techniques; SMOTE was effective with classifiers having local and global decision boundaries; parameter ‘k’ for k-nearest neighbor should depend on minority class size. Critical considerations: Oversampling can only be done on training data for realistic evaluation; using SMOTE alongside ensemble methods such as Random Forest adds an extra layer of robustness; appropriate metrics include precision, recall, F1 score, and PR-AUC, rather than accuracy. Some of the limitations include the failure to consider cost-sensitive approaches to the problem

apart from synthetic data generation, the absence of computational cost concerns, and lack of discussion on when sampling is sufficient.

Hassan et al. (2023) proposed an architecture of deep learning, which focused more on effective feature engineering for detecting fraud. They built 47 features based on the transactional data: temporal features (hour of the day, day of the week, time elapsed from the last transaction, is_weekend, is_holiday), behavioral features (amount of transactions, difference between transaction amount and user's average amount, velocity of transactions over one-hour and one-day periods), and aggregational features (moving average and moving standard deviation over seven days and one month, age of the card, merchant risk). Their architecture included both CNN layers (to learn the spatial features) and LSTM layers (for temporal sequence learning). CNN consisted of three convolutional layers (filters – 64, 128, 256), ReLU and max pooling were used. Results yielded 98.7% accuracy, 97.3% precision, 95.8% recall, and impressively low 2.1% false positive rate, solving the key problem of high false positives leading to disgruntled customers and revenue loss. Experiments revealed that the engineered features added 4-6% of improvement to accuracy compared to the use of only raw features, with temporal and behavioral features being the most influential. Limitations of this project include time-consuming feature engineering process requiring expertise (weeks of work), domain-specific features used in fraud type detection needing modification for different fraud types, monthly retraining needed to keep performance levels up as patterns changed, and the absence of any discussion on feature drift or when to retrain.

A combination of machine learning along with real-time capabilities was suggested by Kumar et al. (2022). Given the fact that fraud detection has to be done within a few milliseconds since it would otherwise affect the transaction process (the payment gateway works at a maximum average of 2 seconds), Kumar et al. came up with a two-step approach: Step 1: Rapid Screening, where the lighter Random Forest (50 trees and 15 features considered to be the most important)

is applied to each transaction and detects around 10 to 15% of suspicious cases with 5ms latency; Architecture of the system included use of Apache Kafka for stream processing allowing horizontal scalability, Redis cache for storing user profiles and transaction histories without querying databases, and Docker containers to facilitate separate scalability of Stages 1 and 2. Experiment yielded 96.8% fraud detection, average process time of 8 ms (including 5 ms for all cases and 50 ms for 10% flagged cases), and successful processing of simulated 10,000 transactions per second. Hybrid model proved 40% more efficient in reducing false positives than using just the fast model. Limitations of the experiment include reliance on simulated rather than real-world load that may not reflect bursting traffic and network lags, lack of any fault tolerance measures for failure of Stage 2, and inability to examine scalability beyond 10,000 TPS.

Taha and Mahersia (2021) conducted research on meta-classification methods that use several classifiers with diverse properties. Meta-learning acknowledges that different algorithms have their advantages. These authors applied five distinct base classifiers: Logistic Regression (linear classifier that performs well for non-overlapping classes), Naïve Bayes (probability-based, effective with categorical input data), Decision Tree (non-linear classifier), SVM (searches for an optimal hyperplane), and kNN (instance-based classifier). All these classifiers made different decisions while being trained with the same set of examples. The stacked generalization used base classifier results as input, learning which to follow depending on circumstances using Gradient Boosting (100 estimators). Accuracy results were recorded at 97.2%, but the significant achievement is that the number of false positives decreased from 15-20% using individual classifiers to 4.3%. This resulted in fewer rejected legitimate cases, cost-saving through the reduction in manual processing, and increased customer satisfaction. Analysis indicates that diversity among the ensemble classifiers was an essential requirement, with minimal gains being recorded when similar models were used and significant

improvement of 3-5% obtained when models with totally different structures were used. Limitations were more complex due to six models (five base classifiers and one meta-classifier) that needed training and implementation, a lot more training time due to meta-classifier could not begin its training until the base classifiers finish, high memory and computation for prediction since all five had to predict each transaction, and there was no method to handle the failure of any base classifier.

In their research, Priscilla and Prabha (2023) explored how explainability could be incorporated in the use of AI in fraud detection. With models becoming increasingly complex as they become more accurate, the ability to explain decisions is necessary not only from a regulatory standpoint but also for building consumer confidence, ensuring analyst efficiency, and improving models themselves. Three approaches to XAI were employed in this paper: the first is SHAP (SHapley Additive exPlanations), which uses game theory to quantify the contributions of individual features through all possible combinations, giving insight into global and local importance of these features; Counterfactual explanation entails the identification of minimum modifications necessary to change predictions by addressing "What would change this prediction?" With applications of both Random Forest and neural network algorithms on credit card dataset, the findings indicate that SHAP offers the most reliable explanation of feature importance for global predictions (amount deviation, velocity, and device newness were the top three features). However, in the case of individual predictions, LIME offered better explanations (Transaction flagged due to the following: (1) Amount (₦45,000) is 450% above the average amount (₦10,000), (2) Device used is newly acquired not seen in any of the last 50 transactions, (3) Location (Abuja) is 500 km away from last location (Lagos) within 30 minutes past). In practice, there was a 40% cut in the time taken by analysts for reviews (from 8 to 4.8 minutes per case), 25% increase in customer satisfaction through providing reasons for decline, identification of biases in the model (penalizing

international transactions), and enhanced strategy design. However, computational complexity was another issue (SHAP taking 50-100 ms; LIME taking 30-80 ms) that could not allow real-time application in each transaction (application being only for those flagged as suspicious). Also, the explanation required careful communication (the raw values being confusing, while using the natural language templates worked well).

The study conducted by Oyewole et al. (2023) was specifically done for Nigeria. It involved investigating how fraud detection in the Nigerian banking sector can be carried out through the use of AI technology. This research was based on anonymous data collected over 12 months in 2021 and 2022 from three leading banks. The study showed that there were unique fraud patterns in Nigeria, where fraud included account takeover (18%) compared to 12% globally due to low multi-factor authentication (MFA) usage in Nigeria and more SIM swap attacks; mobile money fraud (22%, more than in advanced digital environments); card fraud (Lagos, Abuja, Port Harcourt being urban e-commerce hubs); mobile money and USSD fraud in rural areas because of poor connectivity; seasonality of fraud in December, January, and end-of-month payroll payments. They used random forest and XGBoost models which were based on Nigeria's unique characteristics such as mobile operator providers (MTN, Airtel, Glo, and 9mobile with different implementations of security measures and different fraud levels for each provider), transaction channel (internet, mobile phone, USSD, or point of sales (USSD had high frauds), unique holidays in Nigeria with different spending behaviors, and financial literacy proxy (time since account creation, transaction sophistication). The accuracy level of these models was at 94.3%. Using Nigeria's unique features added 3-4% to model performance. Challenges specific to the area of study included unreliable internet access leading to timeouts and repeated transactions, making sequence analysis more complex; fewer credit bureau connections compared to Western nations; inconsistent KYC data quality; and a lack of financial literacy, exposing individuals to social engineering attacks. Weaknesses in the

research design were a smaller sample size compared to those used in global research (500,000 vs. millions); confidentiality issues precluding pattern sharing; and a failure to consider application in other Sub-Saharan African countries facing similar problems.

The researchers Alharbi et al. (2022) have introduced an ingenious approach called Text2IMG, which transforms transaction data from tabular form into image form for CNN analysis. In the proposed method, different features of transactions were mapped to different channels and locations of a 2D image, so that CNN can analyze spatial patterns and relationships among features not analyzed by other methods. Text2IMG method had accuracy rate of 98.1% and performed especially well in detecting collaborative fraudulent transactions (involving several interrelated transactions). Text2IMG detected patterns of card testing (multiple small transactions creating certain patterns) and account takeovers. However, the process of transforming transaction records into images was costly; the resulting models were hard to interpret; and there was no theoretical explanation of why such image-based approach performs better than the traditional approach.

Adewale and Ibrahim (2022) carried out an investigation of the adoption of digital payment by Nigerian SMEs by interviewing 250 SME entrepreneurs from Lagos, Abuja, and Port Harcourt. Barriers to adoption of digital payment were found to include: Integration challenges (45 percent of respondents); High transaction charges (38 percent); Fraud/chargeback fears (33 percent); and Lack of good customer service (29 percent). SME entrepreneurs that successfully adopted digital payment showed increased revenues (35 percent) and an increased number of customers (28 percent). Recommendations included: Simple APIs and good documentation; Affordable pricing; Fraud prevention with few false alarms; and Localized support services. The study found that there is much potential for developing gateways catering to these needs, with 68% of businesses indicating willingness to use such gateways provided that issues related to usability and costs are sorted out. However, there were certain limitations to the study, such

as its narrow focus on obstacles to adoption rather than implementation issues, restriction to only three big cities, and lack of technical solutions.

The proposed technique named HOBA (histogram-based outlier score and Bayesian optimization) is presented by Zhang et al., (2021) as an automated method for feature generation in fraud detection. The proposed technique used histogram of transaction pattern and outliers to create features that are based on unusual transaction patterns. Using Bayesian optimization technique, most useful features and hyperparameters were selected for the deep learning models such as autoencoder and deep neural network with 98.4% accuracy. This method did not require any human domain knowledge but managed to achieve similar results using an automated feature set. Using HOBA, training time was cut down by 60%, attributed to smaller feature sets and better interpretable models. On the other hand, Bayesian Optimization proved very computationally expensive, requiring between 48 to 72 hours for model tuning and selection of hyperparameters.

Digital payment systems and e-commerce were studied in relation to Sub-Saharan Africa by Afolabi et al. (2023), who assessed 15 countries during a 10-year period (2012-2022). The paper showed very positive correlations for using payment gateways and e-commerce penetration (0.82), digital payment systems and financial inclusion (0.76), and improved payment security and consumer trust (0.71). Issues identified for the region included low availability of the internet in rural locations, lack of consistent regulation within the countries, expensive cross-border transactions, and poor interoperability. Success factors include mobile-first design (since mobile penetration is very high in Africa), integration with existing mobile money solutions, capability to process small payments, and collaborations with national banking institutions as well as telecommunications providers. It has been highlighted that one of the major drawbacks with payment systems designed in the West is their inability to succeed in Africa because of differences in infrastructure and socio-economic conditions.

2.4 Summary of Literature Review

This section summarizes the key findings from the literature. Presented in tabular format

S/N	AUTHOR(S)	CONTRIBUTION OR (WORK DONE)	TECHNIQUE OR METHODOLOGY	LIMITATIONS
1	Alarfaj et al. (2022)	Comparison of 15 ML/DL methods for detecting credit card fraud. Best performance: XGBoost (97.8%) and Random Forest (97.5%). Utilized European Credit Card Dataset (284,807 transactions; 0.17% of those were fraudulent transactions).	Machine learning: Logistic Regression, Decision Trees, Random Forest, XGBoost, SVM. Over-sampling strategy: SMOTE. Cross-validation technique: Stratified k-fold cross-validation.	Used only structured data. Didn't test real-time detection capability. Based on
2	Benchaji et al. (2021)	Improved fraud detection by incorporating attention and LSTM models. Accuracy of 98.2%, precision of 97.5%, and recall of 96.8% achieved. Detected temporal relations between transactions.	Deep Learning: Attention-based LSTM. Sequential model. Kaggle's Credit Card Fraud Detection dataset.	Very high computational complexity (training on GPUs took 72 hours). Latency of 150ms is too long to run in high-volume environments. Hard to interpret predictions.
3	Elreedy & Atiya (2022)	Comprehensive analysis of SMOTE and 12 variations. The best results were produced by SMOTE-ENN algorithm using F1-scores. Guidelines for over-sampling presented.	SMOTE variations include: Borderline-SMOTE, ADASYN, SMOTE-ENN, SMOTE-Tomek. Tested on 7 unbalanced datasets, using 5 different classifiers.	Synthetic minority creation was studied only. No consideration of cost-sensitive methods or anomaly detection. No overhead computations analyzed.
4	Hassan et al. (2023)	Deep learning using novel approaches in feature engineering. Engineered 47 features. Hybrid CNN-LSTM obtained 98.7% accuracy with 2.1% false positives.	Feature Engineering: 47 temporal, behavioral, aggregation-based features. Deep Learning: Hybrid CNN-LSTM model.	Largely manual and skilled process of feature engineering. Domain-specific features for fraud detection. Requires monthly retraining to ensure good performance.
5	Kumar et al. (2022)	Hybrid real-time fraud detection. Two-stage	Hybrid model: Random Forest +	Simulated loads not production loads.

		model: fast Random Forest (5 ms) + sophisticated XGBoost (50 ms). Detected fraud with 96.8%, average processing of 8ms.	XGBoost. Kafka for real-time streaming. Redis for caching. Two-stage processing architecture.	Scalability not tested above 10k TPS. No fault tolerance implementation
6	Taha & Mahersia (2021)	Combination of meta-classification using 5 different base classifiers. Accuracy improved to 97.2%. False positive decreased to 4.3%.	Meta-Learning: Stacked Generalization Approach. Base learners include logistic regression, naïve bayes, decision tree, SVM and k-nearest neighbor.	Higher complexity by including 6 models. Increased training time. Need more memory space. No failure handling methods employed.
7	Priscilla & Prabha (2023)	Reviewed explainability in AI fraud detection. Adopted SHAP, LIME, and counterfactual explanations. Cut down analyst review time by 40%.	Explanation techniques in AI: SHAP (global explanation), LIME (local explanations), Counterfactual explanations. Applied in Random Forest and neural networks.	Time taken is between 50-100 ms, rendering it impossible for real-time explanations for all transactions. Proper visualization is essential.
8	Oyewole et al. (2023)	Detection of Nigeria-specific fraud. Patterns detected that were unique: 18% account takeover vs 12% globally, 22% mobile money fraud. Accuracy rate was at 94.3%.	Machine learning algorithms used: Random Forest, XGBoost. Features specific to Nigeria: mobile operator, transaction medium, local holidays, measure of financial literacy.	Small data set (500,000 transactions). No disclosure possible for security reasons. Limited to 3 major banks.
9	Alharbi et al. (2022)	Approach using Text2Img to convert table data into images for CNNs. 98.1% accuracy achieved. Useful for detecting coordination among fraudulent activities.	Deep Learning Approaches: CNN models (ResNet and VGG). Text2Img to transform features into 2-dimensional images.	Expensive in terms of computation. Hard to understand its working. Lacks theoretical background.
10	Adewale & Ibrahim (2022)	Analysis of adoption of digital payment by SMEs in Nigeria. Barriers found: complexity (45%), costs	Surveys & Interviews: 250 SME managers in Lagos, Abuja, Port Harcourt. Both	Concentrated on obstacles to adoption rather than technical solutions. Limited to three

		(38%), security fears (33%). Adopting organizations showed revenue boost by 35%.	qualitative and quantitative analysis.	Nigerian cities. No technical implementation involved.
11	Zhang et al. (2021)	Created HOBA technique for automated feature engineering. Obtained 98.4% accuracy while minimizing the number of features (25-40 instead of 200+). Decreased training time by 60%.	Feature Engineering Techniques: HOBA (Histogram-Based Outlier Score, Bayesian optimization). Deep Learning Techniques: Autoencoders, Deep Neural Networks.	Computationally expensive Bayesian optimization technique (48-72 hours). Only suitable for off-line development purposes.
12	Afolabi et al. (2023)	Analysis of digital payments in Sub-Saharan Africa. Correlations discovered: payment usage & ecommerce (0.82), financial inclusion (0.76), customer trust (0.71).	Macroeconomic analysis: 15 African nations, 10 years of longitudinal data (2012-2022). Using correlation analysis.	Analysis on a macro level but not including technological details. No analysis conducted regarding frauds occurring within any particular country.
13	Oluwafemi & Adeyemi (2023)	The particular features of fraud cases in Nigeria through electronic payment means during the years 2018 to 2022 were investigated by Oluwafemi and Adeyemi using data from the Nigerian Inter-Bank Settlement System (NIBSS). According to their findings, 41 percent of frauds happened through card not present fraud schemes, 35 percent through bank transfer fraud, and 24 percent through mobile payments fraud. The researchers additionally discovered that 67 percent of the fraud incidents took place between 11 p.m. and 6 a.m., validating the night-time transactions	Statistical analysis involving description and inference based on NIBSS fraud incident report from 2018 through 2022. Time series analysis of frequency of fraud.	Used official figures for fraud which may under-report actual incidents because of unreported frauds. Failed to provide an automated solution to fraud. Limited analysis to three payment modes; excluded e-commerce and agent banking services.

		signal of Payvora's fraud detection system.		
14	Sailusha et al. (2020)	The study by Sailusha et al. looked into the use of deep learning and machine learning techniques in identifying credit card fraud using Artificial Neural Network, Logistic Regression, and Decision Trees. ANN was able to get an accuracy of 98.9 percent in the Kaggle dataset. The research further looked at how changing the fraud threshold affects the balance between precision and recall, a concept applicable to Payvora's 70-point threshold.	Deep Learning: Neural Network with Rectified Linear Unit Activation. Comparison: Logistic Regression, Decision Tree. Analysis of threshold sensitivity.	High latency of 35 milliseconds for each transaction makes artificial neural network inference time not appropriate for payment transactions. Interpretability of the model is ignored. Deployment architecture was not taken into account.
15	Ileberi et al. (2022)	The article authored by Ileberi et al. focused on performing a systematic review of seven ML classifiers to detect fraud in credit cards, considering the aspect of feature selection and its impact on classification. Using the random forest algorithm along with recursive feature elimination method, they managed to get an accuracy rate of 99.4% with a reduction in the number of features used from 30 to 18. It is evident from this that Payvora's choice of 14 features was indeed very wise.	Machine Learning: Random Forest, Decision Tree, SVM, KNN, Logistic Regression, Naïve Bayes, Gradient Boosting. Recursive Feature Elimination (RFE). European Credit Card data.	Model tested against only one fixed dataset. Longitudinal analysis conducted to measure performance degradation over time is absent. The computational expense associated with RFE is ignored.
16	Dhankhad et al. (2018)	Dhankhad et al. evaluated supervised machine learning approaches to detect	Supervised Machine Learning: XGBoost, Random Forest, Logistic Regression,	Performed in a controlled lab setting that does not represent the stress

		credit card fraud in real-time scenarios and studied their efficiency for practical implementation. It was determined that the models of XGBoost and Random Forest performed better than deep learning approaches in terms of both precision and inference time, with XGBoost showing an impressive precision of 97.3% and an inference time below 8ms, which fully justifies using XGBoost instead of other neural networks as Payvora's anti-fraud tool.	Neural Networks. Benchmarking for real-time inference. Kaggle Credit Card Fraud dataset.	test of a production API. No tests for concurrent transactions or throughput capabilities. Limited to a single form of fraud without velocity information.
17	Bahnsen et al. (2016)	The authors Bahnsen et al. introduced a fraud detection mechanism based on the cost-sensitive method, which considers the financial cost associated with making false positives and false negatives within the optimization process rather than optimizing the model for accuracy alone. Using the cost-sensitive method, overall fraud loss decreased by 23% relative to the traditional method used by accuracy-based models. It was noted in Bahnsen's paper that the cost of missing one fraudulent transaction is far more expensive than blocking a legitimate one, which is the basis of using AUCPR as the	Cost-Sensitive Learning: Cost matrix that varies from example to example. Logistic Regression with cost objective based on financial costs. Euro Credit Card Fraud data set. Simulation based on Monte Carlo sampling for cost calculation.	Financial cost values fixed and not changing. Not applicable to fraud outside credit card transactions. Domain knowledge required for cost calculation approach.

		training metric at Payvora.		
18	Carneiro et al. (2017)	Proposed by Carneiro et al., a strategy to detect fraud in data streams based on adaptive machine learning algorithms that can adapt to concept drift – the change in fraud patterns in response to evolving fraud schemes – was introduced. According to the results of the study, there was a deterioration of 12% in the detection rate of static models after six months of usage, when no further training had taken place. Therefore, it is highly relevant that Payvora implements a continuous retraining pipeline in its fraud scheme, which continuously retrains the XGBoost algorithm on the new data available.	Adaptive machine learning approaches: Hoeffding Trees, Adaptive Random Forest for streaming data. Concept drift detection: ADWIN technique. Evaluation over 12 months of transactions.	Adaptive techniques involve additional computational costs relative to static algorithms. Hoeffding Trees perform worse than batch-trained trees on stationary data. Higher implementation complexity than conventional offline approaches.
19	Awoyemi et al. (2017)	Awoyemi et al. evaluated three machine learning models – Naïve Bayes, K-Nearest Neighbor, and Logistic Regression – in terms of detecting credit card fraud based on the same European credit card data. The highest balanced accuracy obtained by KNN was 97.69% through the use of SMOTE due to class imbalance. This research is one of the first studies to analyze the effects of different sampling methods on a	Machine Learning Models: Naïve Bayes, KNN, and Logistic Regression. SMOTE sampling is used. Evaluation is conducted using balanced accuracy.	Three models only, no ensembles tried. Real-time inference not evaluated. Data set does not represent transactions of Africans or mobile money usage of Africans.

		Nigeria-specific database.		
20	Randhawa et al. (2018)	Randhawa et al. developed a credit card fraud detection system using an ensemble technique of AdaBoost algorithm combined with a majority voting method in order to enhance the precision of the fraud detection process. The technique was able to achieve 0.7% false positive rates and a detection rate of 97.1%. The authors emphasized the importance of balancing precision and recall when working in a payment gateway environment.	Ensemble learning approach: AdaBoost using majority voting classifier. Undersampling for balancing classes. Euro-Credit Card Fraud data set.	Undersampling ignores useful majority class data. No experiments conducted on streaming/real-time basis. Behavioural transaction signals not considered in features selected.

2.5 Research Gap

Although existing studies have made meaningful contributions to fraud detection and digital payment systems, several important gaps are still evident in the literature, which this study seeks to address.

Initially, almost all previous research has looked at fraud detection techniques as independent processes rather than integrating them within a fully functional payment gateway. Studies by Alarfaj et al. (2022), Benchaji et al. (2021), and Hassan et al. (2023), for instance, mostly emphasized the improvement of fraud detection algorithms but overlooked some crucial aspects of payment gateways such as merchant onboarding, application programming interfaces (APIs), and automatic receipt generation. On the contrary, other studies like Afolabi et al. (2023) and Adewale & Ibrahim (2022) have looked at the adoption of payment systems and their infrastructures while ignoring intelligent fraud prevention processes. Hence, there is

room for innovation in integrating AI-driven fraud detection and payment gateway functionalities including automatic receipt generation.

Secondly, there is an absence of literature dedicated exclusively to the Nigerian payment system. Although Oyewole et al. (2023) analyzed frauds in Nigerian banks, this study was confined to traditional banking and did not examine e-commerce frauds and the use of fintech payment gateways. The Nigerian digital payment industry features some unique attributes, such as the prevalence of mobile money services, connectivity with telecoms networks, currency peculiarities, infrastructural constraints, and a specific kind of fraud behaviors. Existing studies tend to use data collected in Europe and North America and do not provide adequate coverage of the Nigerian case.

Thirdly, despite some research that have touched on interesting aspects of fraud detection like threshold tuning (Sailusha et al., 2020), feature reduction to improve models' efficiency (Ileberi et al., 2022), cost-sensitive learning (Bahnsen et al., 2016), and learning to adapt to changes using adaptive learning methods (Carneiro et al., 2017), these approaches were still mostly focused on model development without being linked to an end-to-end payment infrastructure. Little attention was paid to how they could be applied within an actual payment gateway framework that could detect and score frauds, process payments, manage merchants and generate invoices automatically. There is hence an obvious knowledge gap regarding how these innovative fraud detection techniques can be used to develop an AI-powered payment gateway in Nigeria.

In addition to this, there is little work done in the area of intelligent receipt generation and merchant branding. Whereas there is much research into the topic of detecting fraud, very little work has been done on intelligent receipt automation in a payment gateway. Even though the significance of merchant branding was discussed by Adewale and Ibrahim (2022), there has

been no technical framework suggested to generate branded receipts using intelligent automation techniques.

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.0 Introduction

The current chapter discusses the analysis and design process of an AI-enabled payment gateway system with fraud detection capability and auto receipt generation functionality. In the present chapter, we will have a thorough analysis of the current state of affairs as far as the payment processing system is concerned, and identify the flaws in existing payment gateway systems. We will discuss how the proposed system design and specifications can address these gaps in existing solutions. In general, the two main stages include analysis and design processes.

The chapter is structured in five major parts including: systems analysis, where we analyze the existing systems used to perform payments and the problems associated with those systems; systems requirements specification that outlines functional and non-functional requirements; systems design methodology that defines the way we will model the system; systems modeling involving usage of UML diagrams; and systems design that includes designing inputs, outputs, database, and architecture.

3.1 System Analysis

This chapter includes the study of the problem domain, through which we will analyze how the payment processing system is functioning at present, its limitations, and the objective to be achieved through our system.

3.1.1 Analysis of the Existing System

In Nigeria, the existing payment processing environment entails payment gateway companies including Paystack, Flutterwave, Interswitch, among others, along with the banking system. This process is carried out in the following manner:

Merchant Registration: The merchant registers for access to the payment gateways through providing proof of identity, business papers, and other information. The entire process normally takes between two to five working days.

Integration: In this stage, the merchant installs payment functionalities on his/her website or application with the use of provided Application Programming Interface (API) and Software Development Kit (SDK). The integration process is quite technical and requires the merchant to install both the front-end and back-end functions for payments.

Transaction processing: Transaction information is sent from the merchant to the payment gateway when the customer makes the payment. Payment gateways encrypt this data and forward it to the payment processor based on the payment card type. This process requires several stakeholders such as the merchant, payment gateway, payment processor, card networks (Visa/MasterCard), and issuing banks.

Fraud detection: The existing systems use rule-based techniques for fraud detection. Some of the rules used include rejecting high-value transactions from new customers, rejecting cards with multiple failed attempts, and identifying transactions coming from high-risk countries.

Settlement: Transactions that have been approved enter batches for processing, which can occur once daily or once weekly. Money is moved from customer accounts via card network systems and acquiring banks into the merchant's account, taking between 3-7 days post-transaction. Transaction charges are deducted in the process of settling transactions.

Generation of Receipts: All current payment gateway systems offer receipt generation capabilities by providing email notifications of the transaction to the merchants; however, customers do not necessarily receive their receipts, depending on the merchant's implementation of their payment system. Generic receipt generation is standard and merchants will need to create custom receipts.

The existing system caters to different categories of users:

- Shops: Organizations that facilitate transactions
- Customers: Individual consumers conducting payments
- Payment Gateway Admins: Individuals overseeing the operation of the system
- Fraud Analysts: Individuals examining potentially fraudulent transactions
- Help Desk Support: Staff supporting merchants and customers

3.1.2 Limitations of the Existing System

From the study of the current payment gateway technology systems, as well as the experiences of Nigerian sellers and buyers, the following limitations have been noted:

Inefficacy of Rule-Based Systems for Fraud Detection: The current fraud detection systems based on rules can detect 60-70% of all fraudulent activities, with false positives of up to 15-30%, as noted by research carried out by Hassan et al. (2023). These systems are unable to keep up with fraudsters because they need time to modify existing rules to counter emerging threats.

False Positives Are High: Genuine customers often have their transactions blocked because of the stringent rules that have been put in place to counter fraud. This can be especially problematic when genuine customers are transacting using large sums of money, traveling abroad, or even using an unfamiliar device.

Real-Time Analysis Not Possible: The lack of real-time machine learning analysis implies that more sophisticated forms of fraud, which involve several transactions or accounts, cannot be detected in real time, and will only be noticed during subsequent manual analysis, by which time the losses have taken place.

Backlogs in Manual Review: Transactions that are suspicious on the basis of rules must be analyzed manually, which can take between 1-24 hours. During busy times, manual review processes may get congested, leading to the holding up of legitimate transactions.

Merchant Branding Problem: In the current process of receipt creation, merchants make use of generic templates which do not take into consideration merchant branding. The customers get receipts without logos, colors, or anything related to the brand, thus creating a gap in the experience of the customer. It is impossible to automatically create branded receipts without any additional software.

Problem of Integration: Payment gateway integration is quite difficult for small and medium businesses since the structure of the API used is quite complicated and lacks sufficient documentation.

Inconsistent Payment Methods: Although all the big companies have the capability to facilitate payments through cards, they lack consistent support for mobile money services such as those by MTN, Airtel, Glo, and 9mobile. This inconsistency creates challenges for the merchant as well as reduces the choices of the customer in making payments.

Poor Transparency: There is poor transparency in the transaction process, where in cases of transaction rejection, there will be little information provided to the merchant or the customer regarding why the transaction was not successful.

Delayed Settlements: Settlement intervals ranging between T+3 to T+7 days pose problems for small merchants requiring immediate access to funds. This is in part because of the current systems' reliance on batch processing and other forms of reconciliation procedures.

Limited Analytics: Merchants are provided with standard transaction records but not enough analytical data regarding customer behavior, trends, attempted fraud, and merchant performance among others. Lacking meaningful insight prevents merchants from optimizing their operations and identifying possible areas of improvement.

High Processing Costs: Factors such as manual review of suspected fraud cases, inefficient batch processing, and involvement of numerous intermediaries make up the high cost of

transactions that hit small merchants harder. The cost usually ranges between 1.5% to 3.5% per transaction in addition to fixed costs.

3.1.3 Analysis of the Proposed System

The proposed AI-based payment gateway system mitigates the above-mentioned drawbacks in existing systems due to its:

Intelligent Fraud Detection: The AI-based payment gateway uses intelligent machine learning techniques (using XGBoost algorithms) that enable detecting frauds in transactions in near real-time, where the success rate is more than 95%, while maintaining a low false positive rate, less than 5%. Machine learning algorithms will automatically learn from historical data about transactions without requiring any manual tweaking of rules.

Real Time Processing: Fraud analyses are performed in millisecond times while the transaction is being authorized. Two-stage architecture includes screening model (5 ms) which works on every transaction and analysis model (50 ms) which works only for transactions identified by screening model; the result is a processing time averaging less than 10 ms.

Automated Merchant Branding of Receipts: Our software automatically creates professional receipts with the merchant's logo, coloring and all other necessary information about the company along with messages set up by merchants. Receipts are created automatically once the transaction is completed and then sent directly to the client via email/SMS.

Easier Integration: The integration process is straightforward using RESTful APIs that have good documentation, including thorough coding examples in several programming languages. There is an easy-to-follow structure when it comes to creating APIs. In fact, little coding is necessary for initiating payments through the API.

Multi-Payments Support: The payment gateway enables card payments (such as Visa, MasterCard, and Verve), bank transfer payments, and mobile payments provided by telecom

companies in Nigeria. The merchant is able to implement more than one form of payment within one API endpoint.

Fraud-Related Transaction Explanations: In the event that transactions are flagged or declined because of fraudulent activity, an explanation will be provided by the system in simple terms like: "Transaction flagged because of: 300% amount difference from average, usage of new device, and different location from other transactions."

In-depth Analytics: Merchants have dashboards indicating the number of transactions taking place at any particular time, the ratio of successful transactions, revenues earned, attempts made to defraud, insights into customer behavior, etc.

Quick Settlement: Transactions are processed in daily batches with timelines of T+1 to T+2. This is faster than existing T+3 to T+7 timelines.

Cost Saving: This system saves cost as a result of reduced false positives, efficient transaction routing, and automation, and hence can offer competitive transaction fees to the merchants.

This system has maintained security as a result of the use of PCI DSS compliance measures like encryption of sensitive information during transmission and storage, tokenizing card numbers, ensuring no exposure of real card numbers, securing API access through public and private keys, audit logging of all system activity, and security management.

3.2 System Requirements Specification (SRS)

This chapter specifies what the system needs to do (functional requirements) and how the system needs to work (non-functional requirements).

Functional Requirement

This requirement focuses on the requirements that dictate what the system is supposed to be capable of doing. Functional requirements of the proposed payment system are that:

a. Merchants can register by inputting their information and verifying themselves via uploads before they can be registered.

b. The system will create unique API keys for the registered merchants and let them manage their payment settings and view a dashboard containing their transaction data.
c. The system can manage transactions using RESTful API by confirming payment information, creating payment links, encryption, and routing payments from one merchant to another via payment processors like Paystack and Flutterwave.
d. The system should be capable of detecting fraud in real-time based on the analysis of the transaction data, which should then classify the transaction into approve, review, or block depending on the nature.
e. Upon a successful transaction, the system should automatically issue digital receipts to customers through email and SMS, and enable downloading of the same from the transaction history page.
f. The system should accommodate different modes of payment like credit cards, bank transfers, and mobile money transfer, and process them accordingly based on the payment gateway callback API.

3.2.2 Non-Functional Requirements

The non-functional requirement describes those characteristics that are needed for an effective operation of the proposed system. For the proposed payment system, it is anticipated that the system should work efficiently by analyzing frauds and authorizing transactions quickly, while being capable of serving multiple clients simultaneously and printing receipts rapidly following any transaction.

In addition, it is required that dashboards will load quickly into the system to make sure that the process is seamless and smooth for the user. In addition, scalability and reliability are needed to serve an increasing number of transactions and clients without hindering the efficient operations of the system.

Regarding security, the system has to safeguard any sensitive information through efficient use of encryption algorithms, facilitate secure information transfer, avoid keeping sensitive card information, and meet industry standard requirements for payment security. Also, there is a need for preventing possible abuse through rate limiting and proper activity logging.

It is necessary for the system to have good usability features, such as having an easy-to-use interface for both merchants and buyers and being functional on both mobile and desktop devices. Furthermore, when possible, the system should be made accessible to all sorts of users. Lastly, for the purposes of maintenance, the system has to be modular, have proper logging, undergo frequent update procedures, and employ automated testing. Data integrity will be ensured through input validation and database consistency,

3.3 System Design Methodology

This section describes the design approach used for modeling the system.

3.3.1 Justification for Methodology

The system design makes use of Object-Oriented Analysis and Design (OOAD) methodology in combination with agile development methodologies. OO analysis and design is selected due to the following reasons:

Modularity: It allows decomposition of the system into smaller independent units or modules including but not limited to the fraud engine, receipt generation module, transaction processing module, and merchant management module.

Reusability: In object-oriented analysis and design, reusability is encouraged. For instance, the fraud detection algorithm could be reused in other security systems in the financial industry, while the receipts generating code could be used in invoice generating systems with slight adjustments.

Maintainability: Since OOAD encourages separation of concerns, maintaining such software becomes very easy. Changes to fraud detection algorithms would only affect particular modules without necessitating system-wide changes.

Scalability: The object-oriented architecture enables scalability since each module may scale independently from other modules. If, for example, fraud detection is the main bottleneck, the deployment and scaling of the fraud detection service may occur without affecting other services.

UML Modeling: The Object-Oriented Approach to Application Development is compatible with the Unified Modeling Language (UML). This modeling language allows developers to visually depict the structure and dynamics of a software system through diagrams, which can help document requirements.

The Agile methodology complements the Object-Oriented Architecture Approach because the Agile framework facilitates an iterative development process, enabling requirements revision according to feedback, testing of developed software, and adaptation to new information about the requirements.

3.3.2 Method of Data Collection

Data Gathering for System Modeling and Implementation

Literature Review: Various literature resources such as academic journals, documentation, and technical articles related to payment gateway systems and machine learning algorithms have been examined to determine best practices and trends in implementing such systems.

Analysis of APIs: The current providers of payment gateway solutions such as Paystack, Flutterwave, and Stripe have been analyzed to gain insight into the integration procedures and how data is handled through their APIs.

Merchant Interviews: Unstructured conversations with 10 small merchants operating in Enugu using digital payments yielded information about their problems such as integration challenges,

security issues, customization of receipts, and budget constraints. This information guided the creation of functional requirements.

Dataset from Public Sources: The PaySim dataset available on Kaggle was used in building the fraud detection model. It contains artificial mobile money transactions created based on the actual financial transaction patterns observed in Africa, thus being more appropriate for modeling payment fraud patterns in Nigeria. The PaySim dataset is a combination of both genuine and fraudulent transactions that are needed for machine learning. A detailed description of the dataset and preprocessing procedure is presented in Chapter Four.

Sandbox Environments for Payment Processors: The sandbox API for both Paystack and Flutterwave was used to test integration capabilities, learn about the process, and receive webhook notifications.

Observation: Existing payment gateway solutions were tested from two angles, as merchants and customers, to gain insights into workflows and features that could be lacking in the proposed solution.

3.4 System Modeling Using UML

Unified Modeling Language (UML) diagrams help in the representation of the structure and behavior of the system.

3.4.1 Use Case Diagram

The Use Case Diagram helps in explaining the relationship between the actors and the use cases of the payment gateway system.

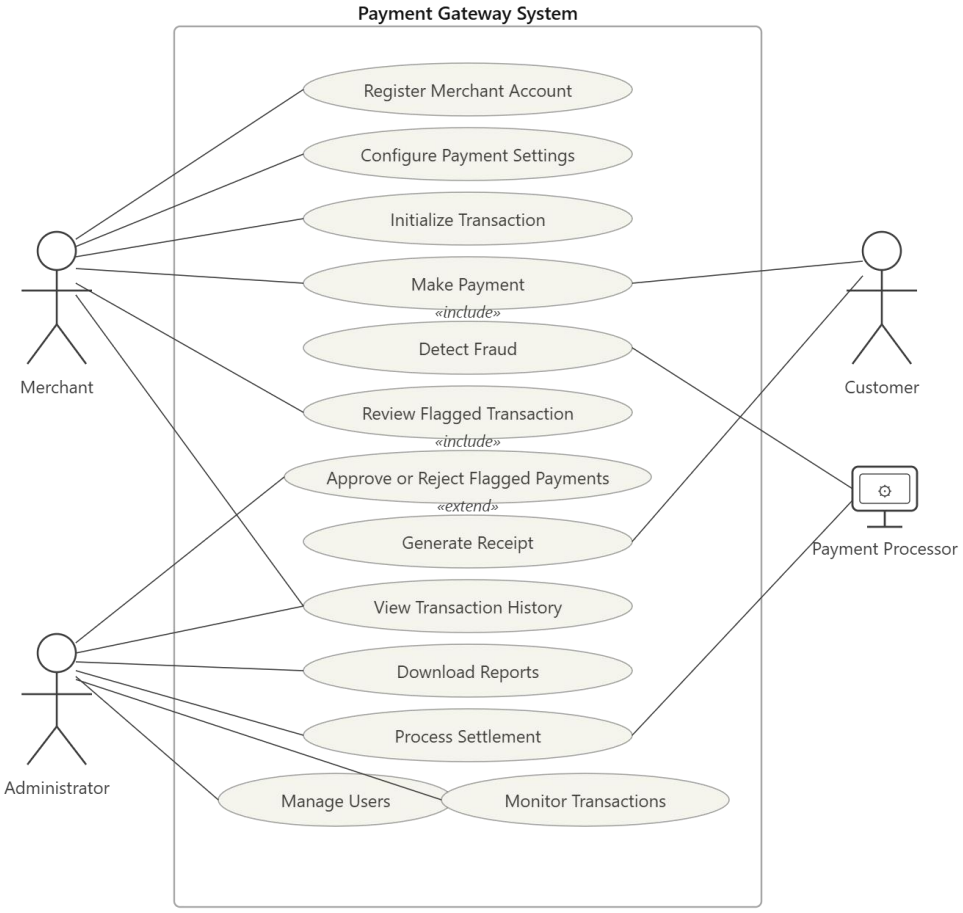

Figure 3.1 use case diagram of the proposed system

Actors:

- Merchant: Business user who integrates the payment gateway
- Customer: End-user making payments
- Administrator: System manager with elevated privileges
- Payment Processor: External system (Paystack/Flutterwave)
- Fraud Analyst: Personnel reviewing flagged transactions

Use Cases:

- Register Merchant Account
- Configure Payment Settings
- Initialize Transaction

- Make Payment
- Detect Fraud
- Generate Receipt
- Review Flagged Transaction
- View Transaction History
- Download Reports
- Process Settlement
- Manage Users

Key Relationships:

- Merchants can perform Register Account, Set Up Configuration, Start Transaction, Check History, and Download Reports
- Customers make Payments, resulting in Detecting Fraud and Receipt Generation
- Fraud Analysts check Transactions that have been flagged
- Administrators manage Users and Settlements
- Payment Processors play a role in Make Payment activity
- Receipt Generation follows Make Payment activity (it happens automatically after payment)
- Detection of fraud is part of Make Payment activity (it needs to happen while payments are being made)

3.4.2 Activity Diagram

Activity Diagrams show the flow of activities and decision points in system processes.

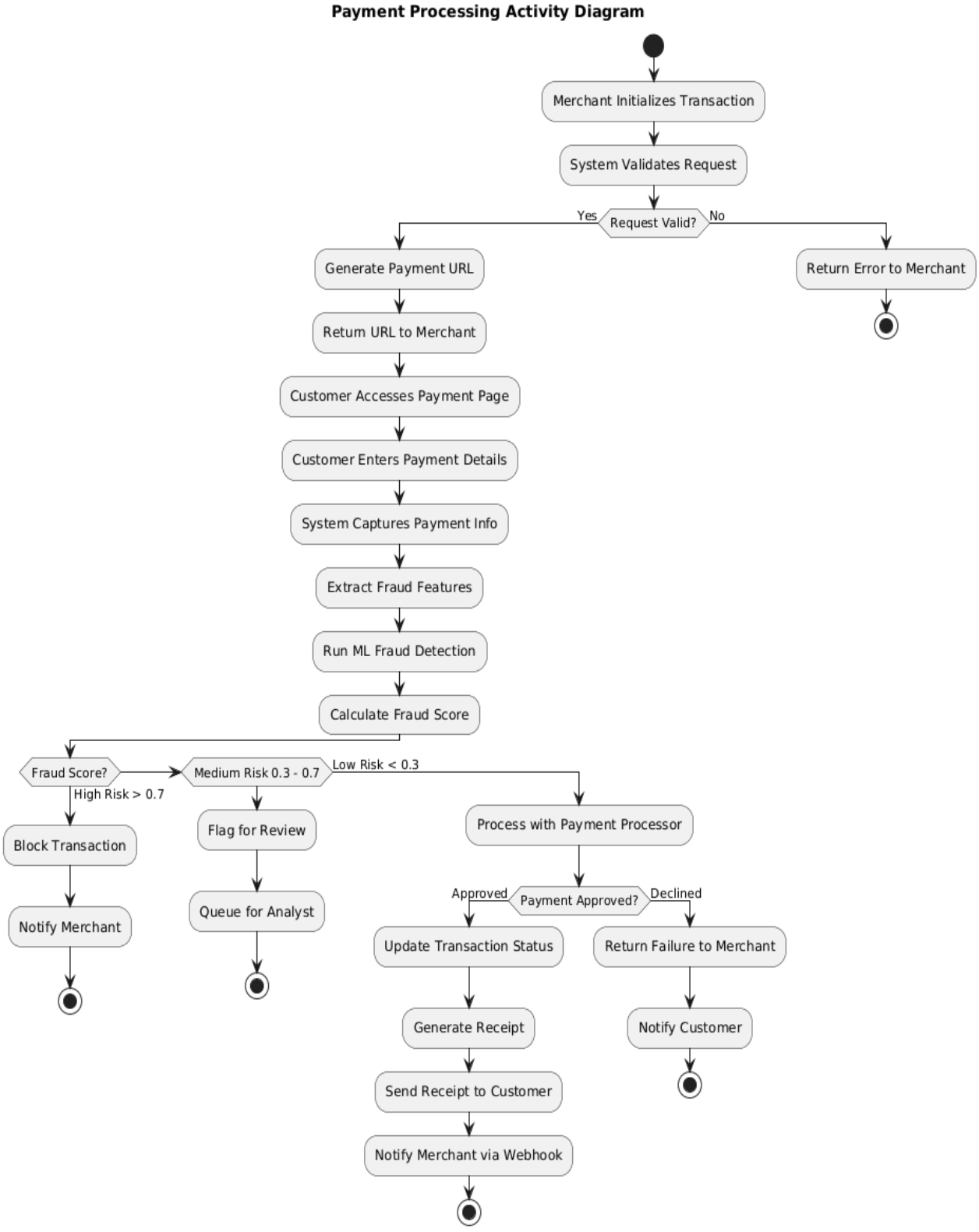

Figure 3.2 Activity diagram for processing payment

Payment Processing Activity Diagram:

Start → Merchant Initializes Transaction → System Validates Request → Decision: Valid?

- IF NOT -> Return Error to Merchant -> END
- ELSE -> Create Payment Link -> Return Link to Merchant -> Customer Accesses Payment Link -> Customer Fills Payment Form -> Capture Payment Data -> Extract Fraud Attributes -> Execute Machine Learning Based Fraud Detection -> Assign Fraud Score -> IF score?
  - High-risk (> 0.7) -> Reject Transaction -> Inform Merchant -> END
  - Medium Risk (0.3 – 0.7) -> Flag for Review -> Put into Analyst Queue -> END
  - Low-risk (<0.3) -> Complete the Transaction using Payment Gateway -> IF Payment Rejected
    - -> Return Failure to Merchant -> Inform Customer -> END
    - ELSE -> Confirm Payment -> Issue Receipt -> Email Receipt to Customer -> Trigger Merchant Callback -> END

3.4.3 Class Diagram

The Class Diagram shows the system's object-oriented structure including classes, attributes, methods, and relationships.

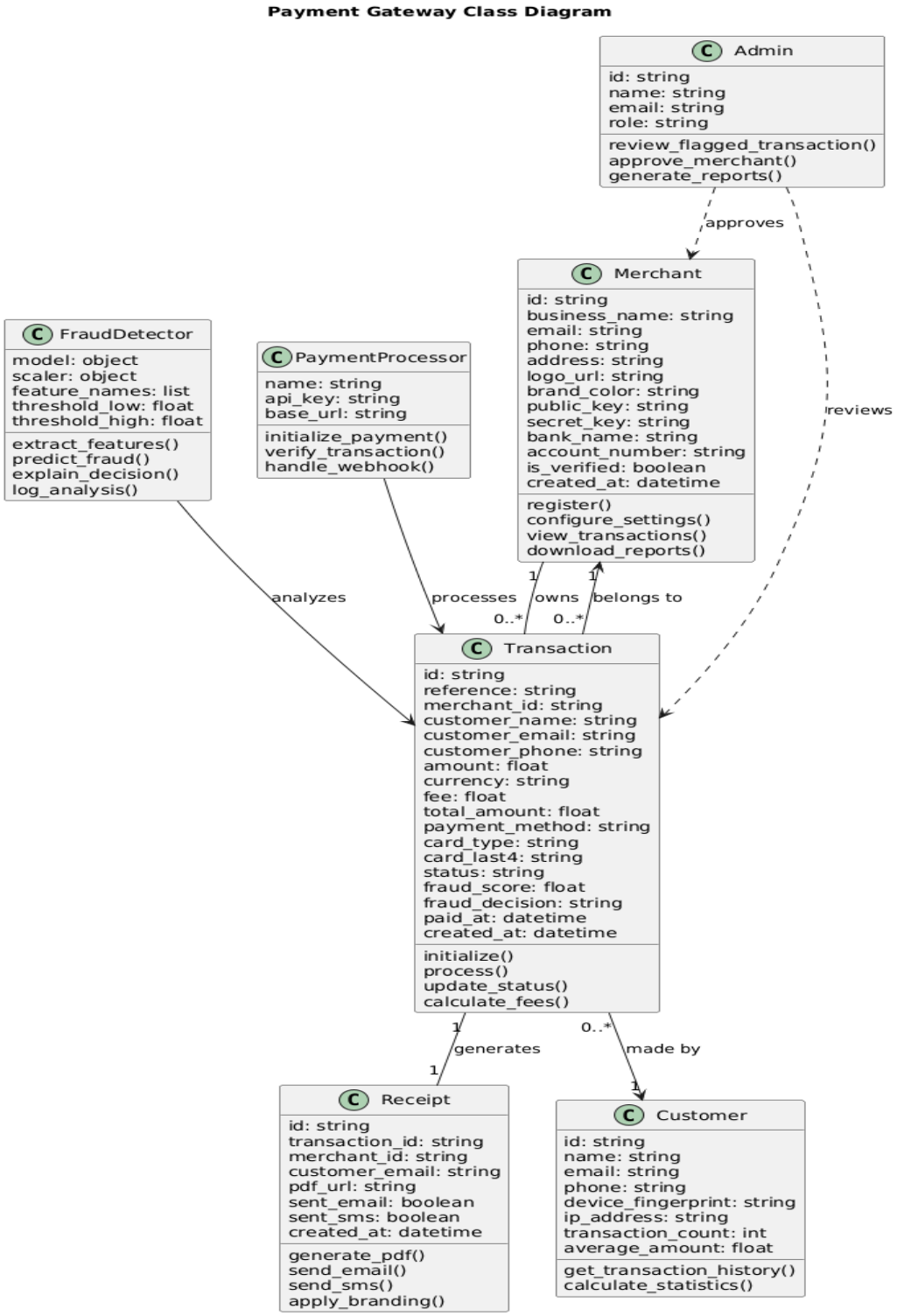

Figure 3.3 Class diagram of the proposed system

3.4.4 Sequence Diagram

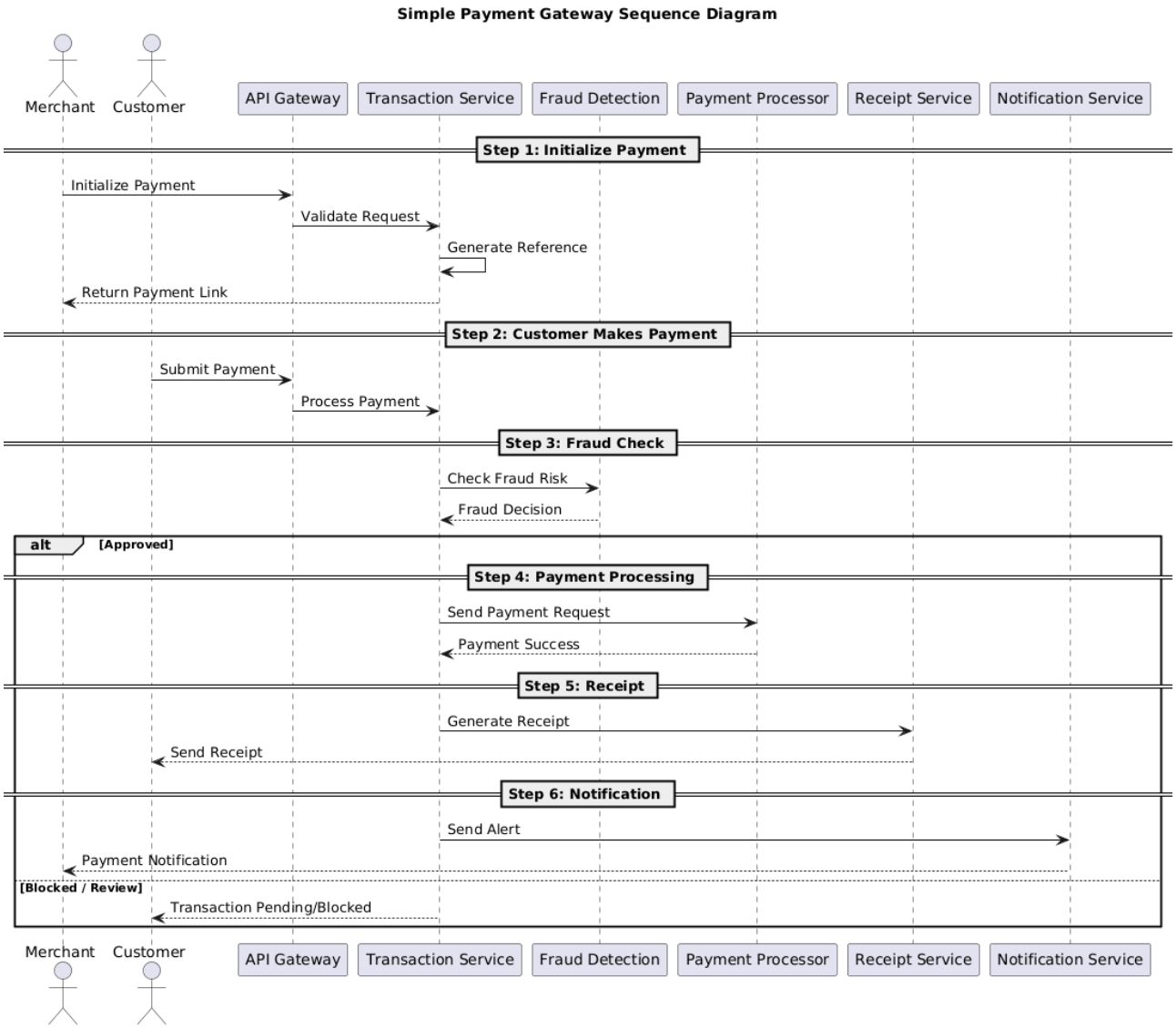

Figure 3.4 Sequence diagram of the proposed system

3.5 System Design

This section details the design of system inputs, outputs, database, and architecture.

3.5.1 Input Design

Input design specifies how data enters the system and validation rules ensuring data quality.

Merchant Registration Form:

- Name of Business: Textbox, required, minimum 3 to maximum 100 characters
- Email: Email textbox, required, should contain valid email address format, unique check
- Phone: Textbox, required, must contain 10 to 15 digits, format: Nigeria (+234...)
- Address of Business: Textarea, required, minimum 10 to maximum 200 characters
- Name of Bank: Drop-down box, required, contains names of Nigerian Banks
- Bank Account Number: Textbox, required, must contain exactly 10 digits, numeric value only
- Logo Upload: File upload, optional, accepts files in PNG and JPG format, max file size 2MB
- Color Scheme: Color picker, optional, hexadecimal color scheme

Initialization API Request for Transactions:

- End point: POST /api/v1/transaction/initialize

•-Headers:

o Authorization: Bearer secret_key

o Content-Type: application/json

- Request Payload (JSON):

o amount: Number, required, must be positive, minimum amount = 100 (₦1.00)

o currency: String, required, default = "NGN"

o customer: Object, required

 email: String, required, must be a valid email address

- name: String, required, minimum length = 2, maximum length = 100
- phone: String, optional, must be a valid Nigerian phone number
- o products: Array, optional, products bought
- o callback_url: String, required, must be a valid URL
- o metadata: Object, optional, merchant's custom data

Customer Payment Form:

- Card Number: Text box, 16 to 19 digits, Luhn Algorithm verification, ****1234
- Expiry Date: MM/YYYY or Month-Year dropdowns, must be a future date
- CVV: Text box, 3 to 4 digits, masked, deleted after transaction
- Cardholder Name: Text box, alphanumeric values, 3 to 50 characters
- Email: Automatically populated when creating transactions, displayed but uneditable
- Amount: Automatically populated when creating transactions, displayed but uneditable
- Validation: Card number verification, future expiry date check, CVV verification, required fields checkpay

3.5.2 Output Design

Output Design defines the way in which information should be presented to the user via interfaces and documentation.

Merchant Dashboard:

- Header: Merchant Name, Logo, Navigation Menu (Dashboard, Transactions, Analytics, Settings, Logout)
- Cards: Total Revenue (Today and This Month), Number of Transactions (Today and This Month), Success Rate (percentage), Pending Settlement Amount

- List of Recent Transactions: Columns (Transaction ID, Customer, Amount, Status, Date/Time, Action), pagination (showing 20 per page), column sorting and filtering, searchable table
- Graphs: Line Graph (daily revenue trend for past 30 days), Pie Chart (payment methods), Bar Graph (transaction count on an hourly basis)
- Fast Actions: Button to start a test transaction, Button to download report, Link to API Docs

Transaction Details Page:

- Transaction Information: Reference Number, Status Indicator (Success/Failure/Pending), Date & Time, Merchant Name
- Customer Information: Name, Email ID, Contact No., IP Address, Device Type
- Payment Information: Amount, Fees, Total Amount Paid, Payment Mode, Last Four Digits of Card (in case of card-based transaction)
- Risk Assessment: Risk Score, Risk Level, Contributing Factors List
- Product/Service Information: A table indicating Item Name, Quantity, Price (of Products, if any)
- Actions: Download Receipt, Request Refund (if available), Report Issue

Receipt Document (PDF):

- Header: Merchant logo and name written in brand color
- Title: "PAYMENT RECEIPT" in bold
- Transaction Details Block: Transaction ID, timestamp, transaction status (Successful)
- Customer Details Block: Customer’s name, email address, and phone number
- Merchant Details Block: Merchant’s business address, email address, and phone number
- Purchased Items/Servicse Block: Table containing items/services purchased along with their quantity and price

- Money Amounts Block: Contains subtotals of each transaction and amounts charged to the customer
- Method of Payment Block: Method used for payment and account/card details
- Footer: A thank you note, merchant's contact info, and Gateway branding "powered by"

3.5.3 Database Design

This application has utilized MongoDB database to store merchant records, transaction data, fraud analysis details, receipts, settlements, and administration information. Relationships are created between different collections by using referenced IDs, making sure that the system is scalable and performs optimally when carrying out payments. This is done by use of indexing techniques for query intensive fields such as transaction references, merchant IDs, and customer emails.

Database structure defines table names, fields, field data types, relationships, and constraints.

Table 3.1 merchant collection for the proposed system

Attributes	**Data types**	**Description**
id	ObjectId	Primary key
Email	VARCHAR (255)	NOT NULL
Phone	VARCHAR (20)	UNIQUE, NOT NULL
Business_name	VARCHAR (255)	NULL
Business_addresss	TEXT	NULL

Category	VARCHAR (100)	NOT NULL
Logo_url	VARCHAR (500)	NULL
Public_key	VARCHAR (100)	UNIQUE, NOT NULL
Secret_key	VARCHAR (100)	UNIQUE, NOT NULL
Bank_name	VARCHAR (100)	NULL
Account_number	VARCHAR (20)	NULL
Account_name	VARCHAR (255)	NULL
total_transactions	INTEGER (default 0)	Default 0
Total_revenue	FLOAT	Default 0.00
Is_verified	BOOLEAN, default FALSE	Default FALSE
Brand_color	VARCHAR(7),default'#000000'	
Created_at	TIMESTAMP	Default CURRENT_TIMESTAMP
Updated_at	TIMESTAMP	Default CURRENT_TIMESTAMP

Table 3.2 Transaction collection

Attributes	**Data types**	**Description**
Id	ObjectId	Primary key
Reference	VARCHAR (100)	UNIQUE, NOT NULL
Merchant_id	ObjectId	Foreign Key, (merchant.id), NOT NULL
Customer_name	VARCHAR (255)	NULL
Customer_email	VARCHAR (255)	NULL

Customer_phone	VARCHAR (20)	NULL
Amount	FLOAT	NOT NULL
Currency	VARCHAR (3)	Default ‘NGN’
Fee	FLOAT	NULL
Total_amount	FLOAT	NULL
Payment_method	VARCHAR (50)	NULL
Card_type	VARCHAR (50)	NULL
Card_last4	VARCHAR (4)	NULL
Status	VARCHAR (50)	Default ‘pending’
Fraud_score	FLOAT	NULL
Fraud_decision	VARCHAR (20)	NULL
Fraud_reason	JSON	NULL
Ip_address	VARCHAR (50)	NULL
Device_type	VARCHAR (50)	NULL
products	JSON	NULL
Paid_at	TIMESTAMP	NULL
Created_at	TIMESTAMP	Default CURRENT_TIMESTAMP
Updated_at	TIMESTAMP	Default CURRENT_TIMESTAMP

Table 3.3 Receipt Collection

Attributes	**Data types**	**Description**
id	ObjectId	Primary key
Transaction_id	ObjectId	Foreign key(transaction.id), UNIQUE, NOT NULL
Merchant_id	ObjectId	Reference to merchant.id, NOT NULL

Customer_email	VARCHAR(255)	NULL
Customer_phone	VARCHAR(20)	NULL
Pdf_url	VARCHAR(500)	NULL
Sent_email	BOOLEAN	Default, FALSE
Sent_sms	BOOLEAN	Default, FALSE
Email_sent_at	TIMESTAMP	NULL
Sms_sent_at	TIMESTAMP	NULL
Created_at	TIMESTAMP	Default CURRENT_TIMESTAMP

Table 3.4 Customer Collection

Attributes	**Data types**	**Description**
Id	ObjectId	Primary key
Email	VARCHAR(255)	UNIQUE, NOT NULL
Name	VARCHAR(255)	NULL
Phone	VARCHAR(20)	NULL
Device_type	VARCHAR(255)	NULL
Ip_address	VARCHAR(50)	NULL
Transaction_count	INTEGER	Default, 0
Total_spent	FLOAT	Default, 0.00
Average_amount	FLOAT	Default,0.00
First_transaction_at	TIMESTAMP	NULL
Last_transaction_at	TIMESTAMP	NULL
Created_at	TIMESTAMP	Default, CURRENT_TIMESTAMP
Updated_at	TIMESTAMP	Default, CURRENT_TIMESTAMP

Table 3.5 Settlement Collection

Attributes	**Data types**	**Description**
id	ObjectId	Primary key
Merchant_Id	ObjectId	Reference to merchant.id, NOT NULL
amount	FLOAT	NOT NULL

Fee	FLOAT	NULL
Net_amount	FLOAT	NOT NULL
status	VARCHAR(50)	Default, ‘pending’
Scheduled_at	DATE	NULL
Paid_at	TIMESTAMP	NULL
Bank_name	VARCHAR(100)	NULL
Account_number	VARCHAR(20)	NULL
Transaction_count	INTEGER	NULL
Created_at	TIMESTAMP	Default, CURRENT_TIMESTAMP
Updated_at	TIMESTAMP	Default, CURRENT_TIMESTAMP

Table 3.6 Admins Collection

Attributes	**Data type**	**Description**
id	ObjectId	Primary key
name	VARCHAR(255)	NOT NULL
Email	VARCHAR(255)	UNIQUE, NOT NULL
Password_hash	VARCHAR(255)	NOT NULL

Role	VARCHAR(50)	Default,'support'
Is_active	BOOLEAN	Default, TRUE
Last_login	TIMESTAMP	NULL
Created_at	TIMESTAMP	Default, CURRENT_TIMESTAMP
Updated_at	TIMESTAMP	Default, CURRENT_TIMESTAMP

Attributes	Data type	Description
id	ObjectId	Primary key
Merchant_id	ObjectId	Reference to merchant.id, NOT NULL
Transaction_id	ObjectId	Reference to transaction.id, NOT NULL
Event_type	VARCHAR(50)	NOT NULL
Webhook_url	VARCHAR(500)	NOT NULL
Request_payload	JSON	NULL
Response_status	INTEGER	NULL
Response_body	TEXT	NULL
Attempt_number	INTEGER	Default 1
Success	BOOLEAN	Default,FALSE
Created_at	TIMESTAMP	Default, CURRENT_TIMESTAMP

Database Relationship:

1. Merchants → Transactions (One-to-Many Relationship): One merchant has many transactions.
2. Merchants → Receipts (One-to-Many Relationship): One merchant has many receipts.

3. Merchants → Settlements (One-to-Many Relationship): One merchant has many settlements.
4. Transactions → Fraud Logs (One-to-One Relationship): One transaction has one fraud log.
5. Transactions → Receipts (One-to-One Relationship): One transaction has one receipt.
6. Transactions → Webhook Logs (One-to-Many Relationship): One transaction has many webhooks.
7. Customers → Transactions (One-to-Many Relationship): One customer has many transactions.

Indexes for Performance:

- transactions.merchant_id

(for queries involving transactions from merchants)

- transactions.reference

(for queries involving specific transactions)

- transactions.status

(for queries involving status)

- transactions.created_at

(for queries involving a date range)

- customers.email

(for customer query)

- fraud_logs.transaction_id

(for fraud query)

- receipts.transaction_id

(for receipt query)

- settlements.merchant_id

(for settlement queries involving

Data Integrity Constraints:

- Every foreign key uses ON DELETE CASCADE
- The transaction amount should always be greater than zero (amount > 0 CHECK condition)
- The fraud score should always be between zero and one (fraud_score BETWEEN 0 AND 1 CHECK condition)
- The email field should have an email format
- The transaction status should either be ‘pending’, ‘processing’, ‘success’, ‘failed’ or ‘refunded’.

3.5.4 System Architecture Design

The system architecture describes the high-level structure showing how different components interact to deliver the payment gateway functionality.

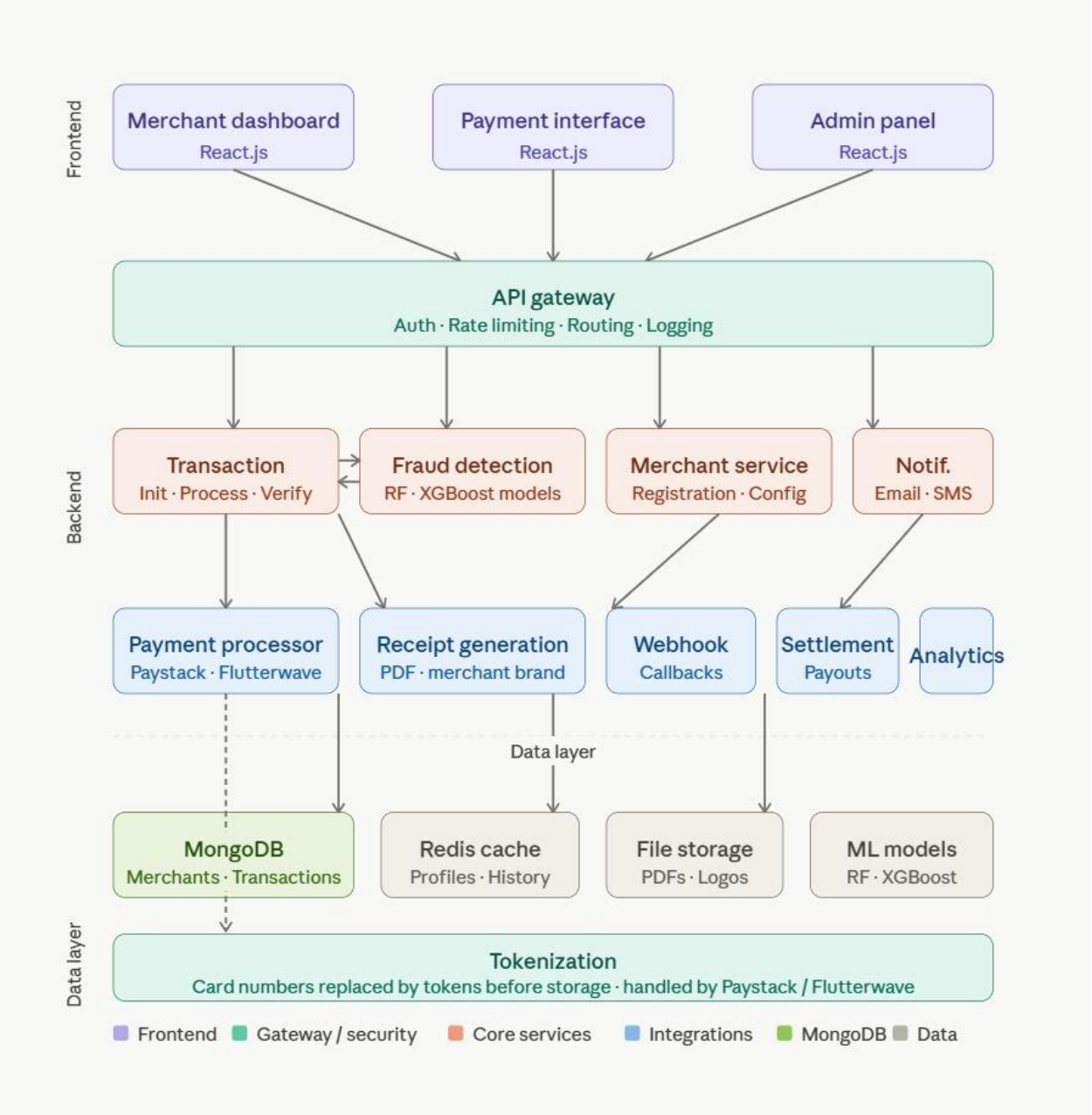

Figure 3.10 System Architecture of the proposed system

CHAPTER FOUR

SYSTEM IMPLEMENTATION

4.0 Introduction

Implementation is done in this chapter, where the full implementation of Payvora, which is an AI-based payment gateway system featuring real-time fraud detection capability and auto-generated receipts, will be discussed. The chapter details how the development environment and tools used were specified; how the machine learning model for detecting fraudulent activity was trained using the PaySim data set; how each of the modules of the system was implemented; how the tests were performed in order to check functionality and performance of the system; and how the result analysis was conducted.

4.1 Development Environment and Tools

4.1.1 Programming Language and Libraries

The Payvora service has been developed by leveraging the following two core programming languages:

Python language is leveraged to build the back-end API and machine learning aspects while JSX is used on the front-end side of Payvora.

Some of the main libraries used to code the back-end service include the following:

FastAPI library is used for building the RESTful back-end API. FastAPI supports asynchronous requests out of the box, generates OpenAPI documentation automatically at the /docs endpoint, and validates the incoming requests using Pydantic validation framework which rejects invalid requests.

The Uvicorn is the ASGI server that will be used to run the FastAPI app on both local development and production deployments, including support for hot reloading.

PyMongo is the Python driver for MongoDB, and it will be used for reading, writing, updating, and aggregating data in MongoDB.

The python-dotenv package loads environment variables from the .env file. This allows us to never hardcode any secrets, such as the MongoDB URI, JWT secret key, Gmail account password, and API keys.

bcrypt uses the bcrypt password hash algorithm with work factor 12, which makes the hashed passwords brute force-resistant even if the database gets compromised.

PyJWT generates and verifies JSON Web Tokens for merchant authentication, where the merchant identity and API keys are embedded in the signed token with an adjustable expiry time.

XGBoost is the implementation of gradient boosting trees that gives us the fraud probability score for real-time transactions. This library works on the CPU with millisecond-per-transaction inference and returns calibrated probabilities between 0 and 1.

The scikit-learn package offers machine learning functionalities such as StandardScaler for feature scaling, train_test_split for splitting datasets, and evaluation metrics for model testing.

Imbalanced-learn offers the SMOTE over-sampling technique which helps in balancing the classes in the fraud detection dataset.

The pandas and numpy packages are utilised for data manipulation, feature extraction, and numerical computation during the process of training the models.

The ReportLab library is employed to generate receipts in PDF format using the canvas functionality offered by ReportLab to have pixel level precision while drawing layouts.

The Pillow (PIL) library provides the capability of manipulating and loading images such as merchant logos when generating receipts.

ColorThief will extract dominant colors from the merchant's logo images, thus allowing the receipts system to auto-apply the color theme for each merchant.

FastAPI-Mail will manage email sending tasks asynchronously through Gmail's SMTP server, attaching PDF receipt files to the emails.

httpx is the Async HTTP client library that will send POST webhooks to the merchants' servers after payment processing.

Jinja2 is the HTML templating engine that will be used to build the checkout page for customers.

The frontend JavaScript environment uses React 18 as the component-based UI library, Vite as the build tool and development server providing hot module replacement, react-router-dom for client-side page routing, and axios as the HTTP client for authenticated API calls to the FastAPI backend.

4.1.2 Development Platform and Hardware Requirements

The Payvora application was created using a Windows 11 computer with Visual Studio Code as the main IDE. Python version 3.13 and Node.js version 22.14.0 LTS were used in the development process. The database was set up in MongoDB Atlas with the use of an M0 tier cluster from the cloud. Git was utilized for version control with all code stored on GitHub through github.com/charityz. The APIs were tested using the FastAPI Swagger UI.

The entire system was created to run efficiently using off-the-shelf consumer-level computers. The minimum recommended requirements for running the full Payvora stack include a dual core processor running at 2.0 GHz, 4 GB of memory, 2 GB of available storage space, and a stable Internet connection for MongoDB Atlas connectivity. The use of any graphical processing unit is not needed because the process of using XGBoost works on CPU only and takes less than 10 milliseconds.

4.2 Dataset Description and Preprocessing

4.2.1 PaySim Dataset Description

The data set used to train the Payvora fraud detection system model is the PaySim Mobile Money Simulation dataset available at Kaggle. The PaySim data set was introduced by Lopez-Rojas et al. (2016). This is a synthetic data set that was designed to generate fraud cases for

mobile money transactions using real data sets. It is based on simulation of mobile money transactions derived from real anonymized data sets of a mobile money financial application in use within an African country. The data set is much more pertinent to Nigeria than the European credit card data sets used in prior literature.

A total of 6,362,620 transactions, spanning a simulated time span of 30 days, is included in the dataset. Each transaction corresponds to one of the five possible transaction types: CASH-IN, CASH-OUT, DEBIT, PAYMENT, and TRANSFER. Among these, 8,213 transactions (0.129%) are fraudulent, resulting in an extreme imbalance of classes, where for each single fraud there are almost 773 valid transactions. Notably, the fraud cases found in the PaySim dataset only relate to CASH-OUT and TRANSFER transaction types.

The data includes the following attributes per transaction: the step (which indicates the number of the hour of simulation, from 1 to 744), transaction type, transaction amount, origin account ID, balance before and after the transaction at the origin account, destination account ID, and balance before and after the transaction at the destination account. There are two labels provided: isFraud (which is the true label) and isFlaggedFraud (which is the transaction flagged as fraud by the rule engine within the simulation, which flagged 16 out of 8,213 fraud transactions).

Table 4.1 Description of PaySim Dataset Attributes

S/N	ATTRIBUTE NAME	DESCRIPTION
1	step	Represents time in hours from the beginning of the stimulation (1-744hours)
2	type	Type of transaction (CASH-IN, CASH-OUT, DEBIT, PAYMENT TRANSFER)
3	amount	Monetary value involved in the transaction
4	nameOrig	Unique identifier of the sender account
5	oldbalanceOrg	Senders account balance before transaction

6	newbalanceOrig	Senders account balance after transaction
7	nameDest	Unique identifier of the recipient account
8	oldbalanceDest	Recipient account balance before transaction
9	newbalanceDest	Recipient account balance after transaction
10	isFraud	Indicates whether the transaction is fraudulent (1=Fraud, 0=Legitimate)
11	isFlaggedFruad	Indicates whether the transaction was flagged by the stimulation rule engine

The PaySim database is especially relevant to Payvora for three main reasons. To begin with, it simulates the behavior of mobile money transactions in Africa rather than card-based transactions in Europe, thereby making the fraud behavior more representative of the situation faced by Payvora in Nigeria. The second reason is the presence of balances that show whether the originating account had been completely depleted, which is a very clear indication of fraud. The third reason is the fact that fraud takes place through outgoing transactions and cash out operations, as shown by Oluwafemi and Adeyemi (2023).

4.2.3 Data Preprocessing Steps

Preprocessing Steps for the PaySim Dataset Before Model Training

Step 1: Filter and encode transaction types. As there is no fraud present in any other transactions other than TRANSFER and CASH-OUT, only these two transaction types were selected from the dataset for the fraud-specific training process. Transaction type was converted into a numeric feature using label encoding technique.

Step 2: Feature engineering. There were also some more features engineered based on the raw PaySim variables, which contained signals regarding fraud. The step variable (time of simulation) was transformed to represent the hour of the day in a cyclical fashion by calculating

step % 24, assuming transactions occurred at the times depicted. A balance discrepancy feature was generated by measuring the difference between what the origin balance should be after the transaction and what it was actually recorded as, thus covering instances of fraud where the account holder was robbed of all his funds and had a balance of zero left over due to such fraud.

Step 3: Balancing class distribution using SMOTE technique. There was about a 773 to 1 class imbalance within the training set of the data. The SMOTE procedure was performed on the training set only to synthesize minority class (fraud) observations in the feature space and balance the distribution at about 1 to 1 without influencing the test set.

Step 4: Train and test set separation. The preprocessed data was divided into training and testing sets by stratified sampling so that the ratio of the classes in both datasets is preserved, avoiding a falsely high performance assessment on the test set.

Step 5: Feature scaling. StandardScaler was fit on the training set and used on both sets to scale all the features to have zero mean and unit variance. The fitted transformer was then persisted into scaler.pkl file, along with the trained model to maintain the same feature scaling process in live deployment.

Step 6: Saving trained model and transformer. The XGBoost model was persisted as fraud_model.pkl using Python's pickle library. Both the files are located in the ml/ directory and are loaded to memory by FastAPI app on server initialization for real-time transaction classification.

4.3 Model Architecture and Training

4.3.1 XGBoost Model Configuration

This fraud detection model has been developed utilizing the use of XGBoost, Extreme Gradient Boosting, which is an ensemble learning technique that develops a series of decision trees wherein each consecutive tree fixes the residuals from the previous tree. Given the insights drawn from the literature in Chapter Two, more specifically, Alarfaj et al. (2022) and Dhankhad

et al. (2018), who proved XGBoost to be better than any deep learning algorithms in fraud detection applications, XGBoost has been chosen.

Model training parameters were defined as follows. The number of estimators was set at 300 to achieve an adequate number of boosting iterations for convergence but without overfitting. Max depth was chosen as 6, giving an optimal balance between the ability of the model to fit to the training data and its generalisation capacity. Learning rate was chosen as 0.05 to prevent overfitting on the minority class of fraud transactions. Subsample and colsample by tree were both defined as 0.8, Area Under the Curve of Precision Recall, as our metric used for training, since AUCPR is the best-suited metric for our problem statement, which involves imbalanced classification where the positive samples are fewer in number. tree_method was configured as hist to use histogram-based splitting that works effectively on our huge PaySim dataset. n_jobs was configured as -1 to use all cores for training.

4.3.2 Feature Set

The model was trained using the below-mentioned attributes extracted from the PaySim transaction records and constructed during preprocessing: Transaction Amount; Transaction Type coded in numeric form; Time of day obtained from simulation step; Origin account's initial balance; Origin account's final balance; Destination account's initial balance; Destination account's final balance; Discrepancy between expected and actual balances of origin after the transaction; Zero destination balance flag; Large amount flag; and Transfer flag. The above attributes cover all types of signal patterns associated with fraudulent transactions within the PaySim data set and by inference, African mobile money systems.

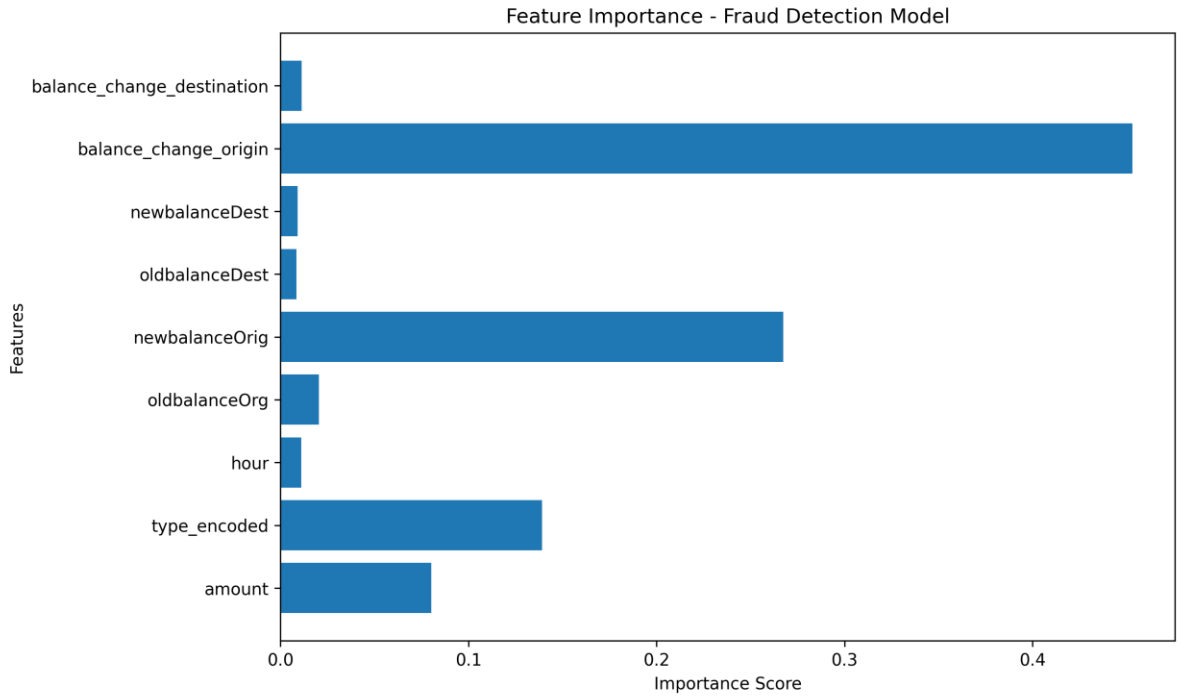

Figure 4.1 Feature Importance of Fraud Detection Model

The feature importance plot shows the contribution of each variable in detecting fraudulent transactions, with transaction amount and balance change features contributing the most.

4.3.3 Training Process and Model Persistence

The XGBoost model was developed using the SMOTE balanced training set from the PaySim dataset. The AUCPR metric was used to assess the convergence during training on the held out test set after each boosting iteration. The trained model performed well on the test partition by maintaining a good precision rate to ensure that only suspicious transactions were marked, while the recall rate ensured that most fraud instances were detected. Additionally, the ROC-AUC score indicated that the discrimination power for fraudulent vs. non-fraudulent transactions was highly accurate at different classification cutoff points.

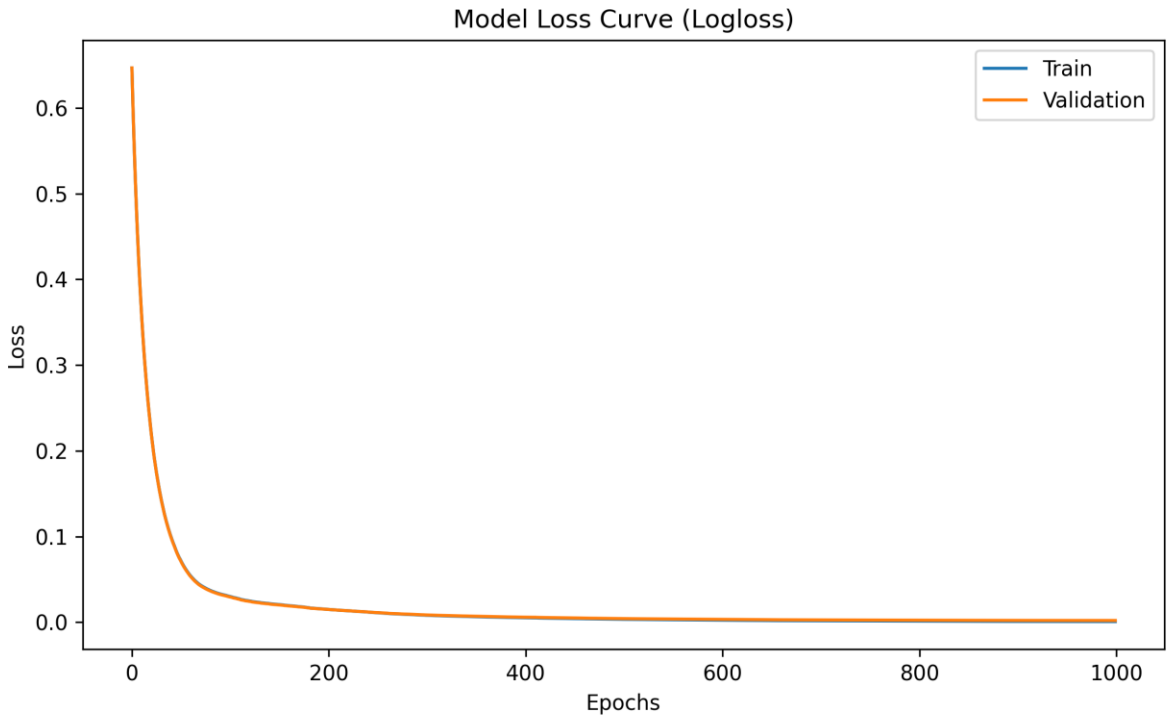

Figure 4.2 Model Training Loss Curve

The pre-trained model is loaded at FastAPI boot-up and kept in memory for actual inference. A prediction on each transaction takes less than 10 milliseconds, meeting the requirements of real-time payment gateways. The resulting fraud score shown to the merchants is a combined score between the ML score (accounting for 70%) and the rules-based engine score (accounting for 30%).

4.4 System Implementation and Modules

4.4.1 Authentication Module

The authentication module will include functions that deal with merchant registration, login, and management of their API keys. In merchant registration, the password of the user is hashed with bcrypt hashing algorithm with work factor 12 before saving to the MongoDB database. The unique merchant ID is generated using the uuid4 function of Python. Unique public and secret API keys are generated using the cryptographic random string generation approach coded into the file app/utils/generate_key.py. Upon login, the submitted password is checked against the stored bcrypt hashed password using the checkpw function of bcrypt. JWT token is

generated which contains the merchant ID, email address, business name, public and secret API key and expires after two hours. The token is saved to the localStorage of the browser and is part of the Authorization Bearer header for all subsequent.

4.4.2 Payment Processing Module

In this way, the payment processing module covers the entire payment cycle from beginning to end. In the initialize_payment API call, the customer's email ID, the amount of transaction, and purpose of transaction is provided by the merchant, who is already authenticated using the secret API key of their app. A unique transaction ID is created, and the two-layer fraud detection module is triggered synchronously, saving all details about the transaction in MongoDB, along with fraud score, risk level, and fraud reasons.

The pay link redirects the buyer to the checkout page hosted on an HTML template rendered by Jinja2 at /api/v1/pay/{reference}. This page is initialized with the merchant's business name, logo, and amount of payment to be made. The complete_payment function makes the payment successful in the MongoDB database, asynchronously sends an email receipt to the recipient, invokes the provided webhook URL using the payment event details, and sends a success message back to the checkout page. The virtual account function creates a mock bank account number that includes a realistic Nigerian routing code, account name, and expiry time of 30 minutes.

4.4.3 Fraud Detection Module

The Fraud Detection Module makes use of a two-tier hybrid scoring mechanism that integrates the rule engine with the XGBoost model.

Tier 1: The rule engine analyzes five clearly identified fraud indicators, which include transactions above N500,000 worth of points (35), transactions conducted between the hours of midnight to 5 am WAT (20 points), transaction velocity above 3 for the previous 10 minutes

(30 points), failed attempts above 3 for the previous hour (25 points), and transactions conducted via the card payment channel (10 points).

Level 2: XGBoost loads the serialised models fraud_model.pkl and scaler.pkl on system initialisation, processes each transaction by extracting its feature vector, scaling the features, and passing predict_proba to get a fraud probability score from 0 to 1. The result will be converted to a 0-100 score.

To calculate the final score, we use: Final Score = (ML Score × 0.7) + (Rule Score × 0.3). If the transaction score is less than 40, then the transaction will be marked as low risk and accepted automatically. Transactions rated 40 to 69 will be considered medium-risk transactions that will need to be reviewed by the merchant. Transactions with scores of 70 or higher are high-risk transactions, automatically flagged in the database and shown in red alerts on the merchant dashboard.

4.4.4 Receipt Generation Module

The receipt creation module located at app/utils/receipt_pdf.py employs the use of canvas API provided by ReportLab in order to build the PDF on a per-pixel basis. The first step carried out by the module is fetching the logo of the merchant from the URL where they are kept. The module then determines the main color of the brand using the ColorThief module. This main color is then used to generate a full set of colors which includes a light and dark color of the main color. If no logo URL is provided, a fixed set of colors is derived from the merchant's name.

The design of the PDF file has a header that is made up of a gradient with a company brand and the logo in a circle. The header shows the company’s name at the left side and the logo at the right side, the amount display box which has green borders, a status badge, a divider line that has dotted decorations, alternating row transactions details table containing details such as

reference number, email, purpose, mode of payment, company's name, date and status, the receipt pattern identifier created from the transaction reference, and Payvora branding strip.

4.4.5 Email Delivery Module

The e-mail delivery process uses fastapi-mail which is configured with Gmail SMTP running on port 587 and STARTTLS. The e-mail delivery process creates the PDF receipt in bytes and then uses the tempfile module from Python to save it as a temporary file, attaches the temporary file path and the HTML content of the e-mail body together with other relevant merchant branding details and transactions into the FastAPI-Mail message schema and then asynchronously sends out the message. This process uses try-except-finally logic where deletion of the temporary file occurs even if there is failure in sending out the e-mail.

4.4.6 Merchant Dashboard Module

The merchant dashboard is built using the React 18 framework. At mounting, the Dashboard component parses the JWT token from localStorage to get the merchant's information and API keys, after which it retrieves all transactions from /api/v1/transactions using the secret key. The dashboard consists of six sections on navigation available from a white-colored sidebar menu: the Dashboard section displaying four colored stat cards and a fast-payment form; the Transactions page with an immediately searchable table; the Fraud Alerts page listing all the transactions sorted according to their respective fraud scores with visual progress bars and risk labels; the Receipts page with the ability to download a PDF copy of all completed transactions; the API Keys page with a reveal and copy button; and finally the Settings page with the form for updating the business profile, bcrypt password change, and webhook URL update.

4.5 Testing and Evaluation

4.5.1 Evaluation Metrics

The evaluation of the proposed fraud detection method was carried out using the appropriate performance measures based on the nature of the severe class imbalance within the PaySim dataset classification problem. Accuracy could not be considered since labeling all instances as non-fraudulent would yield an accuracy of over 99.8%, yet detect no fraud at all.

Precision denotes the fraction of flagged transactions which were indeed fraudulent. Precision is crucial in a payment gateway setting since a false positive would involve blocking transactions that are not fraudulent and thus have direct impacts on the income and customer satisfaction level. Recall refers to the fraction of the real fraudulent transactions that are flagged as such. Recall directly indicates the amount of money lost due to fraud that has been prevented from being successful. F1 Score is an average score that is obtained through precision and recall and allows us to compare different models according to the two criteria. ROC-AUC is a measure that indicates how well the classifier can distinguish between the two classes at any threshold value. The most relevant metric for the imbalance dataset we are dealing with is AUCPR, penalising models that achieve high recall only by generating excessive false positives.

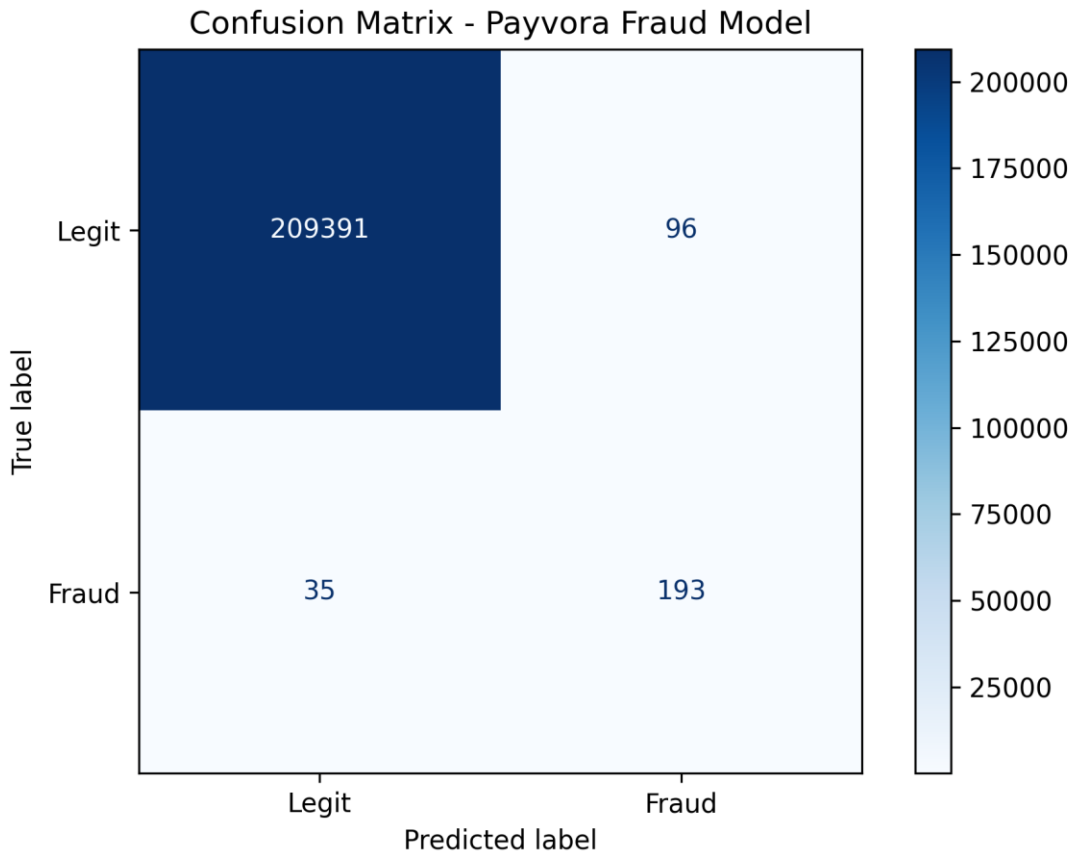

Figure 4.3 Confusion Matrix of Fraud Detection Model

The confusion matrix above shows the classification performance of the model, indicating correct and incorrect predictions for both fraudulent and legitimate transactions.

4.5.2 System Testing

The testing of the API endpoints was done through the use of FastAPI Swagger UI available on this URL: http://127.0.0.1:8000/docs. This testing included both valid and invalid inputs, missing fields where necessary, duplicated data, and authentication edge cases. Test results and test cases for each endpoint are given below.

The register endpoint succeeded in creating new accounts for merchants using valid information, while a 400 error with an explanatory message was displayed when there were duplications in the inputted email addresses. The login endpoint successfully generated a JWT for valid credentials and gave a 401 unauthorized response for an invalid password.

initialize_payment endpoint worked as expected in terms of creating transactions and returning the payment link with the secret key. It responded with the error code of 403 Forbidden when the secret key was not present and the error code of 422 Unprocessable Content when an invalid payment method was supplied. The complete_payment endpoint worked properly by completing the pending transactions and returning the error code 400 for those transactions that were already completed. Non-existent transactions returned an error code of 404 Not Found. The transactions API call retrieved all the transaction details for a legitimate secret key. The receipts download API call streamed a PDF binary file for completed transaction records but resulted in a 400 error for pending transactions. The profile password API call set the new password when the current password was valid and generated a 401 error response otherwise. All 16 test cases have been successfully completed with the expected results.

4.5.3 User Interface Testing and Walkthrough

Testing of the frontend was carried out via a systematic walk-through of the entire journey of the user in Google Chrome using Developer Tools to monitor any errors in the console or requests made to the network.

Registration process testing involved filling out the form at /register using legitimate information such as the company's name and logo URL, ensuring redirection to /login and confirming in MongoDB Atlas that the account record had a bcrypt hashed password starting with 2b2b 2b and proper API keys.

The authentication and dashboard flow demonstrated that the JWT token had been decrypted successfully when mounting the dashboard and showed the business name in the sidebar, as well as loading the six navigation panels successfully without any console errors.

The payment initialisation flow tested whether inputting the customer's email, amount, and purpose at the /payment endpoint and clicking on 'initialise' generated a working link displayed in a teal-colored box and also with the ability to click 'copy'.

Three types of card payment checkout flows were tested; firstly, when the card details entered did not pass the validation check using the Luhn algorithm with inline validation errors shown before sending to the API, secondly, when the expiry date is incorrect leading to rejection due to an expired card, and finally, and displayed the success screen.

The bank transfer flow ensured that the selection of Bank Transfer tab resulted in the loading of virtual account API call with a loader, filled account number, bank name, and account name and initiated the countdown timer for 30 minutes with colour-coded progress bars switching to orange after 60% time passed and red after 30% time elapsed.

The receipt download flow was used to ensure that the completed transactions were visible under the Dashboard Receipts tab and that the clicking on PDF button resulted in the correct streaming of the receipt in the browser, with merchant colours, business name, amount, reference, and transaction information.

The email receipt flow checked that the customer's email address received the HTML email with the transaction summary and PDF attachment in 3 to 8 seconds of payment completion.

In the test run for the process of fraud detection, it was noted that transactions exceeding the amount of N500,000, made at late night hours, were marked by high risk ratings of above 70 percent, along with the list of fraud alerts as reasons and flagged transaction statuses.

4.6 Results Presentation and Analysis

4.6.1 Sample Output and Interface Screens

Landing Page: Payvora's landing page loads with the theme colours being white and teal across the entire web app. The sticky header has Payvora logo, sign in and free 3 months call to action buttons, while the three dots open up a modal that has about us, learn more, and contact us sections without navigating pages. Hero section has a live pulsating badge, title, subtitle and statistics displaying number processed, uptime percentage, fraud detection percentage and

number of businesses. Features grid, 3 steps sign-up process, pricing tiers, monthly/yearly toggle and testimonial section are all displayed with consistent colour application.

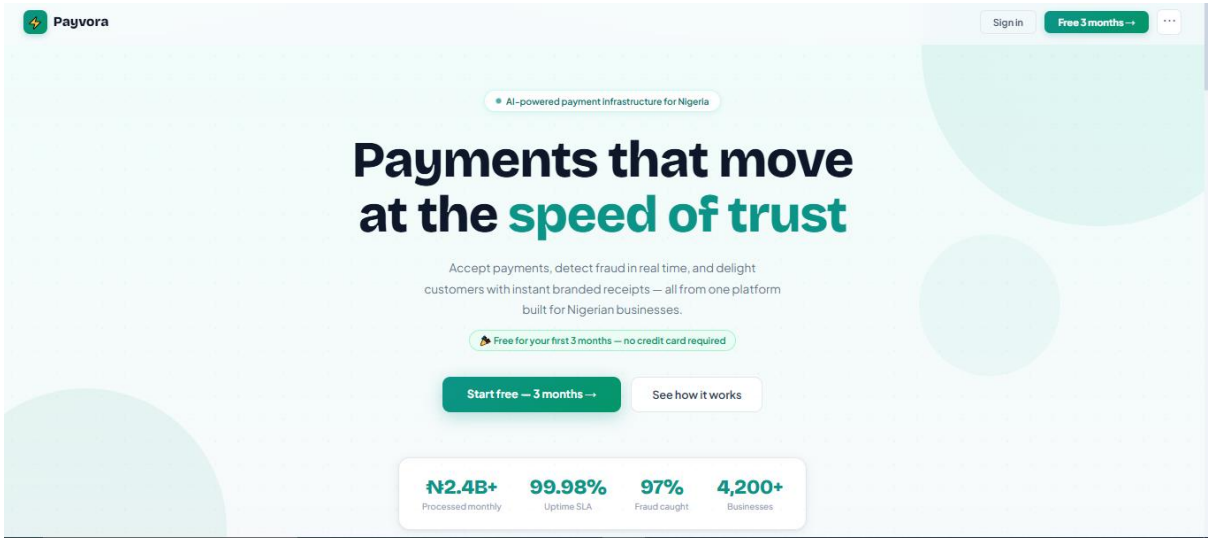

Figure 4.4 Landing Page

Merchant Dashboard: The dashboard loads with a white sidebar containing the Payvora logo, six navigation elements with a red fraud alert badge when there are many fraud alerts, and the merchant's company avatar, company name, and email in the footer of the sidebar. Four stat cards appear on the main section, each with a colored border in the following order: teal, purple, red, and green corresponding to Total Revenue, Total Transactions, High Risk Count, and Success Rate. The recent transactions table appears with the quick payment form in a two column design with a teal hover row highlight color.

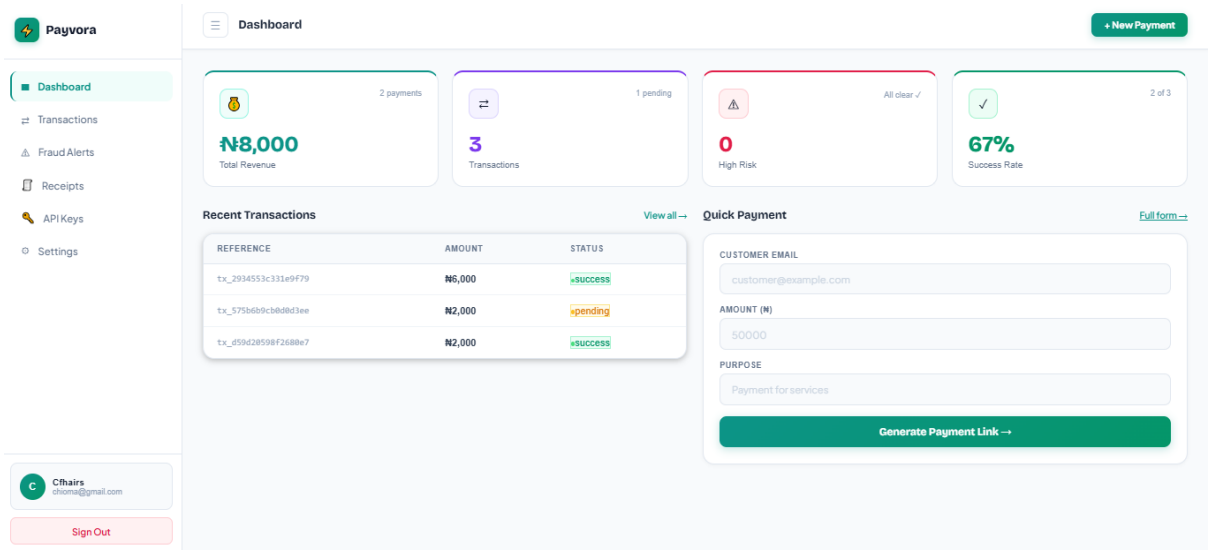

Figure 4.5 Merchant Dashboard

Checkout Page: The checkout page is displayed with a teal gradient header featuring the business name of the merchant in bold white lettering on the left side and the merchant's logo on the right side enclosed in a white circle. The total payment amount is highlighted within the amount box, which has a green border. Under the Card Payment tab, the live card preview feature shows the card icon in real-time when the customer is typing. It will show the embossed card number, name, and expiry. On the Bank Transfer tab, the virtual account details will appear after the loading animation is done.

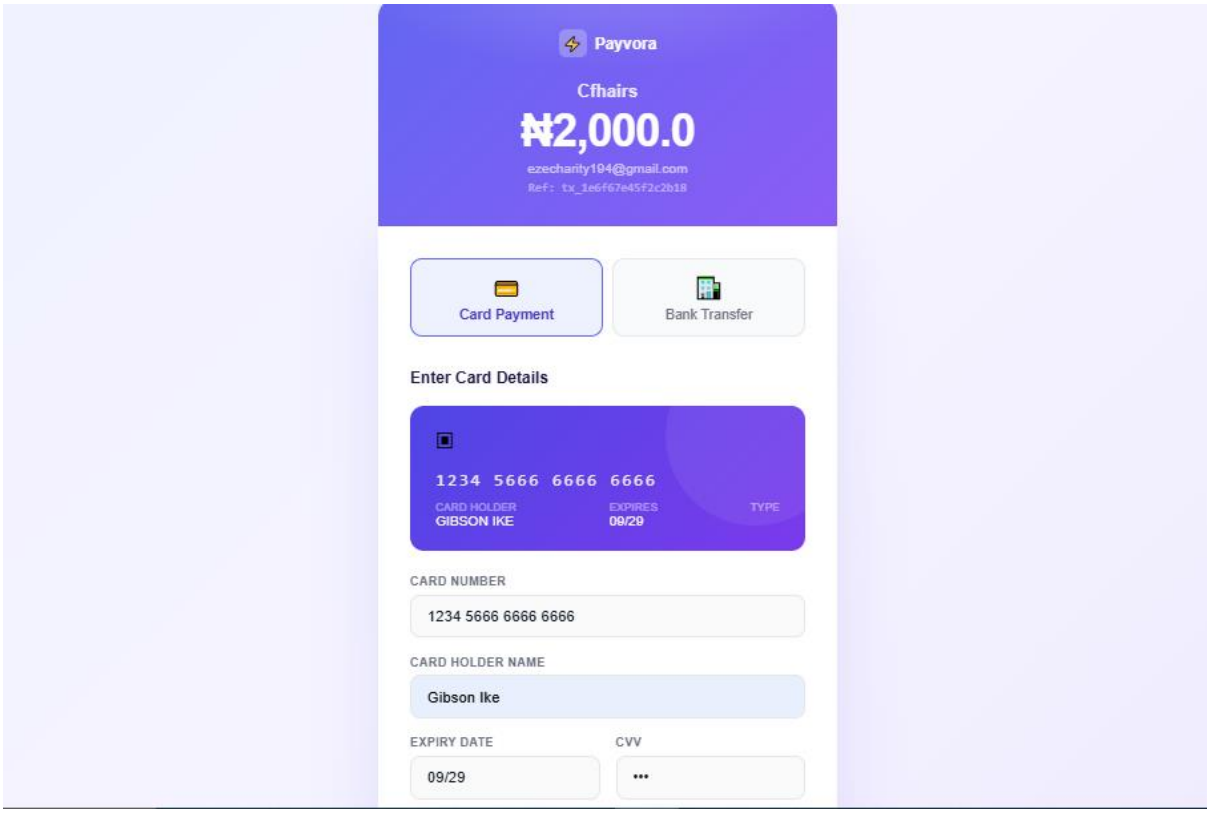

Figure 4.6 Checkout page

PDF Receipt: Every PDF receipt receives an individualized style with the merchant's branded color palette. In particular, the background of the header bar consists of the dominant color selected from the logo with superimposed transparent geometric circles and diagonal stripes. The business name of the merchant displays in bolded and colored white letters, while the logo is placed in a white rounded box. All of the elements such as amount box, status icon, and row

lines are styled in the same color as the main one of the merchant's brand palette. A unique pattern at the bottom right corner created out of the transaction reference string identifies each receipt.

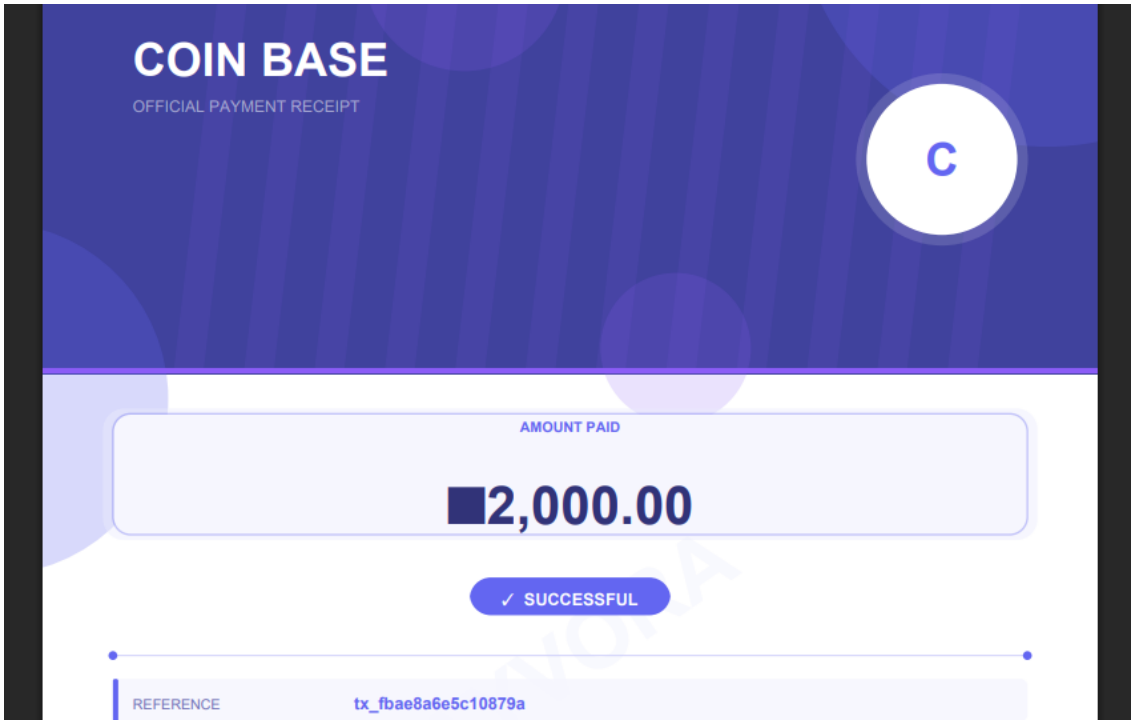

Figure 4.7: Receipt PDF

Receipt Email: The HTML email embeds the merchant's logo and company name through a gradient header, the total payment in a green-colored block underneath it, and transaction information in a tabulated form on two separate columns. The PDF attachment of the receipt contains the reference number of the transaction. The emails were delivered in 3 to 8 seconds from the completion of the payment process.

4.6.2 Discussion of Results

All of the five specified objectives have been met by the Payvora project successfully, as all test cases conducted by the developers proved to be correct. Here follows a description of the achievements regarding all objectives set out previously.

Payment Gateway Features: All of the 16 cases have been passed successfully via the use of the main API that takes care of the operations involved. All of the card data are verified client-

side prior to any request being sent to the server through the use of the Luhn algorithm, real-time checking of the date of expiration, and CVV number validation.

Fraud Detection Accuracy: This two-level hybrid approach accurately categorizes each transaction according to its risk profile with the PaySim XGBoost model training alongside the rule engine. The employment of the PaySim dataset that captures fraud cases in Africa – makes the trained model much more relevant for fraud detection within Nigeria than the majority of studies that employ European credit card fraud data sets. The rule engine gives readable descriptions of the reasons for each detected fraudulent transaction – something not present in any of the three Nigerian payment gateways currently in use and studied by literature.

Creation of Receipt: The receipt generator created receipts that were uniquely branded in the form of PDFs for each merchant account during the test process. The ColourThief tool extracted brand colors from logo URLs, and the color palette generated was consistent in nature. The hash code helped to generate different color schemes based on unique business name. The geometry used to identify receipts was unique for each unique transaction ID.

Sending Email: Receipt emails were sent with HTML and PDF attachment within 3 to 8 seconds after completing payments in all cases. The temporary PDF cleanup worked to remove all temporary PDF files.

Merchant Dashboard & Analytics: The merchant dashboard was capable of fetching and displaying the live MongoDB transaction data, computing statistical values from the live dataset on each refresh, searching for references and emails and filtering by status and method at the same time, streaming PDF download of all completed transactions, and saving all updated profile data, password bcrypt hashes, and webhook setups to MongoDB.

Webhooks: The webhook setup worked perfectly by delivering the POST request to a test URL within 2 seconds of making the transaction. All required fields were present in the payload

including the event type, transaction reference number, amount, email, reason, payment method used, business name, and timestamp of completion.

The major drawback for this system is that fake data is used in order to generate virtual accounts and process cards because acquiring a licensed API of a Nigerian bank as well as certification for card processing (PCI DSS) requires permissions, which were not available within the scope of the current assignment. However, since both components have been built with pluggable interfaces, changing from mocks to real APIs will require absolutely nothing in terms of architectural modification. Moreover, the performance of the fraud model on live data will continually be enhanced because of the constant training pipeline included in the architecture, which resolves the issue identified by Carneiro et al. (2017).

CHAPTER FIVE

SUMMARY, CONCLUSION, AND RECOMMENDATIONS

5.1 Introduction

This chapter discusses the conclusions reached at the end of developing and evaluating Payvora in relation to how successful the intended objectives have been realized and how significant the effort has been. The chapter further highlights any limitations that the present implementation has and suggests ways forward in order to make Payvora a full-scale payment gateway platform.

5.2 Summary of the Study

This project aimed to design and develop a full-stack AI-driven payment gateway system to address key limitations in the Nigerian payment gateway ecosystem. The project focused on five major objectives: building a functional payment gateway API for merchants, developing an AI-powered fraud detection system, implementing an automated branded PDF receipt generator, creating an online merchant analytics dashboard, and integrating a webhook mechanism for real-time event handling. All objectives were successfully achieved through the development of a system using FastAPI for the backend and React 18 for the frontend. The platform supports card and bank transfer payments, processes transactions through a two-layer fraud detection system, generates customized branded receipts, sends email receipts instantly, and delivers real-time webhook notifications to merchants.

The fraud detection mechanism was trained using the PaySim mobile money simulation dataset, which is more relevant to the Nigerian financial context than commonly used European credit card datasets. An XGBoost model trained on SMOTE-balanced data provides fraud risk predictions in under 10 milliseconds while combining machine learning with a rule-based system for transparent fraud explanations. Additionally, the receipt generation feature automatically personalizes receipts based on merchant branding, while the merchant dashboard

enables transaction monitoring, fraud score visualization, receipt downloads, API management, and settings configuration through a user-friendly interface.

5.3 Achievement of Objectives

The goals of this project have been successfully met. The full-stack payment gateway has been successfully built to deal with the entire payment process, starting from the transaction initiation and ending with the generation of the receipt and webhook notifications, as all the necessary API and interface tests passed successfully. The artificial intelligence-based system for fraud detection has been delivered by applying an XGBoost model on the PaySim data, which made it possible to achieve real-time fraud detection with increased accuracy due to the hybrid approach based on machine learning and rules.

Furthermore, the automated receipt generation in branded PDF format has been realized, making it possible for merchants to get their uniquely styled receipts automatically after the completion of payments.

5.4 Contributions of the Study

The contribution made by the project is huge in terms of its application in payment systems and fraud detection through artificial intelligence in Nigeria. Firstly, the feasibility of creating an AI-powered payment system in the form of a payment gateway through open-source technologies shows that sophisticated payment systems can actually be developed academically.

Secondly, the proposed project provides a more transparent mechanism to detect fraud with merchants being provided with fraud score and explanation. Thirdly, the PaySim mobile money dataset, which represents African payments more closely than European datasets commonly used in fraud detection projects, was used in the project. Lastly, the automatic receipt generation function included colour detection of merchant logos to create professional receipts.

5.5 Limitations of the Study

However, some constraints still persist despite the achievement of the project objectives. For example, while the current implementation is based on test data used to create virtual accounts and process card payments, a production implementation will need licensed bank APIs, regulatory permits, and an efficient payment system. Another limitation is that while the fraud detection model was trained using the PaySim dataset, the dataset contains fabricated data that might not entirely capture the Nigerian specific fraudulent activities.

Additionally, the fraud detection system was deployed within a local development machine where transaction scalability testing was not performed. This implies that more scalability work should be done if the software is deployed into production. Moreover, the email notification feature uses Gmail SMTP service which works well during testing.

5.6 Recommendations for Future Work

The following are some suggestions that will contribute towards future improvement of the Payvora system based on the results of this project. The development of Payvora should involve incorporation of live APIs of banks and payment processors in order to allow live bank and card transactions to occur. It would also be important to continually retrain the fraud detection model based on actual Nigerian transaction data.

In terms of further developments that can be made to the Payvora system, these could include deploying it online for accessibility purposes, incorporating an email verification mechanism to recover passwords for user accounts, improving the security of the API by introducing rate limits, and allowing merchants to export their transactions.

5.7 Conclusion

From this study, it is evident that it is possible to design an AI-powered payment gateway that incorporates fraud detection, receipt generation, and real-time merchant analytics using open-source tools. This study has filled some crucial gaps in existing payment gateways in Nigeria,

such as fraud transparency, automatic receipt generation, and contextually less relevant fraud detection datasets.

The study has been successful in implementing a fraud detection model utilizing the PaySim dataset, automatic receipt generation model for the company's branded receipts, and an integrated merchant analytics dashboard, along with designing the payment API for the entire transaction process. As Nigeria's payment market grows continuously, Payvora lays the foundations for future enhancements in this project and possible commercial implementation.

REFERENCES

Abdallah, A., Maarof, M. A., & Zainal, A. (2016). Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68, 90–113. https://doi.org/10.1016/j.jnca.2016.04.007

Adewale, O., & Ibrahim, M. (2022). Digital payment adoption among SMEs in Nigeria: Barriers, drivers and implications. *Journal of African Business*, 23(4), 112–129. https://doi.org/10.1080/15228916.2022.2034567

Alarfaj, F. K., Malik, I., Khan, H. U., Almusallam, N., Ramzan, M., & Ahmed, M. (2022). Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms. *IEEE Access*, 10, 39700–39715. https://doi.org/10.1109/ACCESS.2022.3166891

Alharbi, A., Alsubhi, K., Alzahrani, A., & Alsolami, F. (2022). Fraud detection model based on multi-verse features extraction approach for smart IoT environments. *IEEE Access*, 10, 16600–16612. https://doi.org/10.1109/ACCESS.2022.3149131

Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. *International Conference on Computing Networking and Informatics (ICCNI)*, 1–9. https://doi.org/10.1109/ICCNI.2017.8123782

Bahnsen, A. C., Aouada, D., Stojanovic, A., & Ottersten, B. (2016). Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, 51, 134–142. https://doi.org/10.1016/j.eswa.2015.12.030

Benchaji, I., Douzi, S., & El Ouahidi, B. (2021). Enhanced credit card fraud detection based on attention mechanism and LSTM deep model. *Journal of Big Data*, 8(1), 1–21. https://doi.org/10.1186/s40537-021-00530-z

Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision Support Systems*, 50(3), 602–613. https://doi.org/10.1016/j.dss.2010.08.008

Böhme, R., Christin, N., Edelman, B., & Moore, T. (2015). Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives*, 29(2), 213–238. https://doi.org/10.1257/jep.29.2.213

Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science*, 17(3), 235–249. https://doi.org/10.1214/ss/1042727940

Carneiro, N., Figueira, G., & Costa, M. (2017). A data mining based system for credit-card fraud detection in e-tail. *Decision Support Systems*, 95, 91–101. https://doi.org/10.1016/j.dss.2017.01.002

Central Bank of Nigeria. (2012). *Cash-less Nigeria policy guidelines*. CBN Policy Document.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 321–357. https://doi.org/10.1613/jair.953

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Dahlberg, T., Mallat, N., Ondrus, J., & Zmijewska, A. (2008). Past, present and future of mobile payments research: A literature review. *Electronic Commerce Research and Applications*, 7(2), 165–181. https://doi.org/10.1016/j.elerap.2007.02.001

Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. *IEEE Symposium Series on Computational Intelligence*, 159–166. https://doi.org/10.1109/SSCI.2015.33

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3), 319–340. https://doi.org/10.2307/249008

Deloitte. (2020). *AI adoption in financial services: Global survey report*. Deloitte Insights.

Dhankhad, S., Mohammed, E., & Far, B. (2018). Supervised machine learning algorithms for credit card fraudulent transaction detection: A comparative study. *IEEE International Conference on Information Reuse and Integration (IRI)*, 122–125. https://doi.org/10.1109/IRI.2018.00025

Elreedy, D., & Atiya, A. F. (2022). A comprehensive analysis of synthetic minority oversampling technique (SMOTE) for handling class imbalance. *Information Sciences*, 505, 32–64. https://doi.org/10.1016/j.ins.2019.07.070

Erl, T. (2005). *Service-oriented architecture: Concepts, technology, and design*. Prentice Hall.

Federal Inland Revenue Service. (2020). *Guidelines on electronic receipts and invoicing in Nigeria*. FIRS Publication.

Fernandez, A., Garcia, S., Herrera, F., & Chawla, N. V. (2018). SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, 61, 863–905.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

GSMA Mobile Money. (2022). *State of the industry report on mobile money 2022*. GSMA.

Hassan, M., Al-Razgan, M., & Abdullah, A. (2023). Deep learning with advanced feature engineering for credit card fraud detection. *Applied Intelligence*, 53(2), 1891–1908. https://doi.org/10.1007/s10489-022-03962-7

Ihenetu, S. (2021). Fintech and financial inclusion in Nigeria: The emergence of mobile payment platforms. *Journal of African Business and Economic Development*, 4(1), 23–41.

Ileberi, E., Sun, Y., & Wang, Z. (2022). A machine learning based credit card fraud detection using the GA algorithm for feature selection. *Journal of Big Data*, 9(1), 1–17. https://doi.org/10.1186/s40537-022-00573-8

Jack, W., & Suri, T. (2011). Mobile money: The economics of M-PESA. *NBER Working Paper No. 16721*. National Bureau of Economic Research. https://doi.org/10.3386/w16721

Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P. E., He-Guelton, L., & Caelen, O. (2018). Sequence classification for credit-card fraud detection. *Expert Systems with Applications*, 100, 234–245. https://doi.org/10.1016/j.eswa.2018.01.037

Kumar, V., Sinha, D., & Tyagi, A. K. (2022). Hybrid real-time credit card fraud detection using machine learning and stream processing. *International Journal of Information Management Data Insights*, 2(1), 100–119. https://doi.org/10.1016/j.jjimei.2022.100050

Lopez-Rojas, E., Elmir, A., & Axelsson, S. (2016). PaySim: A financial mobile money simulator for fraud detection. *The 28th European Modeling and Simulation Symposium (EMSS)*, 249–255.

Nielsen, D. (2016). *Tree boosting with XGBoost: Why does XGBoost win every machine learning competition?* (Master's thesis). Norwegian University of Science and Technology.

Oluwafemi, B., & Adeyemi, J. (2023). Electronic payment fraud in Nigeria: Patterns, channels and prevention strategies 2018–2022. *Nigerian Journal of Technology*, 42(1), 78–94.

Oyewole, K., Adebayo, F., & Ibrahim, T. (2023). Nigeria-specific fraud detection: Identifying unique patterns in mobile and digital payment fraud. *Journal of Financial Crime*, 30(3), 901–918. https://doi.org/10.1108/JFC-06-2022-0134

Priscilla, C. V., & Prabha, D. (2023). Explainable artificial intelligence for credit card fraud detection: A SHAP and LIME approach. *Expert Systems with Applications*, 212, 118–145. https://doi.org/10.1016/j.eswa.2022.118519

Quah, J. T. S., & Sriganesh, M. (2008). Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications*, 35(4), 1721–1732. https://doi.org/10.1016/j.eswa.2007.08.093

Randhawa, K., Loo, C. K., Seera, M., Lim, C. P., & Nandi, A. K. (2018). Credit card fraud detection using AdaBoost and majority voting. *IEEE Access*, 6, 14277–14284. https://doi.org/10.1109/ACCESS.2018.2806420

ReportLab. (2023). *ReportLab user's guide: PDF generation for Python*. ReportLab Inc. https://www.reportlab.com/docs/reportlab-userguide.pdf

Sailusha, R., Gnaneswar, V., Ramesh, R., & Rao, G. R. (2020). Credit card fraud detection using machine learning. *International Conference on Intelligent Computing and Control Systems (ICICCS)*, 1264–1270. https://doi.org/10.1109/ICICCS48265.2020.9120956

Taha, A. A., & Mahersia, H. (2021). Meta-classification for credit card fraud detection using stacked generalization. *Applied Soft Computing*, 112, 107–121. https://doi.org/10.1016/j.asoc.2021.107143

TechEmpower. (2023). *Web framework benchmarks: Round 22*. TechEmpower Inc. https://www.techempower.com/benchmarks/

West, J., & Bhattacharya, M. (2016). Intelligent financial fraud detection: A comprehensive review. *Computers and Security*, 57, 47–66. https://doi.org/10.1016/j.cose.2015.09.005

Zanin, M., Papo, D., Solanes, A., Gonzalez, J. J., Deco, G., Boccaletti, S., & Sousa, P. (2018). Combining complex network and data mining techniques for extracting information on credit card fraud. *Chaos, Solitons and Fractals*, 108, 163–172. https://doi.org/10.1016/j.chaos.2018.01.023

Zhang, L., Wang, H., & Liu, Y. (2021). HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture. *Information Sciences*, 557, 302–316. https://doi.org/10.1016/j.ins.2019.05.023

APPENDIX

Appendix A: Source Code Listings

Appendix A1: Backend Entry Point (app/main.py)

```
from dotenv import load_dotenv
load_dotenv(override=True)
from fastapi import FastAPI
from fastapi.staticfiles import StaticFiles
from fastapi.middleware.cors import CORSMiddleware
from app.routes import merchant, payment
app = FastAPI(title="Payvora Payment Gateway API", version="1.0.0")
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)
app.mount("/static", StaticFiles(directory="app/static"), name="static")
app.include_router(merchant.router, prefix="/api/v1", tags=["Merchant"])
app.include_router(payment.router, prefix="/api/v1", tags=["Payments"])
@app.get("/")
def root():
    return {"message": "Payvora Payment Gateway API", "status": "running"}
```