A Project Report

on

No SQL Injection, Vulnerability to detect on Web Application

Submitted in partial fulfillment of the

requirement for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of
Ms. Archana
Assistant Professor
Department of Computer Science and Engineering

Submitted By

BHAGIRATH GUPTA 20SCSE1010073
MAYANK RAJ SINGH 20SCSE1010570
CHANDRAPRATAP SINGH JADON 20SCSE1010030

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
DECEMBER-2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE’S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled **“No SQL Injection, Vulnerability to detect on Web Application”** in partial fulfillment of the requirements for the award of the **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** submitted in the **School of Computing Science and Engineering** of Galgotias University, Greater Noida, is an original work carried out during the period of **August-2021** to **DECEMBER-2021**, under the supervision of **Ms. Archana ,Assistant Professor, Department of Computer Science and Engineering** of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

BHAGIRATH GUPTA 20SCSE1010073
MAYANK RAJ SINGH 20SCSE1010570
CHANDRAPRATAP SINGH JADON 20SCSE1010030

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor Name

(Ms. Archana)

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of BHAGIRATH GUPTA 20SCSE1010073, MAYANK RAJ SINGH 20SCSE1010570, CHANDRAPRATAP SINGH JADON 20SCSE1010030 has been held on________________________and his/her work is recommended for the award of
Bachelor of Technology in Computer Science and Engineering

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date:

Place: Greater Noida

ACKNOWLEDGEMENT

We would like to express my gratitude and appreciation to all those who gave us this opportunity to work on this project and complete this report. Special thanks are due to our guide whose help, stimulating suggestions and critique, and support helped us during the process of fabricating and in writing this report. We sincerely thank her for proofreading and correcting my mistakes when necessary.

We would also like to thank our reviewer and teammates for offering us the necessary advice in areas where we found ourselves lacking and confusing. Without them, it would have been extremely difficult to bring this project to fruition

<u>**Abstract**</u>

NoSQL package is modified from abstract-level down to enhance the synchronous methods supports for the development of a node NoSQL database quickly and using easily. And it makes abstract-NOSQL modularization become possible.

NOSQL database can be extended its capabilities by adding different feature adding. and you (database developer) almost have nothing to do, can have these features. The database user can be free to decide whether to add this feature.

- NOSQL Stream: streamable ability. you need implement it.
- NOSQL Encoding: key/value encoding ability.
- <u>Events</u> : hooked even table ability.

NOSQL Interface is neutral. There is no bias neither synchronous bias nor asynchronous bias. So that more people choose according to their own manner. For myself, I am not very concerned about the performance of JavaScript, I am more concerned about the efficiency of its development, as well as through functional programming (functions, closures such a simple concept) extend out of the rich and wonderful world. I still cannot help but to think about performance issues. Asynchronous itself produces a small gap, because JavaScript reason this gap is magnified.

just saying that the asynchronous and synchronous consideration, if a function is only 1% of the opportunity to visit the IO, most of the time (99%) are in memory access. I want to different considerations, have different choices. And this decision is unlikely that done by the interface instead.

Synchronous operation converts into asynchronous operation is easy, and almost no performance loss, in turn, may not. Conversion is in many ways, set Immediate is not the best, but it is the simplest one.

ES6 generator or node-fibers could be a better way. the coroutine/fiber is lighter and more efficient than thread.

The set Immediate package could be extended to use different implementations (set Immediate, nextTick, ES6 generator, node-fiber) in a different environment. So, the simulated asynchronous uses this way, if you do not implement the asynchronous methods.

List of Table

Table No. 1: Student Data

NAME	ADMISSION NO	SECTION	PROJECT ID
Mayank Raj Singh	20SCSE1010570	1	BT2012
Bhagirath Gupta	20SCSE1010073	1	BT2012
Chandra Pratap Singh Jadon	20SCSE1010030	1	BT2012

Table No. 2: Faculty Data

NAME	DESIGNATION
Ms. Archana	Assistant Professor

Introduction

NoSQL A NoSQL originally referring to non-SQL or non-relational is a database that provides a mechanism for storage and retrieval of data. This data is modeled in means other than the tabular relations used in relational databases. Such databases came into existence in the late 1960s, but did not obtain the NoSQL moniker until a surge of popularity in the early twenty-first century. NoSQL databases are used in real-time web applications and big data and their use are increasing over time. NoSQL systems are also sometimes called Not only SQL to emphasize the fact that they may support SQL-like query languages.
A NoSQL database includes simplicity of design, simpler horizontal scaling to clusters of machines and finer control over availability. The data structures used by NoSQL databases are different from those used by default in relational databases which makes some operations faster in NoSQL. The suitability of a given NoSQL database depends on the problem it should solve. Data structures used by NoSQL databases are sometimes also viewed as more flexible than relational database tables.

Many NoSQL stores compromise consistency in favor of availability, speed and partition tolerance. Barriers to the greater adoption of NoSQL stores include the use of low-level query languages, lack of standardized interfaces, and huge previous investments in existing relational databases. Most NoSQL stores lack true ACID (Atomicity, Consistency, Isolation, Durability) transactions but a few databases, such as MarkLogic, Aerospike, FairCom c-treeACE, Google Spanner (though technically a NewSQL database), Symas LMDB, and OrientDB have made them central to their designs.
Most NoSQL databases offer a concept of eventual consistency in which database changes are propagated to all nodes so queries for data might not return updated data immediately or might result in reading data that is not accurate which is a problem known as stale reads. Also, some NoSQL systems may exhibit lost writes and other forms of data loss. Some NoSQL systems provide concepts such as write-ahead logging to avoid data loss. For distributed transaction processing across multiple

databases, data consistency is an even bigger challenge. This is difficult for both NoSQL and relational databases. Even current relational databases do not allow referential integrity constraints to span databases. There are few systems that maintain both X/Open XA standards and ACID transactions for distributed transaction processing.

Limitations of Relational databases

1. In relational database we need to define structure and schema of data first and then only we can process the data.

2. Relational database systems provides consistency and integrity of data by enforcing **ACID properties** (Atomicity, Consistency, Isolation and Durability). There are some scenarios where this is useful like banking system. However, in most of the other cases these properties are significant performance overhead and can make your database response very slow.

3. Most of the applications store their data in **JSON** format and RDBMS don't provide you a better way of performing operations such as create, insert, update, delete etc on this data. On the other hand, NoSQL store their data in JSON format, which is compatible with most of the today's world application.

What are the advantages of NoSQL

There are several advantages of working with NoSQL databases such as MongoDB and Cassandra. The main advantages are high scalability and high availability.

High scalability: NoSQL database such as MongoDB uses sharding for horizontal scaling. Sharding is partitioning of data and placing it on multiple machines in such a way that the order of the data is preserved. Vertical scaling means adding more resources to the existing machine while horizontal scaling means adding more machines to handle the data. Vertical scaling is not that easy to implement, on the other hand horizontal scaling is easy to implement. Horizontal scaling database examples: MongoDB, Cassandra etc. Because of this feature NoSQL can handle huge amount of data, as the data grows NoSQL scale itself to handle that data in an efficient manner.

High Availability: The auto replication feature in MongoDB makes it highly available because in case of any failure data replicates itself to the previous consistent state.

Types of NoSQL database
Here are the types of NoSQL databases and the name of the databases system that falls in that category. MongoDB falls in the category of NoSQL document-based database.
Key-Value Store: Memcached, Redis, Coherence
Tabular: HBase, Big Table, Accumulo
Document-based: MongoDB, CouchDB, Cloudant

In this project, we are going to use MongoDB as our database

Literature Survey/Project Design

We used a systematic searching procedure to identify all of the available articles that discuss storing and managing data for Big Data situations using NoSQL databases. In our systematic searching procedure, we searched two keywords from the Springer Link and IEEE Xplore digital databases in order to assess the article. Firstly, we used the keyword "NoSQL" to find journal articles published in the English language between years 2011 to 2017. We then used the keyword "Big Data" within obtained set of search results to further narrow the set of analyzed journal article.

For the final preference of articles that applied the NoSQL database system for Big Data management. We used some benchmark to include and exclude articles from the set of articles that were selected through the search of IEEE Explore and Springer Link online databases. To include and exclude articles from the set of articles found through our systematic searching technique, we read the title, abstract, methodology and results of each article. We considered only those articles that were written in English and that used NoSQL Databases. The exclusion criteria were the following: 1) Article that applied NoSQL database other than Big Data management, 2) Article that managed Big Data using other than NoSQL database.

We carefully read and analyzed all of included articles to minutes the key information. We followed a standard data extraction from for the particular analysis of each article. Two of the authors of this study (RA and AK) used our designed standard data extraction form to track the key information and compared them in order to confirm the accuracy of the extracted records. Each article was evaluated for the following key information:
(1) Performance comparison for different NoSQL databases for Big Data processing
(2) Overview of different NoSQL databases and Big Data processing technologies

(3) Database Engine Ranking
(4) Data Models of NoSQL Databases
(5) Transactions on NoSQL databases.

Functionality/Working of Project

The proposed Android Text translation is based on a double objective: (i) Word Transmission and (ii) Script Transfer Mark. For a mobile translator, text libraries of different languages were built using MATLAB, which includes the most frequently used terms in conversations in daily life. The user is asked to enter the text in the android code and click on the language-named button to convert to the text production. Figure 1 demonstrates the system architecture of the mobile translator. This is showed how the system proposed will function.

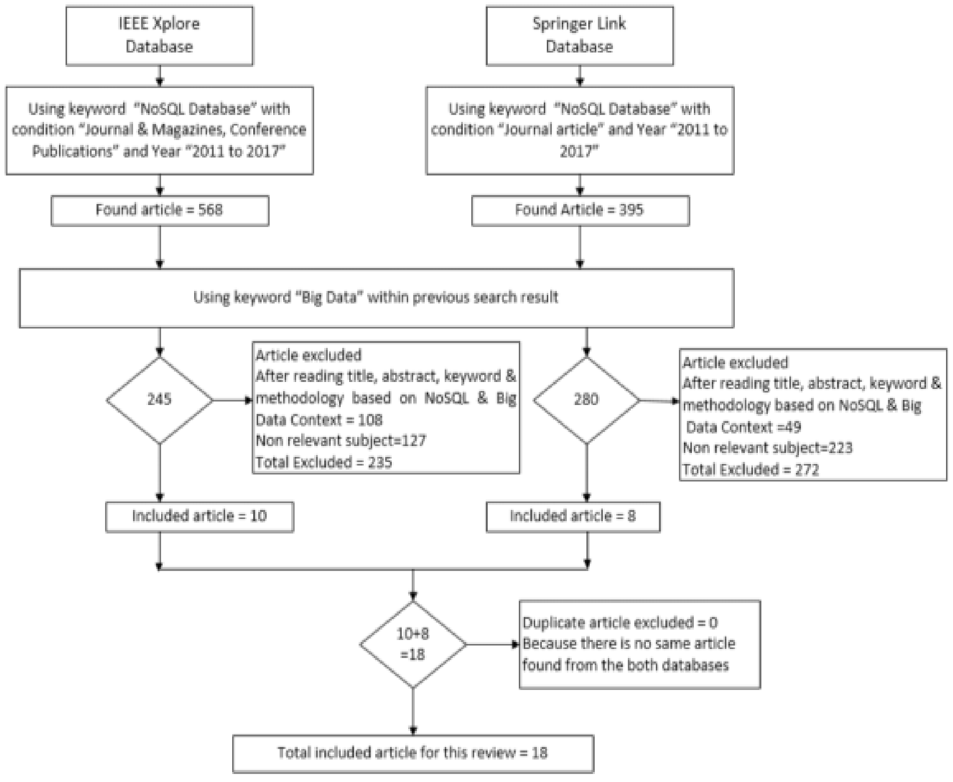
IEEE Xplore Database
Springer Link Database
Using keyword "NoSQL Database" with condition "Journal & Magazines, Conference Publications" and Year "2011 to 2017"
Using keyword "NoSQL Database" with condition "Journal article" and Year "2011 to 2017"
Found article = 568
Found Article = 395
Using keyword "Big Data" within previous search result
245
Article excluded
After reading title, abstract, keyword & methodology based on NoSQL & Big Data Context = 108
Non relevant subject=127
Total Excluded = 235
280
Article excluded
After reading title, abstract, keyword & methodology based on NoSQL & Big Data Context =49
Non relevant subject=223
Total Excluded = 272
Included article = 10
Included article = 8
10+8 =18
Duplicate article excluded = 0
Because there is no same article found from the both databases
Total included article for this review = 18

Figure 1

Characteristics of NoSQL databases

Most of the traditional database systems are based on transactions. These transactional features are also familiar as the ACID (Atomicity, Consistency, Isolation, Durability) [26]. However, Big transactional process does not work properly with the ACID system [27]. Hence, the ACID system has shown to be a problem in different distributed systems that are not fully solvable. Therefore, Eric Brewer [28] introduced the CAP theorem (Figure 1) which is more efficient in different distributed systems. But, later the study [26] noted that the CAP theorem is accepted only two attributes among the three requirements (AP, CP, CA) for Big Data processing at a time. The more details are the following: • Available and Partition-Tolerant (AP): Achieve "eventual consistency" through reiteration and authentication. Example: Voldemort, Couch DB, Cassandra, etc.

• Consistent and Partition-Tolerant (CP): CP systems have trouble with availability whereas keeping data consistent across partitioned nodes. Example: MongoDB, Redis, BigTable etc.

• Consistent and Available (CA): CA systems have problems with partitions and typically deal with replication. Example:

Vertica, MySQL, etc.

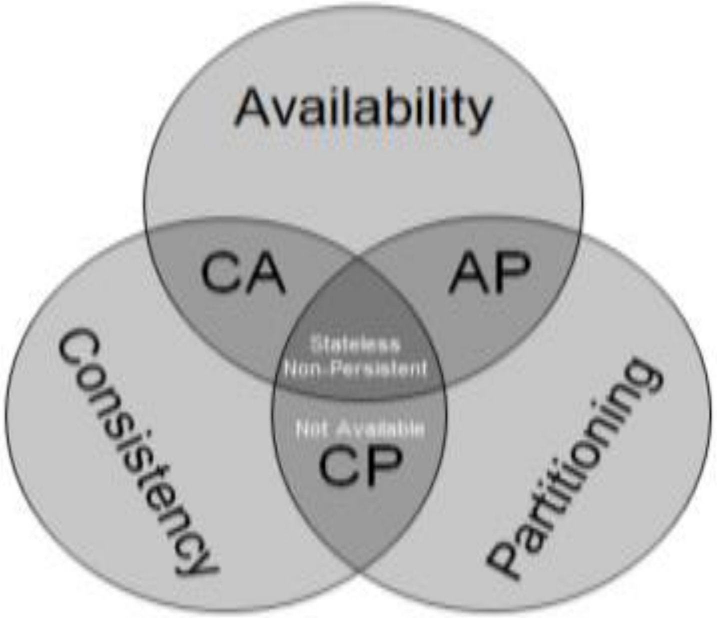

Fig. 2: Illustration of the CAP Theorem {Source: [29]}.

<u>Discussion</u>

We studied particularly the selected 18 articles on "**No SQL Injection, Vulnerability to detect on Web Application**" for the analysis of the present processes to identify future research in the area. The current study presents a summary of the data found in the literature that focused on performance, strengths and weakness of No SQL Injection, Vulnerability to detect on Web Application.

Comparison of NoSQL databases

In this present study, we provide evaluation of four categories NoSQL databases including document based (MongoDB). MongoDB is the most prominent NoSQL databases that are evaluated. MongoDB had the best performance while working with 100% read and 100% blind write. EnqingTang et al. [18] described their experiment for performance measurement between different NoSQL databases including Radis, MongoDB, CouchBase using YCSB and noted that Redis is particularly appropriate for loading and executing workloads for small datasets, but Redis have poor performance for the issues of the vast amount of datasets. The study [8] worked with various NoSQL databases including Redis, MongoDB, CouchBase, and noted that Redis is the best suited for the analysis of the small number of datasets in order to get high performance, and for processing a large amount of data MongoDB is the best choice, whereas CouchBase is better for the fault-tolerant database environment.

Conclusion

In this literature, we presented the NoSQL databases for Big Data processing including its transactional and structural issues. Additionally, we highlight research directions and challenges in relation to Big Data processing. Therefore, we believe that the information contained in this review will incredibly support and guide the progress of Big Data processing.

Reference

[1] N. U. Ahamed, K. Sundaraj, R. B. Ahmad, M. Rahman, and A. Ali, A Framework for the Development of Measurement and Quality Assurance in Software-Based Medical Rehabilitation Systems, Procedia Engineering. 2012; 41: 53–60. https://doi.org/10.1016/j.proeng.2012.07.142.

[2] N. M. Khan, Realtime coaching system, U. S Patent No. 8,279,051, 2012.

[3] A. A. Safaei, Real-time processing of streaming big data, Real-Time System. 2017 53(1):1–44. https://doi.org/10.1007/s11241-016-92570.

[4] InFocus Blog | Dell EMC Services. Big Data and NoSQL: The Problem with Relational Databases. cus.emc.com/april_reeve/big-data-and-nosql-the-problem-with-relational-databases/. Accessed August 7, 2017.

[5] J. Pokorny, New Database Architectures: Steps towards Big Data Processing, Proc. IADIS Eur. Conf. Data Min. (ECDM'13), António Palma dos Reis Ajith P. Abraham Eds., IADIS Press. 2013 3–10.

[6] M. Stonebraker, SQL databases v. NoSQL databases, Commun. ACM. 2010 53(4):10-11. https://doi.org/10.1145/1721654.1721659.

[7] Tech Republic. 10 things you should know about NoSQL databases. http://www.techrepublic.com/blog/10-things/10-things-you-shouldknow-about-nosql-databases/. Accessed August 8, 2017.

[8] J. Bhogal and I. Choksi, Handling Big Data Using NoSQL. in Proceedings - IEEE 29th International Conference on Advanced Information Networking and Applications Workshops, WAINA 2015, 393–398: IEEE

[9] R. Zafar, E. Yafi, M. F. Zuhairi, and H. Dao, Big Data: The NoSQL and RDBMS review, in ICICTM 2016 - Proceedings of the first International Conference on Information and Communication Technology, 2017; 120–126.

[10] I. Chebbi, W. Boulila, and I. R. Farah, Big Data: Concepts, Challenges and Applications, Springer, Cham, 2015 638–647.

[11] S. D. Kuznetsov and a. V. Poskonin, NoSQL data management systems. Program. Comput. Softw. 2014 40(6): 323–332. https://doi.org/10.1134/S0361768814060152.