International Journal of Electrical, Electronics and Computers
Vol-7, Issue-4 | Jul-Aug, 2022
Available: https://aipublications.com/ijeec/
Peer-Reviewed Journal

Development of a Web based Shopping Cart using the Mongo DB Database for Human Sustainability

U.C. Onyemauche[1], C.L. Okpalla[1], M. Benson- Emenike[1], B, U. Osundu[2], C.E. Mbanusi[3], C.U. Onyemauche[4]

[1]Department of Computer Science, Federal University of Technology Owerri, Imo State, Nigeria.
[2]Department of Health Information Management, Imo State College of Health Technology and Management Sciences, Amaigbo, Imo State, Nigeria.
[3]Department of Computer Education, Federal College of Education (Technical) Umunze, Anambra State, Nigeria.
[4]Department of Computer Science, Ebonyi State University, Abakaliki, Nigeria.

Received: 07 Jun 2022; Accepted: 05 Jul 2022; Date of Publication: 10 Jul 2022

***Abstract*— *The databases in use today are of SQL-type. This has its drawbacks such as unnecessary complex queries, rigid schema, non-asynchronous persistence and they are definitely not object oriented. Moreover, SQL-shopping cart is expensive by requiring more programs to function. Therefore, the development of a modern shopping cart using MongoDB will eradicate these set backs. The main aim of this study is to design and execute a modern e-commerce shopping cart using MongoDB database. The method used here is the agile development methodology. Stages involved here include: Brainstorm, Design, development stage, Quality Assurance, deployment and Cycle. The User interface is written with HTML, CSS and JavaScript. The HTML (Hyper Text markup language) is used to create the web pages involved, including the forms through which the user supplies input to the system. Each item in the web page is well labeled to optimize user friendliness. The CSS (cascading Style Sheet) is used to create a mobile-friendly, responsive interface to enable mobile devices to seamlessly use the system.The developed shopping cart will save time and effort for programmers rather than using SQL tools with all the labors with it.*

***Keywords*—Document Oriented, Hyper Text Mark up Language, NoSQL database, Open Source, Shopping Cart**

I. INTRODUCTION

Information technology is fast growing. At each stage, every emerging tool and computing system has to be accompanied by either new technology sets or an update of the existing ones. Such is the case with databases. The databases used in the 70s are of SQL-type. This has its drawbacks such as unnecessary complex queries, rigid schema and non-asynchronous persistence; of course they are definitely not object oriented. In today's world of speed and simplicity, object-oriented methodology has become the order of the day, technologically. It is only fair and rational to migrate to a database which can persist the output of object-oriented technologies in object format. This will save the time and computing resources required to convert from Objects to procedural and back.

The emergence of NoSQL type of database is a solution to this long term problem of computing resources wastage in trying to cache objects from programs with a non-object-oriented database. Following the rate of change of information, one should not expect a 1970 (almost 30 years ago) to be in the forefront of affairs in this revolutionized era. In this modern world, Computing has gone wild and "rascally" that database schemas don't need to be rigid, as different kinds of documents may want to be stored in the database. That's why NoSQLs emerged. These store data in document collections. A document can have any number of contents different from other documents (which is the status quo of today's computing) without affecting the database negatively, unlike SQLs where Rows must follow a rigid structure in their contents or else errors ensure.

ISSN: 2456-2319
https://dx.doi.org/10.22161/eec.74.1

This study is about using a modern NoSQL database called MongoDB in place of the aging SQL database to power an important human activity, called Shopping Cart which buttresses e commerce. Shopping cart allows us to have that convenient of shopping online from home or wherever we are from the comfort of our phones or other computers.

II. RELATED LITERATURE

The first person to use the concept NoSQL was Carlo Strozzi in the year 1998 to forename his frivolous Strozzi NoSQL open source relational database and this negated the typical Structured Query language user interface but still retained the features of a relational database structure. This his concepts; NoSQL Relational DataBase Management System was quite different from the 2009 general perception of NoSQL. Strozzi advocates that, since the most recent NoSQL faction is quite different from the relational sculpt brought together, therefore it is most expedient to be referred to as 'NoREL' rmeaning "not relational". Furthermore (Wikipedia.com) Johan Oskarsson, a developer working with Last.Fm brought again this terminology NoSQL near the beginning of 2009 during the time he organized an occasion to discuss about "open-source distributed, non-relational databases". This forename endeavored to tag the surfacing of an escalating figure of non-relational, dispersed records and accumulated together with unwrap spring replicas of Google's “Bigtable/MapReduce and Amazon's DynamoDB” [4].

From the perspective of Wikipedia.com, NoSQL means “Not Only Structured Query Language”; this stand to accentuate and may strengthen Structured Query Language like-minded Query languages or resemble the databases of SQL also with similar and strong architectural design or may even imply “None SQL” as it proffers a particular method of data storage and retrieval represented quite otherwise from tabular relations exploited in SQL. However, [6], defined NoSQL as a mechanism to database blueprint which encourages depository and inquiring (querying) of stored data outside the traditionalist artifact seen in relational databases. NoSQL databases unlike the relational tabular databases domiciles data inside one data structure as JSON document as such it is termed non relational database blueprint which has no need of a schema. For this flamboyant features, it proffers speedy scalability to come through massive dis - arranged data clones.

NoSQL databases are the better choice for large, unstructured data clones compared with relational databases due to their fastness in data processing and flexibility.

NoSQL is seen as a form of dispersed database, this implies that information can be derived, deposited on varieties of servers, and this can be distant or within. The remarkableness and accessibility of data is guaranteed per adventure part of the accumulated information goes gentry, the remaining data being part of the database will also be running.

Recently, establishments have the dire need to continue to harness and explore massive data at a fantastic speed also with the power to meet up with fast running new online based apps.

Now, we are in the dispensation of hosting and running Apps in the cloud, big data analytic, mobile, web Apps and pervasive computing, NoSQL databases will definitely give the fastness and scalability needed by ensuring it is a desired choice because of its efficiencies via easy of use (Arborian Consulting, 2019).

TABLE 1: NOMENCLATURES OF SQL VS NOSQL

DB Type	Nomenclatures			
SQL	Table	Row	Partition	Join
NoSQL	Collection	Document	Shard	Embedded Document

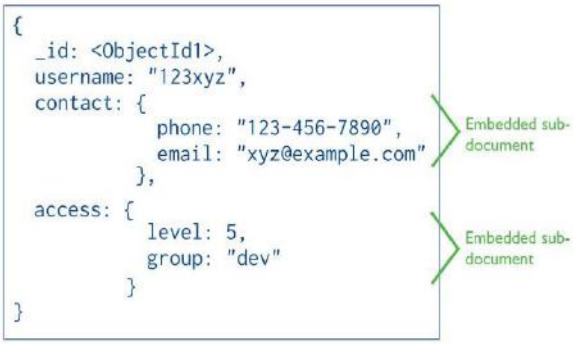
A. 2.1 Document Oriented Model

In contrast to Sequential Query Languages where it is obligatory for you to create a table artefact before adding data, Mango Database agglomerate initially not require their document to have similar structure. This means that document in a mono collection is not expected to have the same set of fields and the type of data for the field cannot be the same crosswise documents inhabited in a collection. Manipulation of data such as adding fields, removing extant fields, updating fields are quite easy here because of flexibility and mapping documents to entities or objects. Matching data fields are seen to be simple. To depict this as it works practically, documents can easily distribute within its space structure that are alike. Validation of document rules during field collection can be enforce without stress [1].

2.2 Document Structure

The main essence of providing an artifact for Mango Database Apps rotates around the artifacts of documents and how application show cases associations within data [5].

Documents that are embedded do get hold of relationships between data by domiciling closely related data in a linear document structure. Therefore, Mango database makes it easy to inter connect documents structure in a mono field or array housed inside a document. These data frameworks that are denormalized permits applications to acquire cum also handle closely concomitant data in a mono database procedure.



```
{
  _id: <ObjectId1>,
  username: "123xyz",
  contact: {
              phone: "123-456-7890",
              email: "xyz@example.com"
           },
  access: {
             level: 5,
             group: "dev"
          }
}
```

Fig . 1: Embedded Documents : Source: (Pirtle & Mitch, 2021).

III. RESULTS AND DISCUSSION

The method used here is agile development methodology. The steps involved include: Brainstorm, Design, Development Stage, Quality Assurance, Deployment and Cycle.

Brainstorm: At the brainstorm stage we meet with Developers and identify their constraints using SQL and how they will gain using MongoDB. We also discuss the best practices to be implemented such as user-friendliness, interface responsiveness, mobile friendliness and cross-platform compatibility.

Design: At design, we produced explanatory Diagrams with unified modelling Language (UML) to show block diagrams, flow charts, and classes.

Development Stage: At this stage, we wrote codes using select programming languages like JavaScript and other web-languages to implement the design.

Quality Assurance: We run the programs to ensure they conform to the design. Where there is a bug, we debug and test again until we achieve near perfection.

Deployment: Here we deploy to host server, having ensured the program works as expected. **The Cycle:** Even though the program runs well, we still brainstorm how to make it better and update it as more technology tools evolves, and cycle continues again.

3.1 Block Diagram of the System

This diagram shows the three-tier architecture implemented in the design of the shopping cart system with MongoDB. They are:

The User Interface or Presentation Layer: This is written with HTML, CSS and JavaScript. This is the part that displays the shopping cart and its associate web pages.

Logic Layer or Application: This part the Server part where the programming Codes that manages the cart and user requests are implemented. Here also controls data flow from/to the database to/from the web pages.

Back end or Database layer: This part is where data stored or retrieved from. It is the MongoDB part of this system.

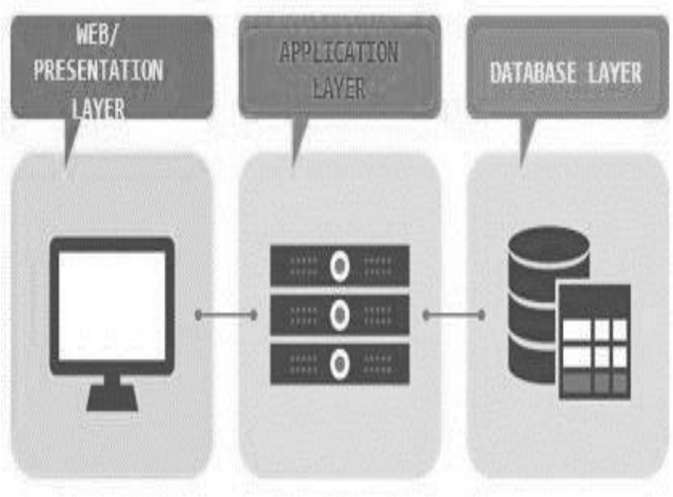


Fig. 2: The three-layer architecture

Use Case Diagram of Proposed System

The use case diagram shows pictorially the various users and what each user is permitted to do in the system. The three users identified are The User: this is previously registered user who comes back to make a purchase by selecting a product, placing order and eventually pays for the products. This is the real customer.

Visitor: This is a new user who just visited the website and is contemplating whether to buy or not. This is not a customer yet.

Admin: This use represents the owner of the website/e-commerce shop. He sees all the customers, their orders and payments.

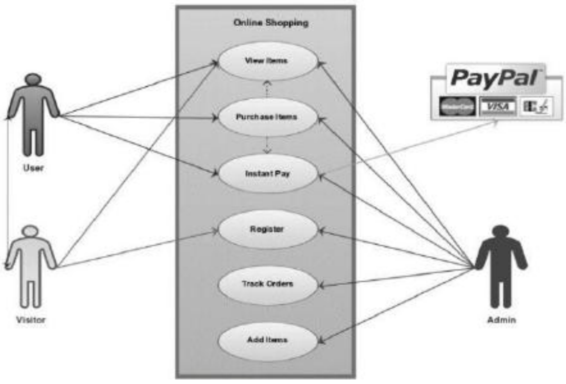


Fig. 3: The Use Case Diagram of the Designed system

3.2 Database Model

The DataBase used here is MongoDB which is document oriented. It does not involve the use of tables and setting up relations rather it revolves around the use of embedded schema. However, an entity relationship diagram was adopted to aid comprehend the internecine workings of the documents. This is depicted in Fig.4.

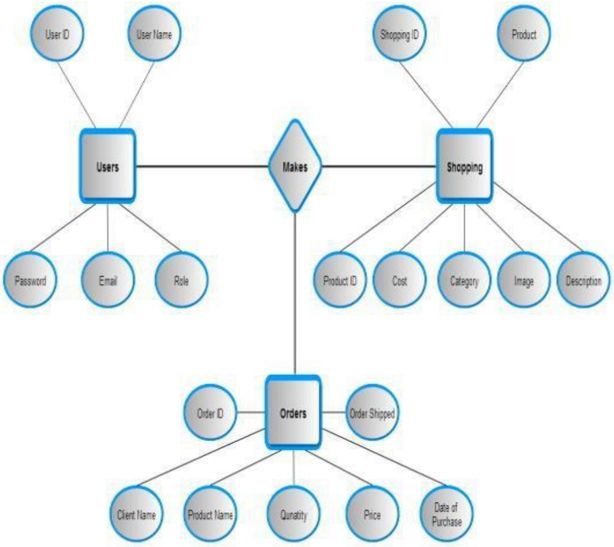


Fig.4: Entity Relationship Diagram of the System

The schema used here was a true representation of what the real e commerce system uses.The collections used here include: cart, user, orders and products.

Product has the following Fields: Id, Quantity, Description, Price

Cart has the following Fields: Id, Product, status, quantity, total

Order has the following Fields: Order1d, customer, quantity, price, date, product.

User has the following Fields: Name, Address, Gender, email, IP address

The structure of these collections is made using the BSON format. Let's use the User collation for an example as shown below:

{

name: 'Mister

Obed', Id: '123', addresses : [name: 'Mister Obed', Id: '123', addresses : [

{ street: '123 Sesame St', city: 'Anytown', State: 'Imo' }

]

Gender: 'Male',

Email: 'Obed@futo.edu',

IPaddress: '102.96.0.98',

}

The add

3.2 The User Interface

The User interface is written with HTML, CSS and JavaScript. The HTML (HyperText markup language) is used to create the web pages involved, including the forms through which the user supplies input to the system. Each item in the web page is well labeled to optimize user friendliness. The CSS (cascading Style Sheet) is used to create a mobile-friendly, responsive interface to enable mobile devices to seamlessly use the system. The Cart Interface pages and buying process is simulated with the images below;

3.3 The Shop / Product Section

This page illustrates a space where customers can pick products needed to be added to the Cart. The user is importuned to check on all the products in order to choose the desired product. Filtering of desired product can be done via categories or price scope. Popping / moving the mouse pointer over a product depicts description and acquirable state. Then Then the Customer will decide whether to purchase or not. See Fig. 5.

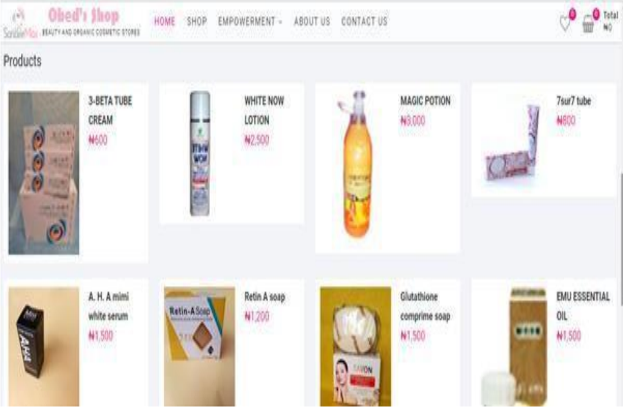
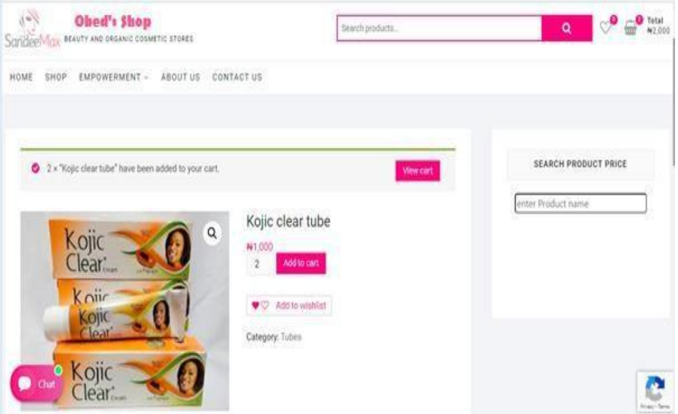


Fig. 5: Product Section

3.4 Cart Interface Page

This section allows Customers to pick all the products they need to purchase and put it to the cart for onward transfer to the Cart. The left side showed all the products the customer have put to the Cart. Right in the Cart page exist the number 2 associated with its price ₦1000; it shows that item number 2 have not been paid for. Customers can still withdraw items they do not want to purchase anymore. See Fig. 6.



2 × "Kojic clear tube" have been added to your cart.

Fig. 6: Cart Interface

IV. CONCLUSIONS

MongoDB, being a document oriented, NoSQL database is more suitable for building e-commerce Shopping Cart than SQL databases. MongoDB is Faster, Flexible and Object oriented. SQL is none of these and is rather older and not a good fit for modern technology, hence the advocacy for its urgent replacement with MongoDB, especially in critical technologies that is permeating into the future. The future is light-weight. The future is fast. This is the jet age.

V. RECOMMENDATIONS

To make a faster e-commerce shopping Cart, the MEAN stack technology should be used. MEAN is an acronym for MongoDB, E and NodeJS. These is purely JavaScript making all activities light weight, optimized, fast and active. This study is recommended to serve as a tool for online business organizations seeking improvements on their existing online shopping carts. Moreover, students and academicians that are in the field of business, computing and information technology can use this as a source of literature and research material.

REFERENCES

[1] A. Aditya, MongoDB: A Giant Database, IEC College of Engineering and Technology, Greater Noida (U.P.) A.K.T.U University, (2019).

[2] Alborian Consulting. "Why MongoDB is the way to go", dzone.com. Archived from the original on June 12, 2018. Accessed July 16, 2021.

[3] Carlo Strozzi,Bridging Relational lightweight NoSQL open-source relational database, 1998. The Definitive Guide (1st ed.), O'Reilly Media, p. 216, ISBN 978-1-4493-8156-1

[4] G. Kenny, . "MongoDB 3.0 Wired Tiger Compression and Performance". object rocket.com/. Retrieved July 28, 2021.

[5] M.Pirtle & K..Mitch, Embedded DB for Web Development (1st ed.), Addison-Wesley Professional, p. 360, ISBN 978-0-321-70533-4. 2021.

[6] K.Scalegrid "Atomicity, isolation & concurrency in MongoDB". scalegrid.io. Retrieved June 5, 2022.