

# Transactional Committal Portal and Database Resolution Controller For Mobile Banking Transactions

Felix. U Ogban, Ndifon Godswill Brendan, Ceasar E. Eko

*Department of Computer Science, Faculty of Physical Sciences, University of Calabar, Calabar, Nigeria*

Received: 11 February 2023; Revised: 23 February 2023; Accepted: 25 February 2023; Published: 28 March 2023

**Abstract:-** Mobile banking has become a bulging feature in banking operations in Nigeria as more and more banks are accepting this technology to provide the growing population of their clients with fast, accessible, reliable, and quality services. However, there are numerous problems associated with mobile banking: non-committals and non-reversal is the utmost. In this paper, a few scenarios such as Debit without dispense, Debit without Committal, and Debit without credit were considered. Information has it that it takes some banks 24hrs to 72hrs or more to reverse failed transactions, and some bank customers have forfeited their money through this means. The problem therefore can be narrowed to outright failures due to timeout in the network during committals into a database or failures to resolve transactions since there should be no gain without a loss or vice versa. A controller bridge was developed to mirror or sniff mobile transactions from bank to bank; internally (bank A to A) or externally (bank A to B), ascertain the success (full committals) of the transactions, or initiate a resolution if a transaction fails. The Waterfall software development model was adopted for use. The software bridge was tested by simulation using mockup data. The system was developed with PHP, HTML, CSS, JavaScript (Ajax and jQuery), and Bootstrap framework. The implementation was done on a local host with two different browsers housing the banks and the bridge residing in any of them. The results in time concerning sniffing, confirmation, and resolution were outstanding. Implying that, the bridge can improve transactional resolutions faster than ever. Banks should be encouraged to adopt or integrated this new committal bridge into their existing system so that failed transactions can be resolved within a minutes.

**Keywords:** Mobile banking, Banking Transactions, Database Resolution, Comittal portal, Transactional committals, Simulation

## I. Introduction

The approach of Technology delivered with it relative ease with which the banking industry executes their organizations electronically without direct contact particularly as it identifies with their various clients. One such complicated Technology is the introduction of mobile banking in Nigeria. Mobile banking refers to the use of a mobile device to carry out financial transactions. The service is provided by some financial institutions, especially banks. Mobile banking enables clients and users to carry out various transactions which can vary counting on the institution. Mobile banking may be a facility that permits customers to initiate and/or perform banking tasks on their smartphones or computers. This service is provided by all the banks in Nigeria. Customers can use mobile banking applications to carry out Some of the following mobile transactions which include:

- Bank-to-bank transfers
- Transfer of funds to self
- Payments to third parties (rent payments, bill payments, etc.)
- Giving standing instructions for periodic payments
- Payments via Neft/Imps/Retg/Upi/Mmid

All of these depend on a repository: the database where the debits and credits are committed for the transaction to be complete.

The proposed system database keeps a record of all interbank and intra-bank activities so that it can easily be accessed, managed, and updated. The databases typically contain aggregations of data records or files, containing information about interactions with specific customers. Typically, the database manager provides users with the power to regulate read/write access, specify report generation, and analyze usage. Some databases offer ACID (atomicity, consistency, isolation, and durability) compliance to ensure that data is consistent when transactions are complete. A Database Transaction is a logical unit of processing in a database management system (DBMS) that entails one or more database access operations. In a nutshell, database transactions represent real-world events of any enterprise. All types of database access operations which are held between the beginning and end transaction statements are considered as a single logical transaction in the database management system (DBMS). During the transaction the database is inconsistent. Only once the database is committed the state is modified from one consistent state to a different one. A database is a shared resource accessed because It is used by

many users and processes concurrently. For example, the banking industry, railway, and air reservations systems, stock exchange monitoring, supermarket inventory, checkouts, etc. Not managing concurrent access may create issues like:

- *Hardware failure and system crashes:*

In case of a system crash, the systems hang up and need to be rebooted. These failures occur because of a hardware malfunction or a bug within the database software or the operating system itself. It causes the loss of the content of volatile storage and brings transaction processing to a halt. The content of nonvolatile storage does not affect by this type of failure. The assumption that hardware errors and bugs bring the system to a halt, but do not corrupt the nonvolatile storage contents is known as the Fail-Stop Assumption.

- *Concurrent execution of transactions, deadlock, or slow performance*

Implementation of concurrency control in a transactional system requires first, understanding the basics of collisions and their effect on such transactions. Therefore, we can either avoid them by detecting or resolving them. For transactions ( which entail the collections of actions that potentially modify two or more entities) to be upheld, the avoidance of actions that does not commit to memory must be taken. Concurrency control and transactions are not simply the domain of databases instead, they are issues that are potentially pertinent to all the architectural tiers of commitals (Ogban and Asagba, 2011). The objective of concurrency control is to ensure the serializability of transactions in a multi-user database management system such that for every debit, there is a corresponding Credit or vis-versa. Because it helps to ensure data integrity and consistency during a database system, concurrency control is one of the most critical activities performed by a DBMS. If concurrency control is not maintained, three serious problems may be caused by concurrent transaction execution: **lost updates, uncommitted data, and inconsistent retrievals**. A simulation is said to be important and useful to apply because of the following reasons:

- *Hazardous situation*
- *No availability of real data*
- *Nonrescuable situation*

A collision in a database is said to occur when two activities, which may or may not be full-fledged transactions, attempt to change entities within a system of record. There are three fundamental ways (Ogban and Asagba, 2011) that two activities can interfere with one another; namely Dirty, non-repeatable, and phantom read.

**i. Dirty read:** Activity 1 (A1) reads an entity from the system of record and then updates the system of record but does not commit the change (for example, the change hasn't been finalized). Activity 2 (A2) reads the entity, unknowingly making a copy of the uncommitted version. A1 rolls back (aborts) the changes, restoring the entity to the original state that A1 found it in. A2 now has a version of the entity that was never committed and therefore is not considered to have existed.

**ii. Non-repeatable read:** A1 reads an entity from the system of record, making a copy of it. A2 deletes the entity from the system of record. A1 now has a copy of an entity that does not officially exist.

**iii. Phantom read:** A1 retrieves a collection of entities from the system of record, making copies of them, based on some sort of search criteria such as "all customers with first name Bassey." A2 then creates new entities, which would have met the search criteria (for example, inserts "Bassey Kanu" into the database), saving them to the system of record. If A1 reapplies the search criteria, it gets a different result set. In our case with the mobile banking application, the problem is not that of reading the memory object, executing the deduction therein, and committing the same but mechanically not dispensing the cash thereafter or reversing failed transactions.

However, there are numerous problems associated with mobile banking. For this work, the researcher will just mention a few scenarios:

- A bank customer, who wishes to make a withdrawal through a Mobile banking application, undergoes the authentication process and receives a debit alert from the bank only to realize that the Mobile banking application was unable to dispense the cash due to an unknown error at that point (Debit without dispense).
- A customer who wants to buy something from a supermarket uses the cashless policy of paying through the POS. In the process of carrying out the transaction, he receives a debit alert without the receiver being credited neither was a reversed alert issued immediately, in some cases beyond 24 hours (Debit without Committal).
- A customer, who wishes to shop online pays for what was bought, receives a debit alert, closes the transaction, and after a while, receives a call from the online agent informing the client that the company was not credited (Debit without Credit).

Findings show that it takes some banks 24hrs to 72hrs to reverse these failed transactions, nevertheless, some bank customers have lost their money through this means.

The problem therefore can be narrowed to outright failures due to timeout in the network during committals into a database or failures to resolve transactions since there should be no gain without a loss or vice versa. To achieve this, the following objectives are established;

1. To design a framework for smart transaction committal and database resolution using a controller bridge.
2. To implement a controller bridge that resolves the transactional failure.
3. To test by simulation, the bridge uses sample data from the bank.

## II. Literature

Globalization and Information Communication Technology (ICT) took the world by storm and this posed incredible difficulties to the money businesses (Garuba, 2008). It has changed the way things are done today regardless of what you plan to do whether you will work with individuals or cash, with words or numbers, innovation assumes a significant part. This offered to ascend to electronic saving money which is the use of data innovation (media transmission and figure) to transmit information starting with one point and then onto the next (Sriramya and R.A. Karthika, 2020).

Mobile banking systems allow users to perform bank-related transactions like balance checks, account transactions, bill payments, fund transfers, credit/debit card management, etc. through mobile telecommunication devices like mobile phones or PDAs (personal digital assistants).

### 2.1 Mobile Banking Advantages & Disadvantages

Mobile banking offers many advantages to both, users and service providers and the bank. It is fast and stress-free to use and saves time. An internet connection is an essential part of online banking because the internet cannot be accessible everywhere which is a major problem in developing countries. However, many individuals can find mobile connectivity at places where internet connections can't be found. Mobile banking is cost-effective for providers because the cost of mobile banking is far less compared with onsite banking. Various sorts of banking services and transactions can be performed with mobile banking. However, mobile banking has many disadvantages such as transfers between interbank may take longer time to resolve when an error occurs like debit without committal since inter-bank transactions must go through the inter-switch before being resolved, this may take more than 72 hours to two weeks. Security issues also are a major concern (Osazevbaru and Yomere, 2015).

### 2.2 Reasons for Mobile Banking Failed Transactions

- Mobile dispenses might be out of money yet for a few reasons or use of unfit notes/mutilated notes.
- the wrong category of notes happens when the money handling agency or the bank staff put the wrong denomination in the mobile banking application (Madawaki et al, 2014).
- Network or power failure as in the case of carrying out a transaction online as mentioned in scenario three (3) of the statement of the problem.
- Technical/mechanical obstacle in mobile Banking: There might be a physical deficiency e.g. reject plate full, broken rollers and clasps, distributors, and so on as well as software e.g. the Front-End Processed (FEP) issue in the mobile bank bringing about non-regulation of money after a win reaction from the switch (Madawaki et al, 2014).

According to a sterling bank report (2021), A Failed Transaction is when your account is debited for a transaction that did not give you the intended value. It happens when:

- (1.) *The mobile banking application does not give you cash but your account is debited for the same amount.*
- (2.) *The mobile bank application declines your transaction, but your account is debited for the same value.*

According to the analysis obtained from a sterling bank website as shown in the figure. 1, it takes the bank less than 2 days to reverse any failed transaction involving intra-bank, and for an inter-bank failed transaction, it takes five (5) days to one week that is if the clients' complaint was received early, most time a month before the reversal can be done. And for foreign transactions, it takes 45 days to 4 months before the problem can be resolved. From the above analysis, the researcher found out that transaction involving the same bank tends to be resolved within a short period because it involves a direct linkage to the bank server while transactional committal involving other banks need to go through an Inter-switch, the regulations bodies like the central bank of Nigeria and inter-bank settle system plc. (NIBSS). More prominent, Inter-bank transactions are the major reason there are delays in the resolution of debt after a failed transaction.

| Transaction | Local Transactions                       |                                       | Foreign Transactions     |
|-------------|--|---------------------------------------|--------------------------|
| Channel     | Sterling Bank's ATM/POS/Merchant Website | Other Banks' ATM/POS/Merchant Website | ATM/POS/Merchant Website |
| Reversal    | 2 Days                                   | 5 – 8 Days                            | 45 Days                  |

Figure 1: When a Failed Transaction Is Reversed

### 2.3 Related Works

#### 2.3.1 Interswitch:

Transactions are defined as business processes with starting points and endpoints. They have specific activities and timelines, through which these transactions take place to achieve the desired effects. Take for example mobile banking transaction has many underlying operations, controls, and processes aimed at making it a successful transaction between the start and finish point.

According to TECHWEEZ 2021, Interswitch offers financial institutions a central, comprehensive, and outsourced platform through which they can implement and manage their user transactions. The switch thus allows a consumer from any bank on this shared platform to perform their transaction. The beauty of the switch is that it allows for different devices to plug- in from pay stack to ATMs to POSs or any other mobile banking devices. It also provides users with transaction authorization by connecting to hosts, interchanges, and co-networks. Other additions allow for plugging in internet payment gateways as well as mobile payments. With an increasing number of banks issuing cards as well as institutions accepting card payments as well as merchants accepting payments made by these parties, a third party offers a service as an intermediary between merchants and institutions offering the cards. The intermediary serves as a clearing house for various transactions.

To clear these transactions, the third party relies on a transaction switch to route transactions received from merchants. A typical transaction sees the merchant initiate a transaction (say payment for goods at the Supermarket via POS), which is routed to the holder’s financial institution for approval. The response from the financial institution is then conveyed to the merchant via the switch. The first step is the transaction moving from the POS API to the acquiring bank’s switch, it then conducts credit authorization, and back to the POS API, the merchant is thus able to complete the transaction.

It is imperative to remember that there are other underlying processes which include loyalty programs for using their cards, and fraud detection to authenticate if indeed, it is the bearer performing the transaction. If either of these processes fails to meet the set controls and guidelines, the transaction is declined. With a central switch, these transactions are faster, cheaper, and platform-independent.

#### 2.3.2 INETCO Insight.

According to INETCO 2021, for many banks, payment processors, and card network providers, the main purpose of active/active architecture is to achieve load balancing, improve throughput, and guarantee response times. For one global card network provider that processes transactions for high net-worth clients across 14 countries in the Middle East and North Africa, adopting this type of infrastructure within their Postilion payments switch environment was crucial to handling their growing transaction volumes without lag or downtime. Postilion provides banking transaction-switching software that routes financial transactions. Most of the time Postilion switch software servers are set up in a relatively simple fashion, whereby transactions flow into the switch, which then sends them through to their appropriate destination as shown in figure 2

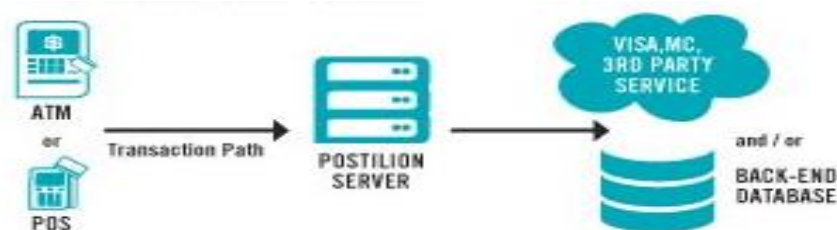


Figure 2: Typical postilion Transaction (Inetco 2021)

This provider needed a real-time solution that could monitor a complex payments network comprised not only of a Postilion active/active architecture with multiple servers and data centers, but also the interconnectivity of their payment switch,

card management system, internal network, telecom provider, and the 54 third party (ATM, POS, back-end) service providers they relied upon. Because the end customer experience was (and still is) paramount, even a one-second improvement in transaction speed would translate into significant returns, Then the introduction of INETCO Insight.

With INETCO Insight as shown in figure 3, the bank can monitor every transaction as it flows through their payments ecosystem and be alerted to any transaction issues, network slowdowns, or problems with any of their multiple servers before getting an angry call from a merchant or an irate high net-worth customer experiencing failed or declined transactions. The banks can capture data from every link of an end-to-end transaction and have both the application payload and network communications details (i.e.: request/response timings, transaction dollar values, network address information) easily accessible within one or two clicks. With a holistic view of the entire payments ecosystem, you can spot customer transaction bottlenecks before they become failures. INETCO Insight also provides the transaction level detail required to isolate whether transaction slowdowns or failures are due to the internal network, applications, or other third-party service providers, speeding up average mean-time-to-repair by 65-85%.

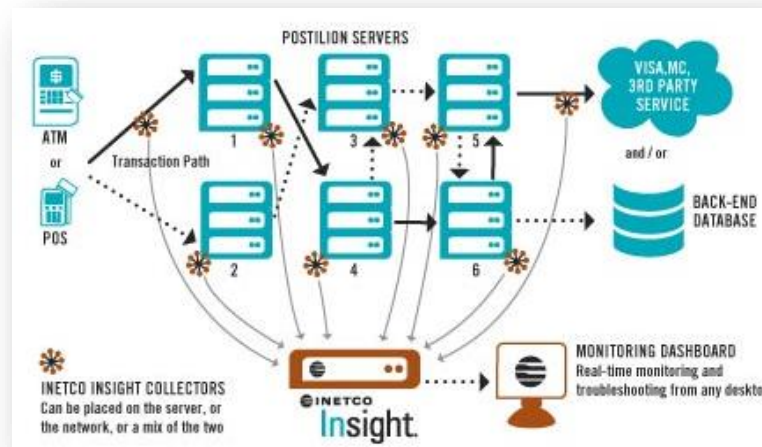


Figure 3: Diagram of complicated active/active postilion transaction (Inetco 2021)

From the review of INETCO Insight, three basic issues were not attended to such as database resolution challenges, staff inconclusive challenges, and transactional time-out challenges.

- Database resolution Challenge: Since the INETCO Insight application was not embedded or attached to the mobile banking application. It was hard for it to keep track of any transaction for easy resolution.
- Staff inconclusive challenges: Bank staffs were not able to interact with INETCO insight because it was offering a third-party service. The staffs were no able to query the INETCO insight database to checkmate the status of a transaction if a transactional failure occurred as opposed to the researcher’s proposed system.
- Transactional timeout Challenge: Transactional failure still exists because of service time out.

Hence, The researcher intends to implement and solve these issues.

### III. System Analysis And Design

The analysis model is a precise abstraction of what the desired system must do, and not how it will be done after the study of the existing system is completed. This includes a systematic study of the existing system. Interacting with the bank staff, and customers of Mobile banking regarding their expectations from the proposed system.

#### 3.1 Review of Existing System

Originally, when a cardholder wants to initiate a transaction through a mobile banking platform as shown in figure 4, the user provides all the necessary information. The mobile banking application then forwards this information to the host processor for confirmation. The host processor enters the transaction request to the cardholder bank to confirm if the requested fund is in the cardholder’s account. If the cardholder requests the transaction, the host processor (bank server)



takes the cash from the cardholder’s account after the confirmation from the bank server to the designated account. Once the funds are transferred from the customer account to the host processor (bank server) checking account, the processor sends the approval code to the mobile banking technology involved and then authorized the mobile banking application to carry out the transaction. The problem, therefore, is when the cardholder is debited without the mobile banking application carrying out the requested transaction after authorization has been given and reverse not done, the customer will have to wait till the close of work so that the person in charge of mobile banking transaction will print the report for the day before analysis can be made to that regard before the problem can be rectified whereas the customer care will keep informing the client that the bank is working on the complaint. This report is printed at night before analysis can be done and secondly, mailing to the Interswitch is done manually.

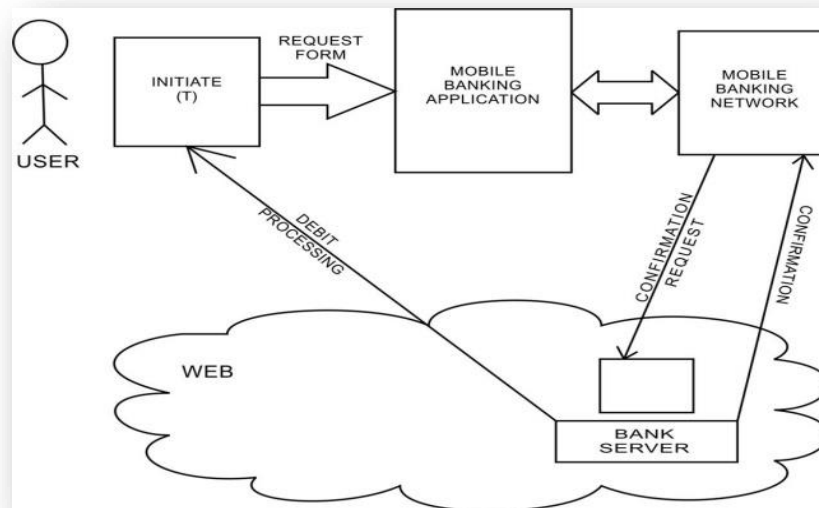


Figure 4: An overview of the existing system

A caption of a transaction between two persons using the Mobile Application. For a transaction, the mobile application could be ATM, POS, e-transact, pay stack, etc. now, the initiating bank could be different, meaning that it could be an intra-bank or inter-bank transaction but what matters at this stage is the fact that whether it is an inter-bank or intra-bank transaction, the originating bank is the same as the external bank when it intra-bank process while when it is inter-bank processes, the originating bank is different and the external bank is different as seen In figure 6

A transaction initiated by User A for a likely sum of ₦20, 000 to be transferred to User B as seen in figure 6, and then the transaction is originated and committed through the handshake to the External Bank. The handshake can be the communication between banks. The sniffing process is to check whether the transaction which was initiated has finally been committed, in as much as there are rules to checkmate databases not been made available to users because a client cannot send money to another account and then request for the receiver Account balance, is not necessary and so confirmation should be sent from whether the intra-bank or the inter-bank to the fact that the transaction was complete because once a transaction is not completed, the account or database of those accounts are still open and as long as there are still open, no other transaction can occur within them. This is the actual scenario but the researcher is not developing the whole system but rather the aspect of the bridge. The system will also show the relationship between the proposed bridge and the existing system. the external bank will need to send a confirmatory message to the bridge.

This System was developed using PHP, HTML, CSS, JavaScript (Ajax and jQuery), and Bootstrap framework. On the Admin Side, the system admin user can manage all the records of the client’s accounts and the admin can also make the transaction for the walk-in clients. Then, on the Client Side, as mentioned above, the client can make their transaction using the system and also track all their transactions. The system has many minor features which are relevant for this kind of system such as some Error trapping to prevent system errors due to human errors. The clients' credentials can be only created by the admin user and the proposed bridge can only be queried by the bank staff.

### 3.2. Database Design

The proposed system is expected to have a database in which any transaction that happens at a bank whether successful or not successful must be stored on the database has shown in figure 5, the database of the proposed system has six(6) tables namely: Customers, customers\_types, Accounts, Accounts\_Type, Transactions, and Transactions\_Type. For instance, the transaction\_Type has a transaction\_Id in it. Where transaction\_Id which keeps a record of all transactions that are successful or not for easy resolution.

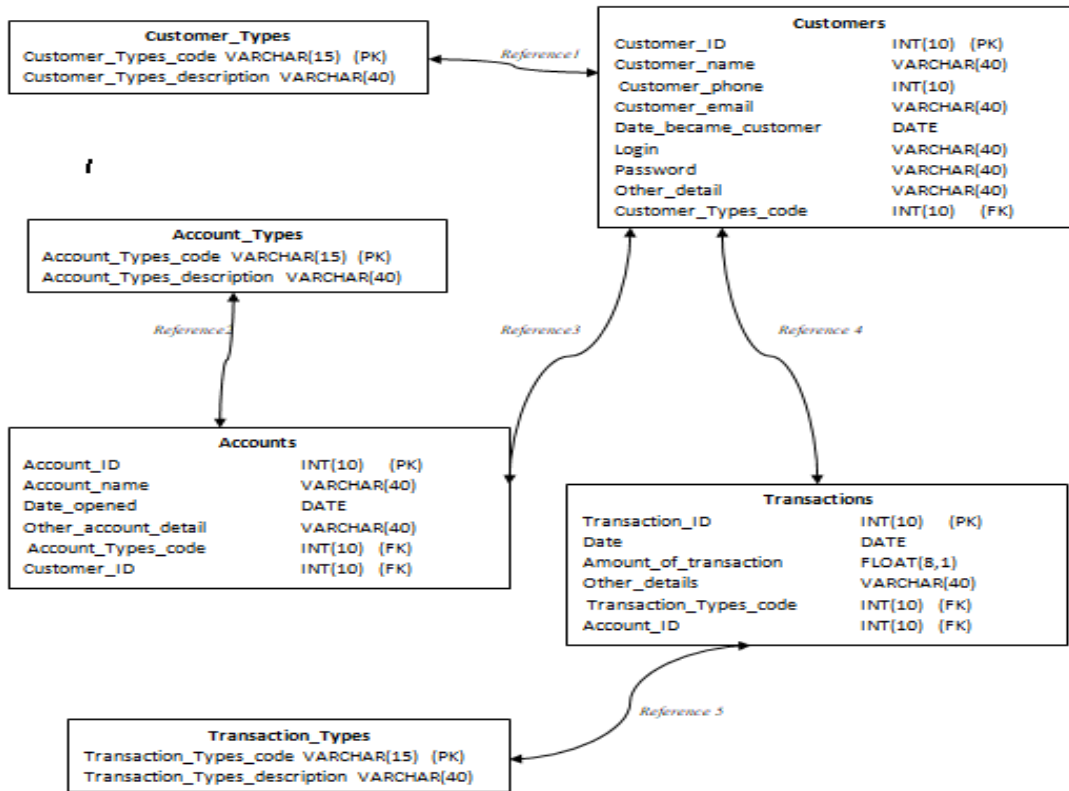


Figure 5: Database Design

#### 3.2.1 Implementation

The proposed Committal Bridge keeps a log of all the interbank activities and also resolves to fail transactional committal while each of the banks will still maintain a log of its activities. The bridge is to keep track of all transactional activities. The bridge is designed for very high traffic in the bank server. Extensive filtering is provided to block unnecessary traffic coming into the bank database to avoid concurrency. When an error occurs in the process of carrying out a transaction due to service time out or any other fact when this sort of error occurs, some transactional error cannot be accounted for by third parties' mobile monies agent but with the bridge, the bank staff can easily query the bridge database to know the status of a transaction since any transaction initiated must pass through the bridge so that the bridge can be on known of what is happening within that transaction. before bank A communicate with bank B to accredit a user in either bank, the API core will need to pass through a bridge of the pending transaction instead of direct communication, so that the bridge will be aware of the fact that a user in Bank A has transfer money to an account which is resident in bank B. So when Bank B receives the money, it acknowledges the bridge of the receipt so that it can be recorded at the bridge end that such transaction was successful. So when a user in bank B does not receive the money from user A in bank A, The proposed bridge resolves the issue without delay thereby reducing the time it takes the banks to resolve transaction committal to a minimal point. A query log of all the failed transactions was also incorporated into the bridge so that the banker can easily query the bridge to know the status of a transaction. The Bridge is expected to have the following Component:

**Sniffer module:** Sniffer is the process of monitoring transaction traffic in real-time, capturing all the data flowing to the sending bank and from the receiving bank. In essence, a sniffer module monitors and captures all transactions through a bank network for the main purpose of easier troubleshooting by the proposed bridge.

**Confirming module:** The confirming module must checkmate the user account to say if the requested amount is at the disposal. When a transaction is committed, the External bank must send a confirmatory message to the originating bank to show that the transaction was successful and verse visa.

**Credit or debit module:** A credit module in an account represents a transfer of the fund to that account while A Debit module in an account represents a transfer of funds from that account. Each transaction transfer fund from a credit account to a debit account. Both the Credit and Debit module has a confirming module and a database resolution

**Database Resolution module:** If the transaction in figure 6 is not committed, the bridge swings into action by invoking the Database Resolution module. In this module, attempt to redo the failed transaction and keep the time count. if it fails again, it tries the second time to reactivate the transaction and also keep the time count but if it fails again, the bridge tries to reverse the transaction to the sender.

With the above mention module of the proposed committal bridge, a customer complaint about transactional committal can be resolved within a period. The bridges confirm and then see the need to be able to resolve or not.

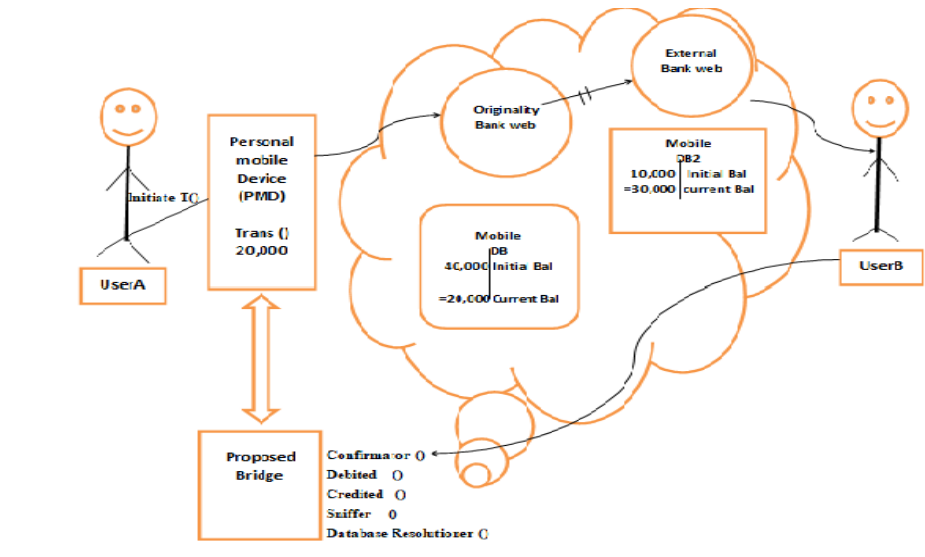


Figure. 6: Entity Relation Diagram

### 3.2.2 Testing

The system was simulated using two non-existing banks (for example, sun bank international and cyber bank of Africa). The system implementation was done on a local host with two different browsers housing the banks and the bridge residing in any of them. Before any transaction can occur within a bank, the transaction must be sniffed by the committal bridge so that the bridge will be aware of what is happening within an account. The bridge is to keep track of all successful and unsuccessful transactions in a bank for easy retrieval. The system is expected to have three databases for three transactions which are, a database for the inter-bank transactions, a database for the intra-bank transactions, and a separate database depending on how many banks are connected to the bridge which the proposed bridge uses to be able to cross-check with the other two databases. With the proposed committal bridge, the bank can easily query the bridge to know what happens within a transaction. As mentioned earlier, the researcher is not building a bank system afresh since it already exists but rather the bridge, and the relationship between the committal bridge and the existing system will be shown. The researcher has to test the proposed system by going through a few basic testing methodologies for testing web-based applications.

### 3.2.3 Deployment

The proposed bridge is an attachment to the existing mobile banking application. When the existing mobile application is installed, the committal bridge is installed automatically with it since it is an attachment. With this, the bridge can always sniff any transaction from the origin and then confirm if the transaction was successful or not. With the new bridge, mobile transaction failure can be automated with ease.



### 3.2.4 Maintenance

Maintenance is an essential work that needs to be carried out on a web-based application so it can continue to reliably and securely function. It is not adding new features. It is not checking log files or ensuring backups have run (these are housekeeping tasks). It is working on the code and the underlying platform to ensure that things are up to date and that it performs as its users would expect also.

### 3.3 System Architecture of the Proposed System

The process begins when a bank user initiates a transaction through a mobile banking application as shown in figure 7, the mobile banking application requests all the transaction detail before it can be processed. After the transaction type has been selected, the mobile banking application sends the unique EMV transaction code and transaction to the processor through the I/O board and modem. The processor uses this information to route the transaction to a committal bridge as shown in figure 7 to checkmate the transaction before forwarding the request to a mobile banking network that is associated with the card. By Federal regulation, each card is required to have two networks so that if the transaction cannot be processed with one mobile banking network it can be processed with the other network. The mobile banking network then sends this information to the customer bank to determine whether the transaction is approved, and the bank debits your account for the amount. This approval or denial is sent back to the committal bridge to forward the say confirmation to the mobile banking technology involve through the mobile banking network to carry out the transaction. Each further transaction is processed in the same manner. So if a transaction is determined not to be correct or an error has occurred during the process or a client was debited without committal, then the proposed bridge is expected to resolute it through a resolutioner and a confirmatory send in the process; the committal; the bridge is like a bridge between the mobile banking Network and the bank server which act as a confirmator, resolutioner, and acknowledge, with this new system a client debit without committal i.e. hanging process can always be resolve without a short period. The pre-emptor bridge sniffs any request or transaction in an account so that it can resolve any incomplete transaction if an error occurs.

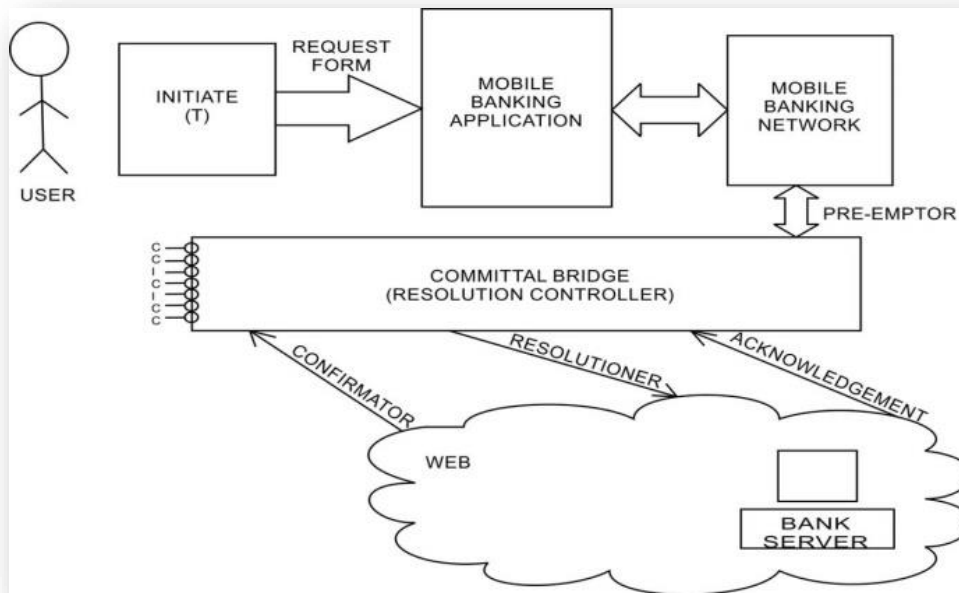


Figure 7: Overview of the proposed system

### 3.4 The system Model

In this phase, a logical system is built to fulfill the given requirements. The design phase of application development deals with transforming customer requirements into a logically working system.

### 3.4.1 Block Diagram Of the proposed system

The committal bridge as shown in figure 8, acts as an intermediary between the bank server and any of the mobile bank application networks; it also acts as a transaction sniffer so that before a transaction is carried out in a banking platform whether through a mobile naming App or any other means, the committal bridge sniff such transaction and also resolve any pending transactional committal within a short period. The committal bridge’s main object is to sniff any transactional within a bank network whether it involves interbank or intra-bank process and then tried to resolve any transactional committal, if a cardholder should initiate a transaction through a mobile banking application, the transaction must be sniffed by the bridge. the committal act as a confirmatory where an acknowledgment can be sent from, that is to say, that any transaction happening in the bank, will be sniffed by the committal bridge as shown in the figure 8 the pre-emptor bridge communicates with the bank server. and also sniff the requested information.

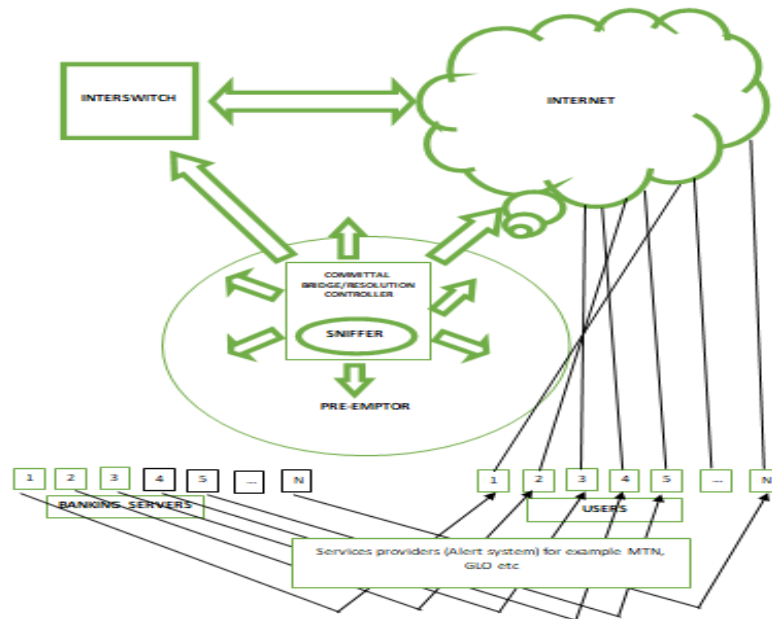


Figure 8: Block Diagram

### 3.5. System Implementation

Before the proposed system can be mounted on any system, A WAMP server must first be installed on that system. As shown in figure 9, WAMP stands for windows, Apache, MySQL, and PHP. Essentially, WAMP set up all the aforementioned applications on a windows box. Apache automatically becomes the local web server of the system so that it can also be tested locally before making it go live.

Apache’s main job is to establish a connection between a server and the browser of a website while delivering files back and forth between client-server structures. Therefore, it is an open-source local server that works both on UNIX and Windows servers while MYSQL as shown in figure 10 controls everything that happens on the database. Insertion, deletion, modification, the update can be made on the database. the Apache and MYSQL must always turn active as shown in figure 9 above before the new application can be lunch on a browser. With this, the application was then ready to be mounted on the Location *C:\xampp\htdocs* and then view on the browser through the URL link: *localhost/Banking/*

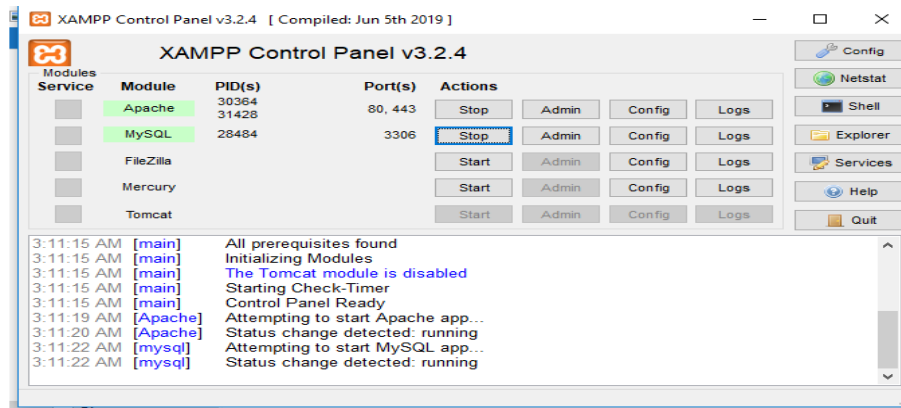


Figure 9: WAMP control panel

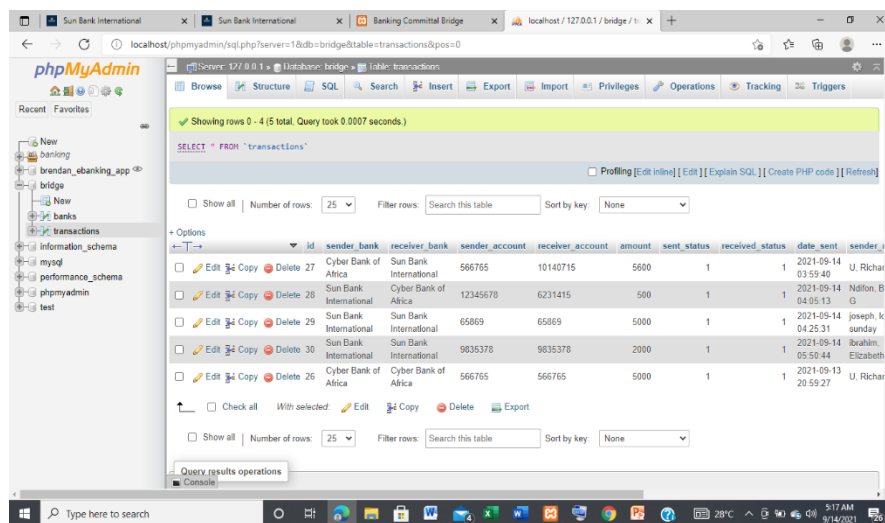


Figure 10: the application Database

During carrying out this research work, some codes were written to do a specific task. Figure 11 script was coded using Ajax and the code as shown in figure 11 was used to check and resolve the transactional failure.

```

94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
  
```

```

<script src="js/functions.js"></script>
<script type="text/javascript">

function resolve() {
$.ajax({
url: './api/?resolve=true',
method: 'GET',
//data: $(this).serialize(),
dataType: 'json',
error: err => {
console.log(err)
//end_loader()
},
success: function (resp) {
console.log(resp);
}
});
setTimeout('resolve();', 60000);
}
setInterval('resolve();', 60000);
</script>
</body>
</html>
  
```

Figure 11: code for error checking and automated resolution

#### IV. Results and Discussion

The process of testing the system involves running a series of a transaction on the new system to ascertain its performance and functionality to users concerning the reversal of funds when a bank customer is debited without committal. The testing process as shown in the figures shows if the system is performing its primary aim or is even working properly.

The implementation was done on a local host with two different browsers housing the banks and the bridge residing in any of them. Two non-existing banks (sun bank international and cyber bank of Africa) were simulated to initiate a transaction from sun bank international to the cyber bank of Africa. When a sun bank user initiates a transaction through a mobile banking application as shown in Figure 12 below, the application gives the user room to input the recipient detail and the intended amount to be transferred, After which the application confirm the correctness of the details by fetching the recipient name before initiating the transaction. Immediately after the transaction is submitted, the intended amount is reduced or debited from the user’s account as shown in figure 13, and credited to the recipient account number.

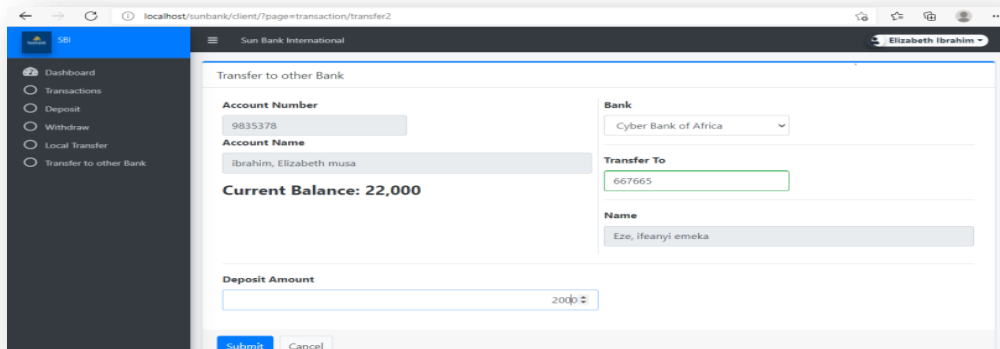


Figure 12: Initiate Transaction

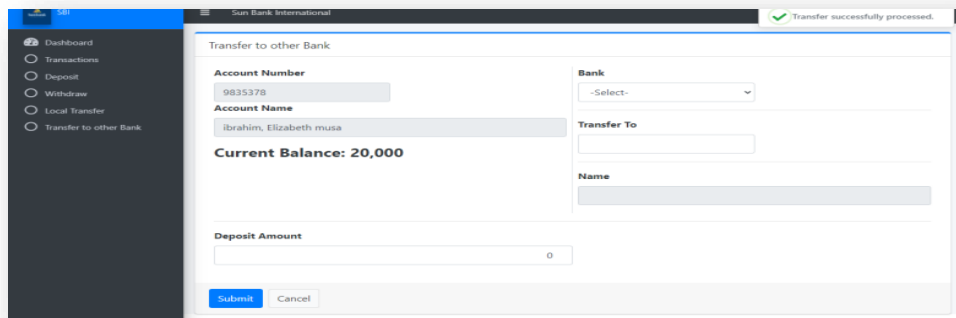


Figure 13: Transaction successfully processed

But in the case whereby the transaction was not successful, then the committal bridge swing into action by sniffing and keeping a log of all interbank activities, and resolving all failed transactions. To know if the above scenario was successful or not, the bridge popped into action as shown in figure 14. The framework for smart transaction committal and database resolution as stated in the objective of this paper is the process by which the bridge sniff transaction from the existing system originating bank, every transaction initiated must be sniffed by the bridge so that the bridge will be on the know of what is happening within a transaction. The bridge can have many databases as possible depending on how many banks are connected to it but for this thesis, the bridge has only two bank databases (cyber bank of Africa and sun bank international). To know the status of the transaction that was

initiated In figure 12, the different bank database consoles on the bridge have to be checked as shown in figure 15, at the recipient bank, the transaction was not successful, Neither was the transaction successful at the sun bank international of the bridge as shown in figure 16 whereas the Originating bank database has already recorded the transaction to be successful.

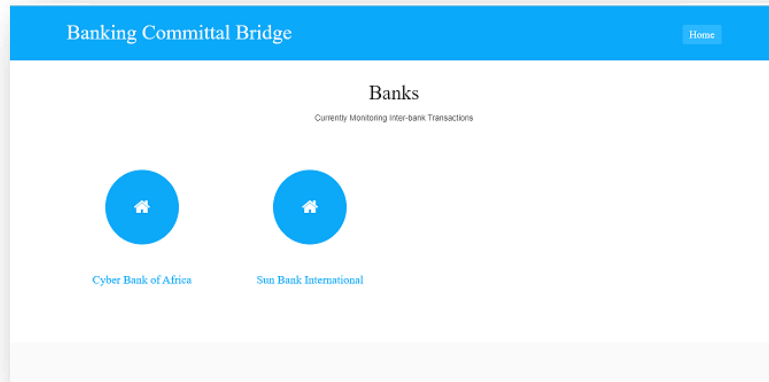
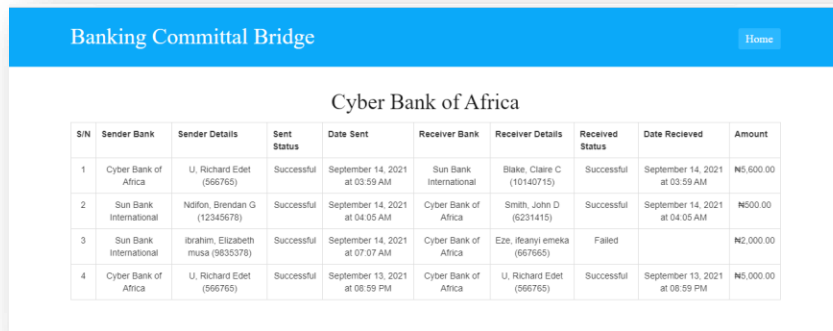
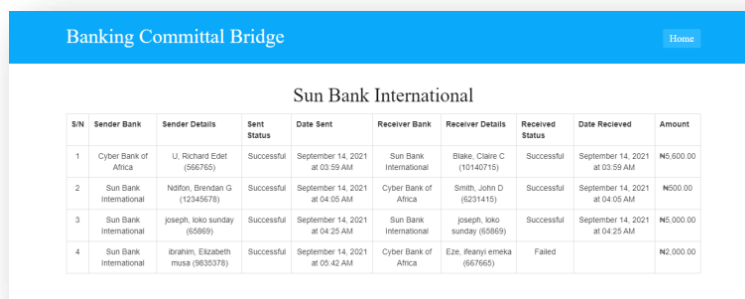


Figure 14: the bridge



| S/N | Sender Bank            | Sender Details                    | Sent Status | Date Sent                      | Receiver Bank          | Receiver Details            | Received Status | Date Received                  | Amount    |
|-----|------------------------|-----------------------------------|-------------|--------------------------------|------------------------|-----------------------------|-----------------|--------------------------------|-----------|
| 1   | Cyber Bank of Africa   | U, Richard Edet (566765)          | Successful  | September 14, 2021 at 03:59 AM | Sun Bank International | Blake, Claire C (10140715)  | Successful      | September 14, 2021 at 03:59 AM | N5,600.00 |
| 2   | Sun Bank International | Ndilon, Brendan G (12345678)      | Successful  | September 14, 2021 at 04:05 AM | Cyber Bank of Africa   | Smith, John D (6231415)     | Successful      | September 14, 2021 at 04:05 AM | N500.00   |
| 3   | Sun Bank International | Ibrahim, Elizabeth musa (9830378) | Successful  | September 14, 2021 at 07:07 AM | Cyber Bank of Africa   | Eze, Ifeanyi emeka (667665) | Failed          |                                | N2,000.00 |
| 4   | Cyber Bank of Africa   | U, Richard Edet (566765)          | Successful  | September 13, 2021 at 08:59 PM | Cyber Bank of Africa   | U, Richard Edet (566765)    | Successful      | September 13, 2021 at 08:59 PM | N5,000.00 |

Figure 15: the Bridge database of Cyber bank of Africa (failed transaction)



| S/N | Sender Bank            | Sender Details                    | Sent Status | Date Sent                      | Receiver Bank          | Receiver Details            | Received Status | Date Received                  | Amount    |
|-----|------------------------|-----------------------------------|-------------|--------------------------------|------------------------|-----------------------------|-----------------|--------------------------------|-----------|
| 1   | Cyber Bank of Africa   | U, Richard Edet (566765)          | Successful  | September 14, 2021 at 03:59 AM | Sun Bank International | Blake, Claire C (10140715)  | Successful      | September 14, 2021 at 03:59 AM | N5,600.00 |
| 2   | Sun Bank International | Ndilon, Brendan G (12345678)      | Successful  | September 14, 2021 at 04:05 AM | Cyber Bank of Africa   | Smith, John D (6231415)     | Successful      | September 14, 2021 at 04:05 AM | N500.00   |
| 3   | Sun Bank International | Joseph, Iko Sunday (65669)        | Successful  | September 14, 2021 at 04:25 AM | Sun Bank International | Joseph, Iko Sunday (65669)  | Successful      | September 14, 2021 at 04:25 AM | N5,000.00 |
| 4   | Sun Bank International | Ibrahim, Elizabeth musa (9830378) | Successful  | September 14, 2021 at 05:42 AM | Cyber Bank of Africa   | Eze, Ifeanyi emeka (667665) | Failed          |                                | N2,000.00 |

Figure 16: the Bridge database of sun bank international (failed transaction)

At this point, the implementation of a controller bridge that resolves the transactional failure which is one of the objectives of this study will be initiated to check for errors as shown in Figure 17. As the system is trying to resolve the error, it's reactivating the



failed transaction and keeps a time count of 3min, if it failed again, it reactivates it the second time and still maintains the same time count as shown in figure 18

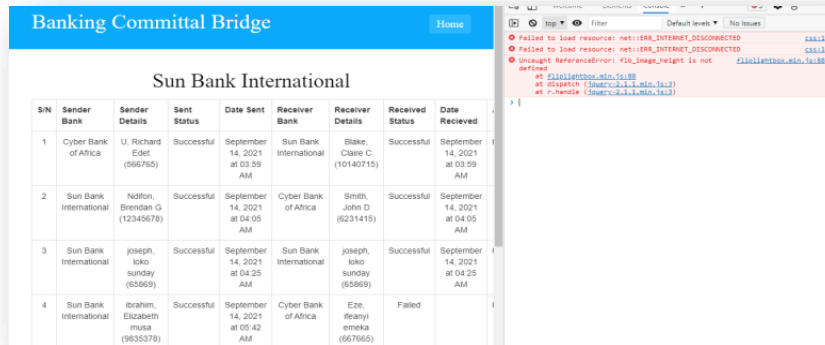


Figure 17: Initiate error checker

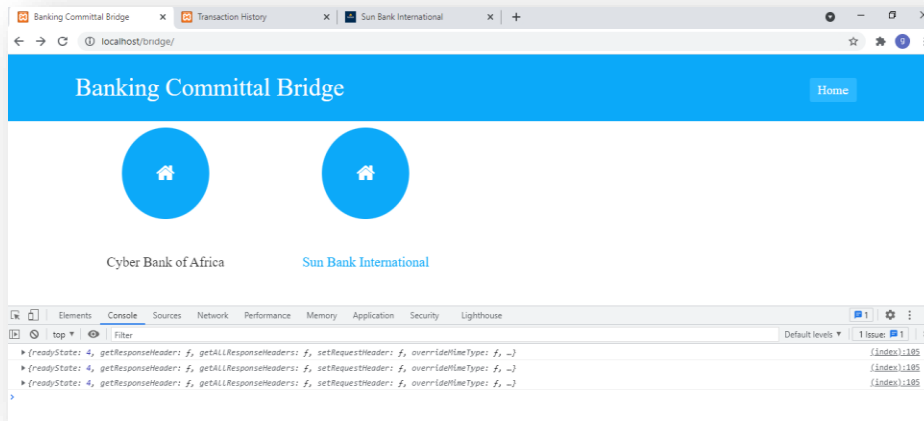


Figure 18: Activate the failed transaction

If at the third trial, the system automatically reverses the fund to the originating bank (sun bank international) as shown in figure 19 but if the fault in the network is ratified before the second reactivation, then the fund will be forwarded to the receiver ( Eze Ifeanyi Emeka). The bridge was tested by simulation using sample data.

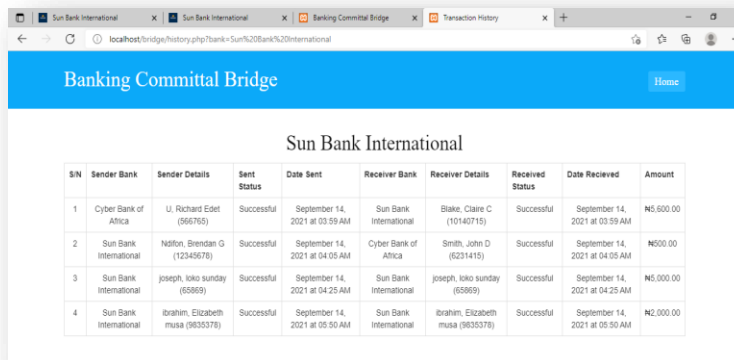


Figure 19: Reversal complete

## V. Conclusions

The proposed committal bridge serves as an intermediary between the bank server and mobile banking network that sniffs and resolves all failed transactional non-committals. At the end of the research work, A controller bridge was developed to mirror or sniff mobile transactions from bank to bank; internally (bank A to A) or externally (bank A to B), ascertain the success (full committals) of the transactions or initiate a resolution if the transaction failed. The software bridge was tested by simulation using mockup data. The results in time concerning sniffing, confirmation, and resolution were outstanding. Implying that the bridge can improve transactional resolutions faster than ever.

### 5.1 Recommendation for future research

Based on the limitations inherent in this study, the researcher will strongly recommend that:

- (1.) A real data-based study should be conducted in the study area to help validate the workability of the new System.
- (2.) Banks' Research and Evaluation units should carry out an assessment study on the system to seriously investigate the possible effect if cooperated.

## Reference

1. Essien Eyo (2011). Proposed algorithm for automated teller machine. Elixir International Journal, Elixir Adoc Network 36 (2011) 3164-3167.
2. Garuba. A. O. (2008). Fundamentals of Banking, Vol.1 Benin Ambik Press Ltd. Ikhilae, E. (2010). Bank PHB Opposes CBN's Move to Free Self from ATM Fraud Suit. The Nation, July 5th P25.
3. Madawaki Mohammed, Mohammed AlhajiAudu and Alexander Solomon Oghoyone (Nov. 2014). "The Effects of Customers Experience on ATM Refund System for Failed Bank Transactions: A Study of Deposit Money Banks in Maiduguri, Borno State, Nigeria".
4. Ndifon, E. and Okpa, I. (2014). Challenges and benefits of the cash-less policy Implementation in the Nigerian economy. European Journal of Business and Management, 6(26): 24-32.
5. Ogban, F.U and Asagba, P (2011). Issues on concurrency controls in a transactional database system: The automatic teller machine (ATM) problem. international Journal of Natural and Applied Sciences (ijnas), vol. 6, nos.1& 2; p. 63 – 71
6. Osazevbaru, H. O. and Yomere, G. O. (2015). Benefits and challenges of Nigeria's cashless policy. Kuwait Chapter of Arabian Journal of Business and Management Review, 4(9): 1-10.
7. Sriramya P, R.A. Karthika(2020). Android App For Atm Refund System For Failed Banking Transactions, Journal of Critical Reviews ISSN- 2394-5125 Vol 7, Issue 9, 2020.
8. <http://www.knowreserve.com/2021/08/what-is-failedatm-transaction.html>