Global Scientific JOURNALS

GSJ: Volume 9, Issue 6, June 2021, Online: ISSN 2320-9186
www.globalscientificjournal.com

A NOVEL MODEL FOR IDENTIFYING AND COLLATING FRAGMENTS IN DISTRIBUTED SYSTEM

EZEUGBOR I. C[1], ONYESOLU M. O[1], MBELEDOGU N. N[1], OKOLO C. C[2]

[1] Department of Computer Science, Nnamdi Azikiwe University, Awka
[2] Electronic Development Institute, Fed Min of Sci and Tech, Awka Capital Territory

ic.ezeugbor@unizik.edu.ng, +2348035287698

ABSTRACT

The problems of complexity, maintenance of query processing and performance according to the request of end users from different strategic areas have called for the development of a novel model for identifying and collating fragments in distributed system; considering the fact that companies and other establishments adopted and are using distributed system for their daily business transactions in so many locations. Algorithms clustering and fragmentation were adopted in identifying database fragments to individual sites and collating these distributed systems to one central store. The methodology used to achieve this study and in design of the system is object oriented analysis and design methodology. In the implementation of the efficient system, create, read, update and delete (CRUD) techniques, bootstrap, cascading style sheet (CSS) and javascript were equally used. The database of the system became workable with the adoption of mongodb database engine. The end result projects a system with external fragmented database capable of saving to a central database. Every individual, industries and organizations that need to fragment database in distributed system can adopt the use of this system for maximum reduction of process time, for complexity reduction and for easy access of data.

Keywords: Novel Model, Fragment, Collating, Distributed System, Clustering Techniques

INTRODUCTION

This study focuses on the introduction of fragmentation concept in the distributed system and to enable the placement of data in close proximity to its place of use, which helps to reduce transmission cost and also the size of the relations that are involved, adding the effectiveness and data integrity of the system. A distributed system has some techniques that can be used to enhance the performance and utility of that distributed database. Of importance to the study is the fragmentation technique. Fragmentation is the division of a single database into two or more prices such that the combination of the prices yields the original database without any loss of information. Each resulting piece is known as database fragment. In fragmentation, each global relation is partitioned into set of fragment relations. However, allocation focuses on the (possibly replicated) distribution of these logical relations across the distributed system's sites. Fragmentation is a design technique used to design a single database into two or more partitions in a way that the partition's combination will yield the original database without any form of insufficiency or addition of information. This reduces the amount of irrelevant data accessed by the application, thus reducing the number of disk accesses. The result of the fragmentation process is a set of fragments defined by a fragmentation schema. The purpose of fragmentation design is to determine non-overlapping fragments which could be the logical unit of allocation. Similarly, fragmentation is the partitioning of database into two or more segments such that each segment represents what the other segment represents, also the combination of the segmented partitions stands as original database without any loss of data. Fragmentation can reduce irrelevant data accesses and increase data local availability. If there is a relation on which many application views are defined at different sites, storing a given relation at one site will result in an unnecessarily high volume of remote data accesses. Storing a given relation at different sites will cause problems in executing updates and may not be desirable if storage is limited. The decomposition of a relation into fragments permits many transactions to be executed concurrently and results in the parallel execution of a single query by dividing it into a set of sub-queries that operate on fragments. Clustering technique comes into play by first of all clustering the number of banks to be used after which the activities to be performed by the banks are also clustered. Then finally, CRUD technique will now be applied for constant check of the system by going into sleeping mood immediately the user's attention leaves the system.

REVIEW OF RELATED WORKS

A combination of authors opined that derived horizontal fragments of a class are generated according to primary fragments of its subclasses, its complex attributes (contained classes), and/or its complex methods. Heuristics are proposed to choose the most appropriate primary fragment to merge with each derived fragment of the member class. At least, derived fragments are merged with a primary fragment that has the highest affinity with it. However, this approach leads to overlaps between resulting derived fragments. Inheritance links are considered in the process of horizontal fragmentation. It is assumed that a pointer is contained in an instance of a storage structure for a class in the class hierarchy. However, storage structures are not implemented in most object DBMS products. How to choose owner class for a member class if the member class have several owner classes that is not mentioned. Further, it is not clear how to calculate affinities between each pair of primary fragment and derived fragment of the same class. Furthermore, there is no evaluation of the proposed algorithms regarding how it will improve the system performance. Some authors came together and stated that derived horizontal fragmentation of each member class is performed according to its owner class in frag(owner, member) list, which is based on the owner-member classification. Derived horizontal fragmentation is implemented with a semi join on the attribute used by the most frequent navigation operations from the member class to the owner class. However, it is not how to decide the owner classes to be used for fragmentation. The resulting distributed database schema is analyzed to show improvements in the system performance. However, the analysis neither considers queries as distributed nor uses any cost models. Moreover, in practice, objects or object variables are often accessed by a big number of queries. There is no algorithm presented to incorporate query information to achieve proper fragmentation schemata. Hoffer and Severance (2013) proposed an affinity-based vertical fragmentation and used Bond Energy Algorithm to cluster attributes according to the affinities between attributes. Since then the affinity measure has been widely used for solving the problem of fragmentation. This approach minimizes the number of fragments accessed by transactions while considering storage cost factors involved in storing the fragments. These among many other works were reviewed.

MATERIALS AND METHODS

For the purpose of providing high performance, high availability and automatic scaling, mongodb database engine was used in implementing a data store. Jquery as seen as a light weight, “write less, do more”, was introduced to enable javascript to be easily used at the sites. Locally installation of packages into this project, specifically into the Node_modeules folder, Npm, node package manager was used. The use of nodejs, as built on chrome’s javascript runtime enables building of fast and scalable network applications. Javascript was included to this work for the following reasons; loading of content into a document whenever the user requires it without reloading their entire page, its use as a client side scripting language, its dynamic nature, it adds special effects on pages like rollover and different types of graphics. Inclusion of colors, layouts and fonts was done with the aid of cascading style sheet (CSS). The framework of this system was designed with the help of bootsrap to help design the sites faster and easier. The framework includes HTML and CSS for design of forms, buttons, tables, navigations, modals. It also gave support for javascript plugins. To overcome the problems encountered in the previous techniques, like complexity, inefficient solutions, clustering algorithm (Algorithm_Clustering) with CRUD (create, read, update and delete) techniques were employed. Sites that have similar and comparable functions were clustered. Similar performance and activities performed were equally clustered. CRUD was then adopted for constant check and guard of the system against unauthorized intrusion. The system is set to improve data security, accessibility, speed of operation and data integrity of a distributed system.

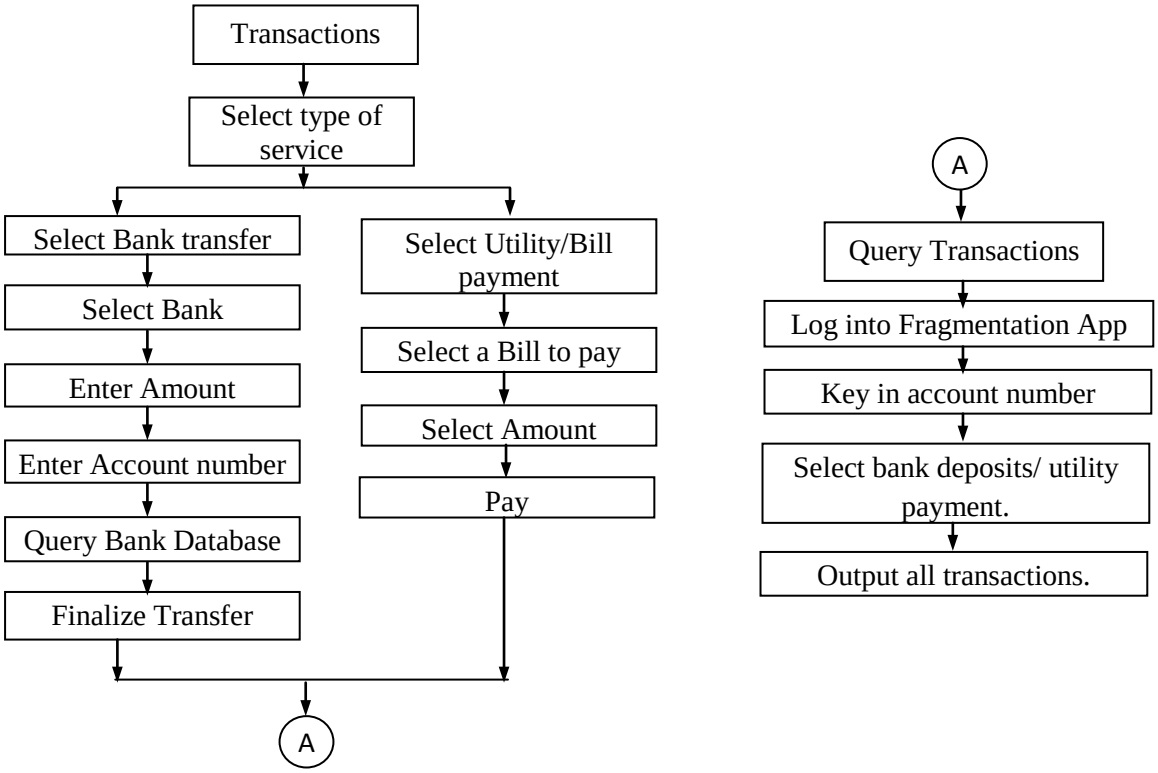

Fig 1: Information Flow Diagram

Design Specification and Algorithms

To solve the problem of taking proper fragmentation decision at the initial stage of a distributed database, clustering algorithm (Algorithm_Clustering), and create, read, update and delete (CRUD) technique were adopted as depicted in Fig 3.

Algorithm_Clustering

Step 1: Set 1 to i
Step 2: Do steps (3 - 12) until i > NS
Step 3: Set 1 to j
Set 0 to k: Set 0 to Sum
Set 0 to Average:Set 0 to clusters matrix CM
Step 4: Do steps (5 - 10) until j > NS
Step 5: If $i \neq j$ AND $CC(S_i, S_j) <= CR$,
go to step (6)
Else,
go to step (7)
Step 6: Set 1 to the $CM(S_i, S_j)$ and $CM(S_j, S_i)$ in the clusters matrix
Add $CC(S_i, S_j)$ to sum Add 1 to k
Go to step 8
Step 7: Set 0 to the $CM(S_i, S_j)$ and $CM(S_j, S_i)$ in the clusters matrix
Step 8: End IF
Step 9: Add 1 to j
Step 10: Loop
Step 11: Average = Sum/k Average(i) = Average Add 1 to i
Step12: Loop
Step 13: Set 1 to m
Step 14: Do steps (15 - 36) until m > NS
Step 15: Set 1 to q Set 0 to Minaverage Set 0 to Minrow
Step 16: Do steps (17 - 20) until q > NS or Minaverage >0
Step 17: If Average(q) > 0 Then
Minaverage = Average(q)
Else

Go to Step 18
Step 18: End If
Step 19: Add 1 to q
Step 20: Loop
Step 21: If Minaverage = 0 Then
Set site number to a new cluster
Else
Go to Step 22
Step 22: End If
Step 23: Set 1 to p
Step 24: Do steps (25 - 28) until p > NS
Step 25: If Average(p) > 0 AND Average(p) < Minaverage Then
Minaverage = Average(p)
Minrow = p
Step 26: End IF
Step 27: Add 1 to p
Step 28: Loop
Step 29: Set 1 to a
Step 30: Do steps (31 - 34) until a > NS
Step 31: If CM(S_{minrow}, S_a) = 1 Then
Set 1 to CSM(S_{minrow}, S_a)
CM(S_{minrow}, S_a) = 0 Step 32:
End IF
Step 33: Add 1 to a
Step 34: Loop
Step 35: Add 1 to m
Step 36: Loop
Step37:Stop

CR: Clustering Range
NS: Number of sites in the distributed database system network
Output: CSM: Clusters Set Matrix
Input: Tmax: number of transactions issued in the database
Fmax: number of the disjoint fragments used for allocation
Cmax: number of clusters in the distributed database system

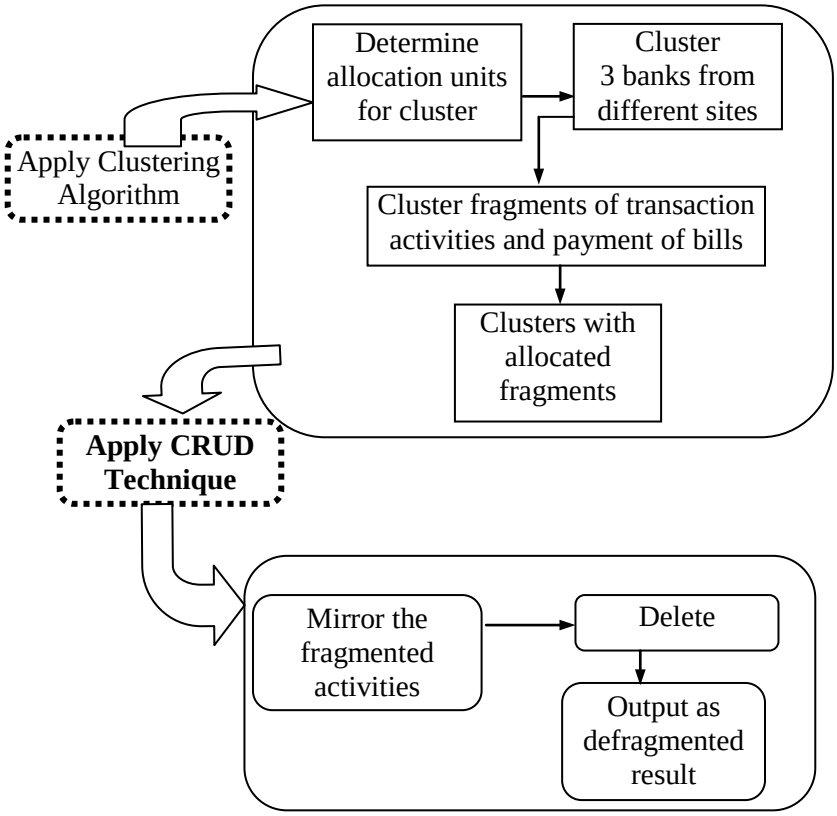

Fig 2: Clusters of Allocated Fragments

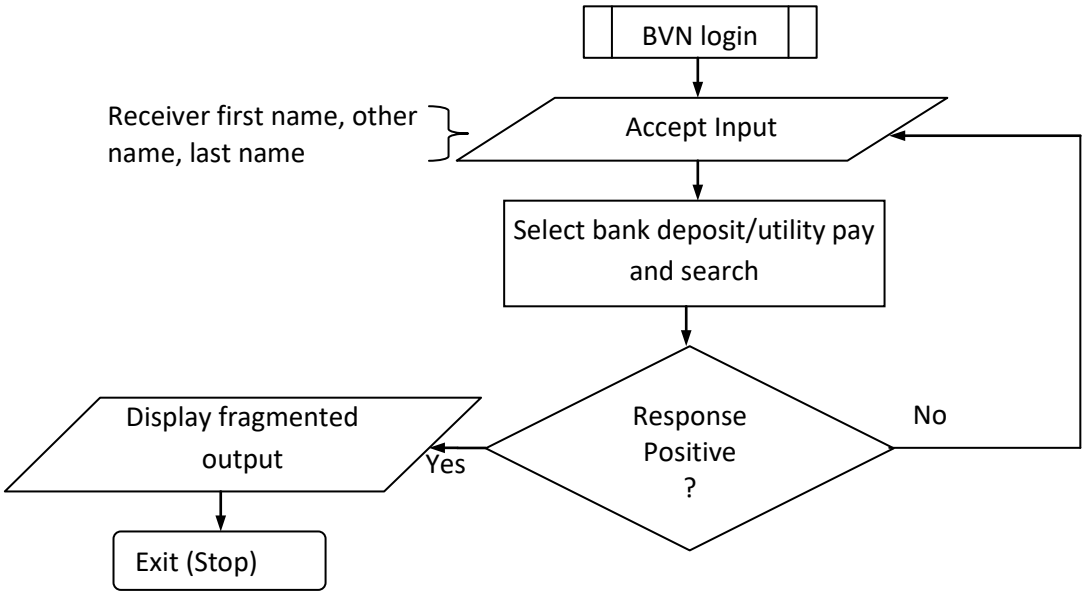

Fig 3: Flowchart on fragmented system

RESULTS AND DISCUSSION

The results of this work are shown in Figure 4 to figure 9. In fig. 4, all transactions made by an individual in all the banks he is registered with, through the help of BVN are brought together and closer for easy view. It also shows the beneficiary bank, transaction mode, date and time, amount, sender and receiver's account number on the fragmented database. The adoption of clustering technique as a model in this work supports and promotes efficiency of the new system such that data are properly arranged on a row, also displaying the transactions made according to the order of transfer. The benefits of this new model to the users are reduction in communication cost by reducing the number of servers that are operational, easy and fast retrieval of data, easy in identifying the exact data in search of and closeness of data to the user. This work was achieved by clustering three banks, collating fragments that are in distributed system, which are the transactions already made, arranging them according to how similar they are to each other i.e., transaction of funds different, then payment of utility bills.

Admin Menu

This menu as labeled figure 4 is made up of Open Account, Search Account and Log Out. Open Account directs the admin to the console that enables the admin open an account for a new customer taking the customer's detail from the keyboard, as shown in figure 5; then storing to the database for subsequent use after which an account number is automatically generated for the user by the system; Search Account in the interface is very necessary to switch from one account to another considering the fact that in an average bank, a multiple customer accounts may all be linked to the same database, a search with name (surname first) or account number allows for ease of transaction after the customers detail and statement is retrieved from the database. Other functions provided on this level are update of an existing customers account, view and delete accounts in the event "the customer can choose to close their account". Logout, the admin logs out after the admin is through with whatever job for the day. Every admin has a personal login ID, with a login ID the system keeps record of user's input and how long a user was active on the system interface. In case of any suspected theft of wrong input of data entered into the system, the user can be held responsible with the help of the user's ID, although there is a counter security measure which in this case is a session timer which times out if system is in sleep mode for a minute.

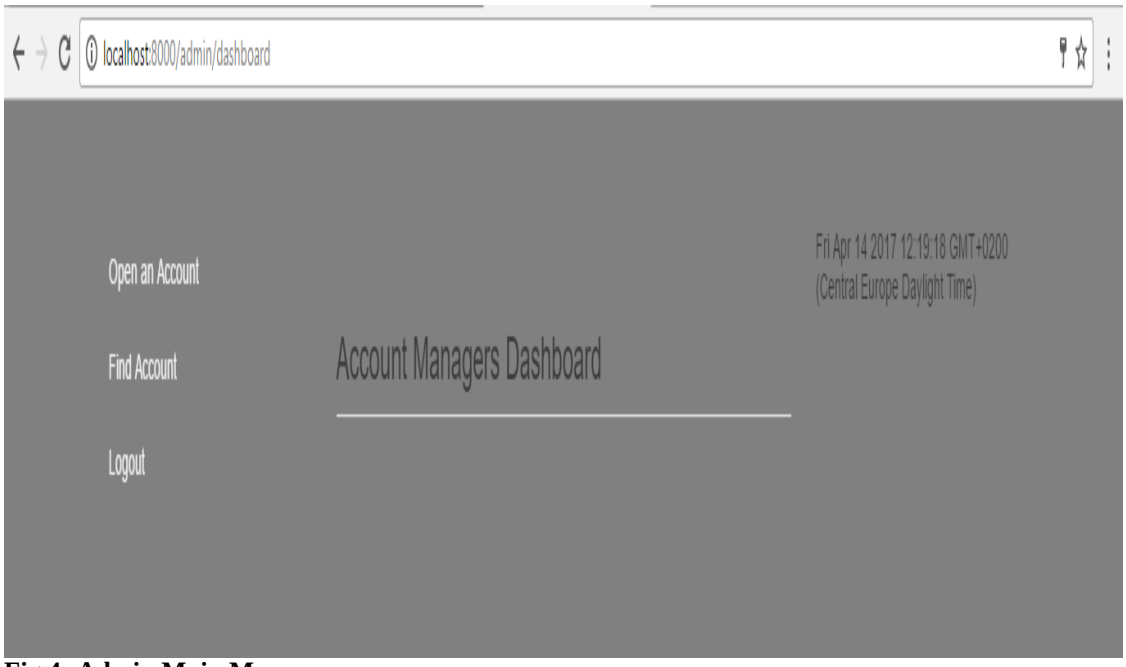

Fig 4: Admin Main Menu

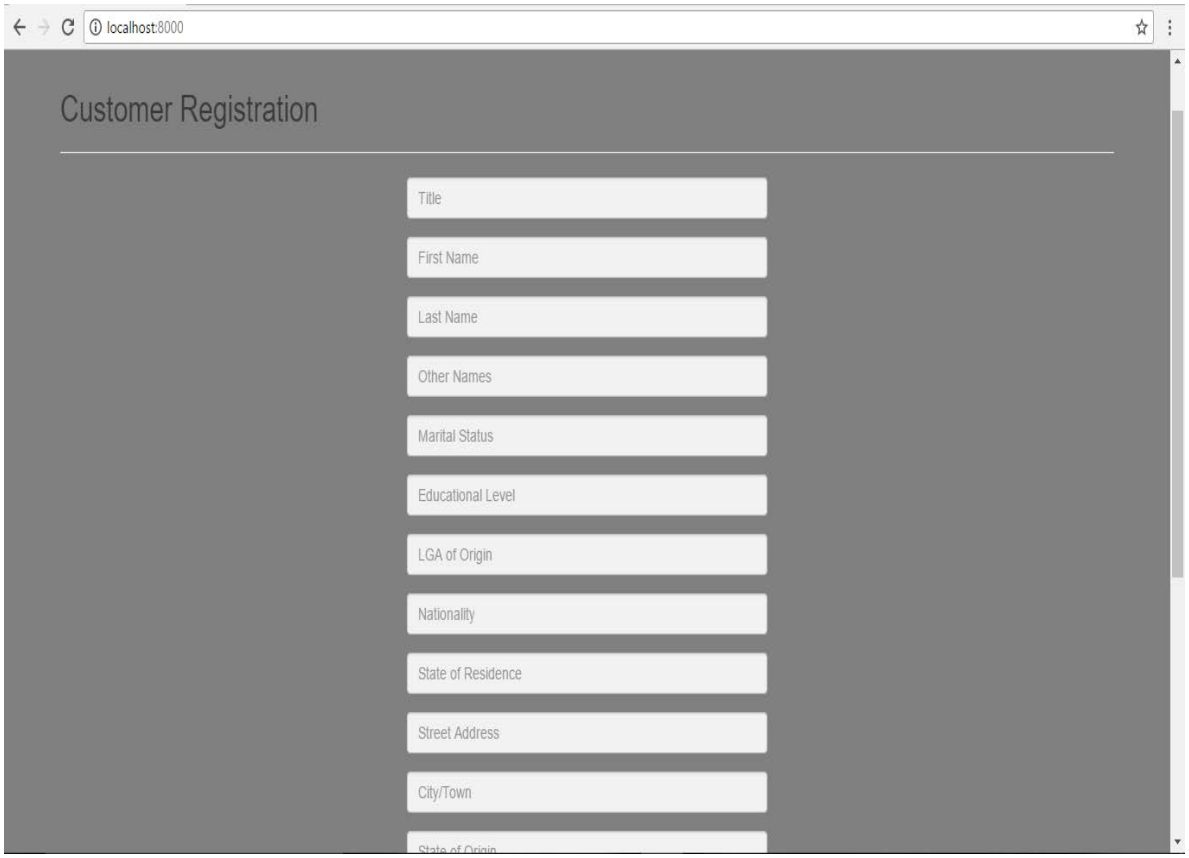

Fig 5: Customer Registration Interface

User Menu

Figure 6 being the user menu is made up of Account Balance, Account Activity, Account Statement, Utility/Bills Payment and Bank Transfer. Account Balance takes a user with the console that contains details of all accounts connected by BVN and the remaining balance in all the accounts displayed. Account Activity enables a user to view the last five activities or transactions. Account Statement displays every transaction made in all the connected accounts and within the selected period of time chosen by the user. The user can either print this statement or download it and can equally save on their personal device for viewing without network access

at their leisure. Utility/Bills Payment as shown in Figure 7 - Here a user is expected to make other transactions aside viewing statement, checking of account balance and so on. This console enables a user the privilege to pay for light bill, water bill and purchase of data or recharge cards with ease from any of the bank accounts connected with the help of BVN fragmentation. Bank Transfer allows a user the opportunity to send funds from any of the user's registered bank accounts to other accounts either to and individual, business partner or family.

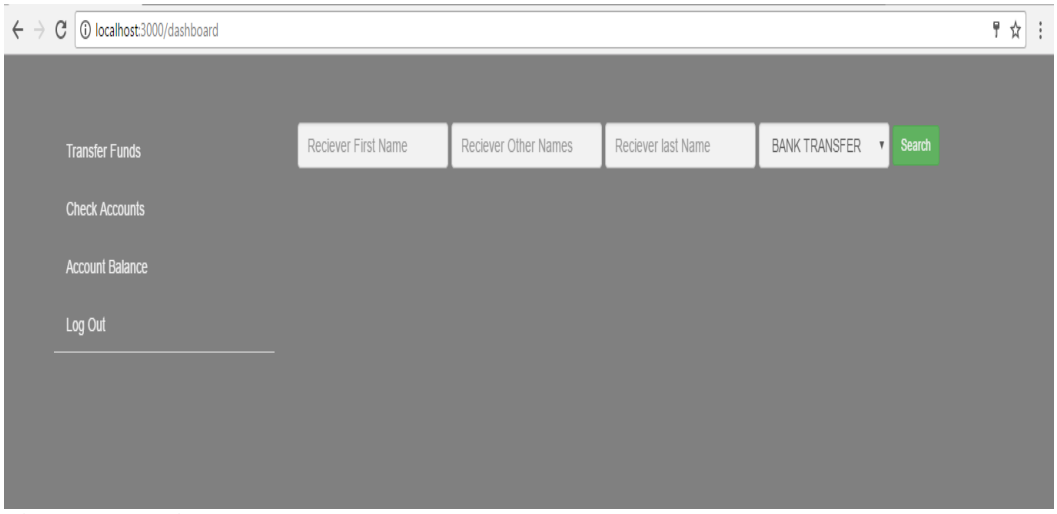

Fig 6: User Main Menu

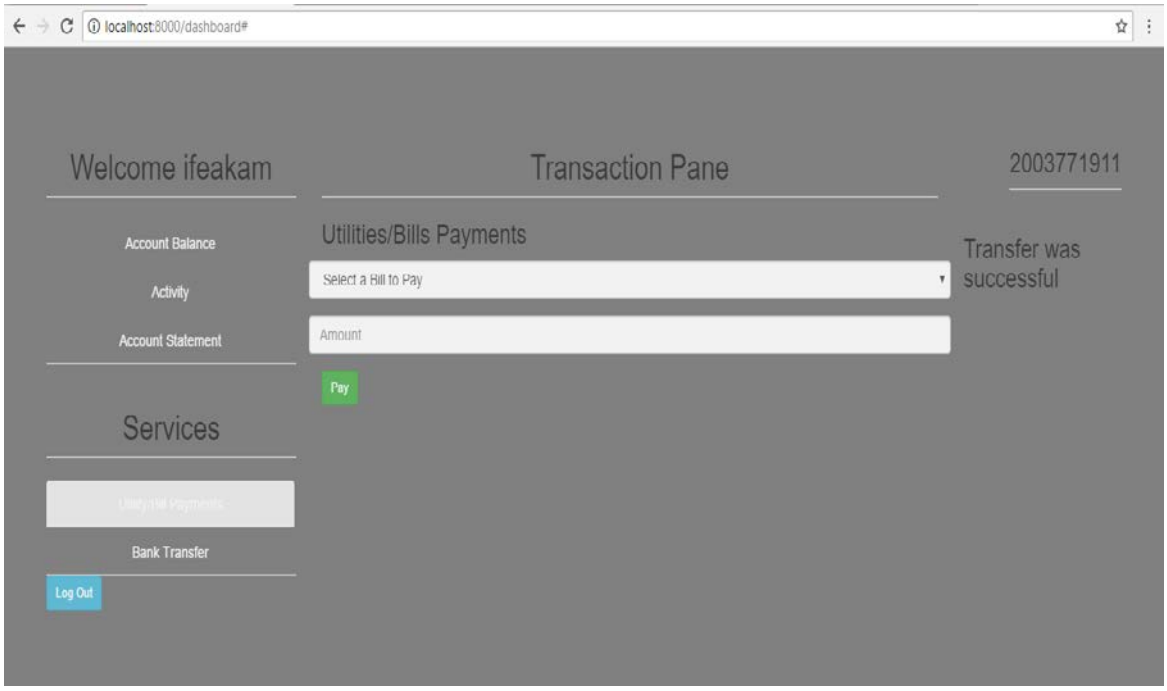

Fig 7: Utility/Bills Payment Interface

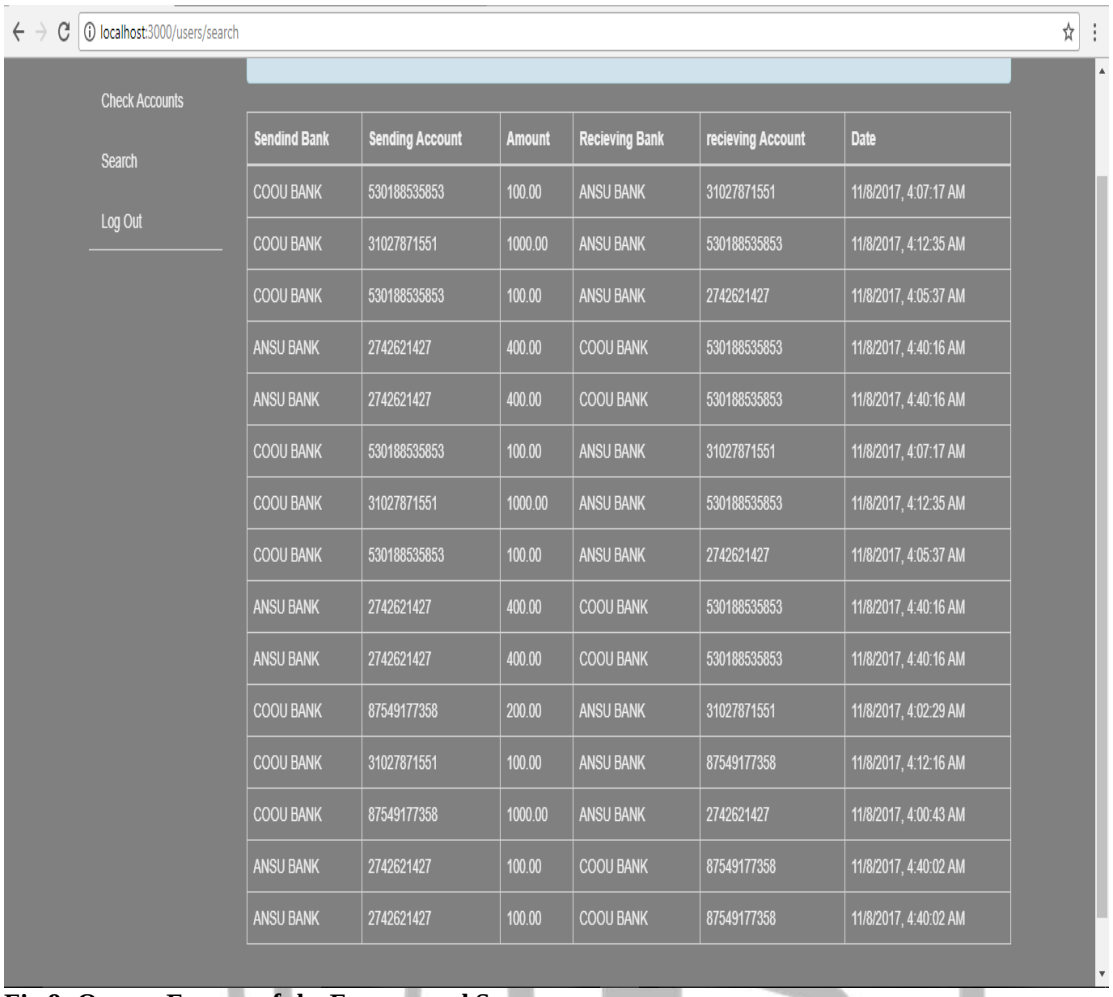

Fig 8: Output Format of the Fragmented System

SUMMARY

A novel model for identifying and collating fragments in distributed system was developed and implemented. The new system was developed to create a central financial technology solution that allows users carry out financial transactions from all their bank accounts using database fragmentation to separate the banks and their operations. It erased the problem of delay in processing data at the banks. It provided a well coordinated system for all the banks for easy access.

CONCLUSION

In this work, a method to promote fragment collation in distributed system was proposed. Two techniques were brought into play; database fragmentation and fragment collation. The development of these techniques in this work is to avoid drawbacks of database fragmentation and data collation like data complexity and redundancy. Data consistency and availability was satisfied to a certain level. The new system was developed to create a financial technology solution that will permit users to carry out financial transactions from all their registered bank accounts with the help of BVN. The result obtained in this work showed that the approach used significantly improved performance requirement satisfaction in distributed system. The effectiveness of this system geared towards reduction of data transfer between sites, improving availability of data, safer transaction and increased security. There should be possible future expansion allowed in a designed system.

REFERENCES

[1] **Bai ao, F., Mattoso and Zaverucha, G. (2004)** *"A distribution design methodology for object DBMS"*,Distributed and parallel databases, springer. 2004. 16. 45-90,2004.

[2] **Gupta, S. and Panda, S. (2012)** *"Vertical fragmentation, allocation and re-fragmentation in distributed object relational database systems-(update queries included)". International Journal of Engineering Research and Development", 4,45-52*

[3] **Eziefe, C.I. and Barker, K. (1995)***"A comprehensive approach to horizontal class fragmentation in a distributed object-based system"*, Distrib. parall. Databases, 3(3): 247-272. 563, 564, 607

[4] **Ozsu, M.T. and Valduriez, P. (1997)***"Distributed and parallel database systems"*. In tucker, A., editor, handbook of Computer science and Engineering, 1093-1111. CRC press. 38.

[5] **Ramakrishnan, R. and Gehrke, J. (2003)***"Database management system"*. McGraw-Hill, 3 edition. 70, 189, 201.

AUTHORS PROFILE

Engr. Ezeugbor Ifeanyi Charlse is a lecturer with the department of computer science, Nnamdi Azikiwe University Awka, Anambra state, Nigeria. He did his first degree in Mechanical Engineering and his second degree was in computer science. He did his master's program in computer science. He is presently offering his PhD in Computer Science. He is a native Aguluezechukwu in Aguata LGA of Anambra state. He is married with 3 kids.