VFAST Transactions on Software Engineering
http://vfast.org/journals/index.php/VTSE@ 2021, ISSN (e): 2309-3978, ISSN (p): 2411-6246
Volume 9, Number 4, October-December 2021 *pp.: 18-24*

Systematic performance, and Security evaluation of .NET models for accessing database

Attaullah[1], Muhammad Usman[2], Muhammad F. Abrar[3], Najeeb Ullah[3], Ibrar A. Shah[2], Muhammad F. Nadeem[4]

[1]*Deparment of Computer Software Engineering, University of Engineering and Technology, Peshawar, Khyber Pakhtunkhwa, Pakistan*
[2,3]*Department of Computer Software Engineering, University of Engineering and Technology, Mardan, Khyber Pakhtunkhwa, Pakistan*
[4]*Informatics Complex, H-8, Islamabad, 44000, Pakistan*
**Corresponding Author email: usman@uetmardan.edu.pk*

ABSTRACT
In .NET, Object Relational Mapping (ORM) is a programming technique used for accessing the database, which has many frameworks, like Entity Framework, LINQ to SQL, NHibernate, Tele rick Open Access, Light Speed. The LINQ to SQL and Entity Framework usability has increased. This is because of the reason that in these two frameworks full CRUD (Create, Read, Update and Delete) operations can be implemented in short time as compared to Transact Queries, which require more time. In case of multiple projects on various models; Transact Query, LINQ to SQL, and Entity Framework, it becomes difficult to decide which model is the best in terms of performance and security. Therefore, in this article, we provide a comprehensive comparison between Entity Framework, LINQ to SQL and Transact Queries in terms of performance and security. For this purpose, we implemented eleven different types of queries on the selected three frameworks. Subsequently, we quantified and evaluated the execution time and memory usage of all the queries. Furthermore, all types of SQL injection attacks have been applied on three separate applications for security evaluation. Our results show that, the Transact Query is more vulnerable to SQL injection attacks as compared to LINQ to SQL and Entity Framework. Our results show that Transact Query outperforms in terms of memory and CPU usage. Our results also help the practitioner in adopting a framework on the basis of query level performance in terms of memory and CPU usage.

KEYWORDS
Database, dot NET, performance, software security, evaluation

JOURNAL INFO
HISTORY: Received: October 11, 2021 Accepted: December 15, 2021 Published: December 31, 2021

INTRODUCTION

The development of any software system requires previous knowledge of the programming environment in order to decide the best approach for implementation [1]. Therefore, every platform has been evaluated based on performance. The .NET Framework provided by Microsoft is widely used today in the software industry. This platform has provided a robust framework where applications can be conveniently developed due to its user-Friendly environment [2] [3]. The .NET Framework is a complete application development platform, which has been established on open internet protocols and standards. It is observed that for every software, a database is required to store the application data. The .NET Framework uses an MS SQL server database, which is provided by Microsoft [4].Object Relational Mapping (ORM) models provide connectivity for .NET with MS SQL. There are multiple ORM models available to make connectivity with MS SQL. In this research study, we have included Entity Framework, LINQ to SQL, and Transact Query because of their wise use in software industry. For Security evaluation of ORM models, SQL injections are included in this research. The ORM frameworks will be evaluated; either they have the ability to fail the SQL injection attacks, or they are vulnerable to SQL injection attacks.

The Transact Query consumes more time of developers, and their testing is complex. Due to this reason, the majority of the developers shifted to LINQ to SQL. The Entity Framework is the latest framework after LINQ to SQL with extra features. We know that the latest version of the Entity framework has increased the usability; it is easy to build applications on Entity Framework as compared to other models. This research focuses on security and performance in terms of memory or CPU usage, which framework would be suitable. Therefore, a comparative analysis is needed to be carried out that compares various frameworks and evaluates which framework is good or better in terms of security and performance, such as the memory and CPU usage. The outcome of this research will help the practitioners whether to adopt the latest frameworks. This research will also help the practitioners decide which query to use in a specific framework focusing on specific metrics, such as memory and CPU usage.

In the rest of the paper, Section 2 describes the existing literature review. Section 3 discusses research goals; Section 4 explains research methodology, Section 5 provides results and Section 6 provides conclusion of the paper.

RELATED WORK

Many researchers have worked on the performance of PHP Frameworks, while others have compared the MS

This work is licensed under a Creative Commons Attribution 3.0 License.

SQL Server, MYSQL, and ORACLE performance in terms of CPU usage.

Dr. V.sivakumar and Jannali (2021) compared the ORM Frameworks by executing three simple select queries on eight ORM frameworks. They implemented ORM frameworks such as Hibernate, JOOQ, Entity Framework, Nhibernate, CakePHP, Laravel, Django, and SQLAL chemy on four languages Java, C#, PHP, and Python, respectively. The research considered the execution time (ms) as the performance metric. They concluded that the Entity Framework takes more time than other frameworks [5].

Similarly, in another research study, Jackub Arm and zuzana compared the Entity Framework and Nhibernate; they concluded that Entity Framework does not allow to map relationships by strange keys when working with SQL views. Therefore, if the database interface is based on views, Entity Framework is not suitable for it. The research study concludes that both frameworks are appropriate when simple Create Read Update and Delete (CRUD) operations are required, but for a more complex project, their benefits disappear, and shortcomings are emerging [6].

The Abdulkadir Karachi (2009) researched the performance comparison of C# 2013, Delphi Xe6, and Python 3.4 Languages. He compared the structure of the programming language, and he considered Length of Code (LOC), Time of Response (in milliseconds), and Memory usage (Kb) for performance. The author concluded that the Delphi XE6 is considerably faster than C# 2013 and Python 3.4 in terms of response time, while Python is stronger than the other two languages only in terms of code density. Furthermore, Delphi XE6 used 50% less memory than C# 2013 and almost 54% less memory than Python [7].

Yener Sonmez also carried out research on the Performance Comparison of PHP and ASP Web Applications via Database Queries; they carried out their research in two phases, in the first phases PHP and ASP were compared using a script that contains a for loop, the second phase was the comparison of ASP-MSSQL, and PHP-MySQL combinations were compared. Insert and select query statements were used with a single database table using for loop. The authors concluded that when we used PHP-MySQL in a combination, it may perform better than ASP-MSSQL Server Combination in the case of SQL queries. Many PHP developers have concluded that the MySQL Database was reported as the most used database for the web applications [8] [9].

Natela & Merab (2012) carried out C# and F# comparison in 2012, they argue that F# has occupied the Microsoft .NET Framework. The F# Language provided security and good productivity, and the language can be used as scenario language. They compare the time parameter of the calculation process. They construct trees and are considered the primary operation of the trees. They capture the parameters based on Finding elements in Trees, insertion of elements, deletion of elements, and traversing the trees. All the steps were carried out by both languages, C#, and F#. The conclusion was the F# shows priority over C# [10].

Yishan Li and Manoharan carried out a performance comparison of SQL and NoSQL databases study, as they compared the read, write, delete, and instantiate operation on key value stores implemented by NoSQL and SQL database. They concluded that SQL database perform better than NoSQL. Some NoSQL databases were much worse, for each database the performance varies with each operation. Some are slow to instantiate, but fast to read, write and delete. Others are fast to instantiate but slow on the other operations. In addition, there is little correlation between performance and the data model each database uses [11].

Comparative studies of laravel and symfony PHP frameworks were studied by Majida Laazairi et *al.*, and they evaluated the most popular PHP frameworks. They proposed a set of dimensions: features, multilingual, system requirement, technical architecture, organization code, continuous integration, documentation, and learning curve, their result showed that these two PHP-based frameworks are viable options for many projects based on PHP. Moreover, it also provides a full-stack web application development environment for developers. Symfony may be more suitable for larger projects and is considered the most stable PHP framework supported by an extended community. Laravel is the most popular framework for developing the complete stack and has by far the flatter learning curve of all the frameworks [12].

The above researches show that there is a difference in the performance of every framework; they belong either to the same environment or are different. Dr. V.Sivakumar and Jannali used a simple select query and evaluated the models, but there are no results for the create update and delete statements. Similarly, there are no results for join, order, and filters in the query. We decided to evaluate the ORM frameworks of the .NET Environment with eleven types of queries. The queries consist of join, order and filters. The .NET Framework, specifically ORM frameworks that are used for database connectivity, must be evaluated to find out, which performs well. Furthermore, we also added the security aspect of the framework, which will evaluate the Frameworks based on security to find out which model is more secure.

In this study, the selected three frameworks are evaluated in a generalized way from every aspect. Therefore, the outcomes of this study will help the developers identify which framework is good or better in which context. This will also help the developers to decide whether to adopt the latest framework.

RESEARCH GOALS

The study aims to evaluate the ORM Frameworks and Transact Query, finding conclusions for the software developers working on .NET Framework, giving them the conclusion which ORM frameworks must be used in the specific situation. We also identify the limitations of the frameworks. Furthermore, our goal is to evaluate which

framework is more secure and trusted, without any additional security layer that is self-defended in the case of SQL injection. Finally, we summarize the goal of the research using the GQM template (Basili et al. 1994).

Object of The Study	To Evaluate ORM Models
Purpose	To Compare
Focus	Their memory consumption, CPU usage, and security threats
From Point of view	Developers
In the Context	Software development

METHODOLOGY

We select the following 11 different types of queries for evaluation.

i. Transact Query of Insertion (Single Row)
ii. Transact Query of Insertion (Multiple Rows)
iii. Select Query of 5 and 10 Columns
iv. Select Query with Join
v. Select Query with Order
vi. Select Query with Filter (Single Filter, Double Filter, Triple Filter)
vii. Select Query with Filter and Join
viii. Select Query with Filter, Join and Order
ix. Delete Query with Filter
x. Update Query with Filter
xi. Update Query with Filter and Order

For each query evaluation, one breakpoint is kept before the query and one breakpoint after the query. This specifies the space to be considered for performance measurement. Now when the application hits the first breakpoint, a memory snapshot is taken. Similarly, a memory snapshot is taken on the second breakpoint. The two snapshots show that the amount of memory consumed by the system for the selected query. Similarly, the snapshot shows the CPU usage in ms (millisecond), which is counted from the first breakpoint to the second breakpoint. This is the amount of time required to execute the query. So we repeat the process 2 to 3 times and then consider their average value. Similarly, we repeat the memory consumption 2 to 3 times, then take the average value. The memory and CPU analysis is repeated for all the queries in Transact Query, LINQ to SQL, and Entity framework. Fig. 1.describes the flow chart of these steps.

Table 1. SQL Injections Attacks

SQL Injection	User name	Password	SQL Attacks
Tautologies	Kkkkk	‘ or 1=1 --’	Select id from tablename where username=‘**kkkkk**’ and password=‘**‘ or 1=1--’’**
Union Query	Kkkkk	‘union select id from abc where 1=1	Select id from tablename where username=‘**kkkkk**’ and password=‘**‘union select id from abc where 1=1**
Piggybacked Queries	Kkkkk	‘ or 1=1; drop table abc; --	Select id from tablename where username=‘kkkkk’ and password=‘**‘ or 1=1; drop table abc; --**’
Store Procedures	Kkkkk	‘; shutdown –’	Select id from tablename where username=‘**kkkkk**’ and password=‘**‘; shutdown –’’**
Alternate Encoding	Kkkkk	abcdabcd'; exec (char(0x7368757464f776e)) --"	Select id from tablename where username=‘**kkkkk**’ and password=‘**abcdabcd'; exec (char(0x7368757464f776e)) --"**’
Inference	Kkkkk	select pass from usertable where username='user' and 1=0 -- AND pin=0	

For security evaluation of ORM Frameworks, a simple login form with two fields, user name, and password is created. The form allows the user when the correct user name and password are provided. When an incorrect username or password is provided, it prevents the user from the menu. The attacks shown in Table 2. Are applied to the login page, and the attacks are investigated whether they compromised the login page.

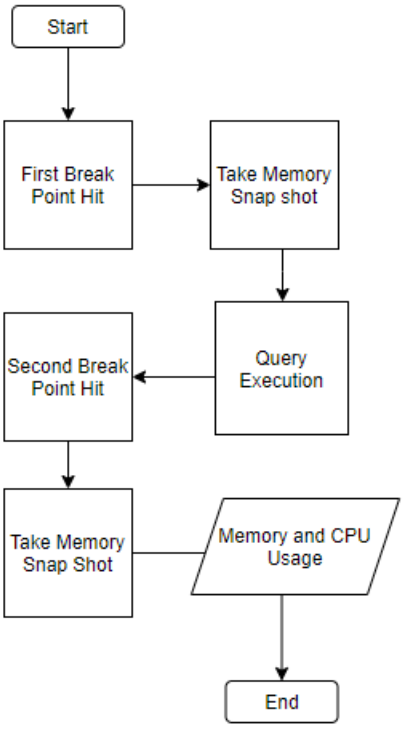

Figure 1 Methodology Steps

RESULTS

As stated above, there are eleven queries in this research, and all the queries have been implemented and executed as explained in Section 3. Here we are going to discuss the results, which have been obtained from the execution of queries. We will start from query one to query eleven, and all the results will be individually discussed, and later conclusion will be derived.

The limitation of this research is that we repeated the methodology steps three times, but the second and third attempt values were near to each other, and the first attempt value is far from the second and third step, so the first step has been ignored. The second and third step average values have been considered for the comparison. The first value is different because when an application is executed for the first time, it needs some external code to run the project for the first time; it affects the CPU and Memory usage.

Table 2.describes memory usage and CPU usage of all the selected 11 queries. We observed that in Single Row Insertion

- By increasing number of columns for insertion, memory usage of the LINQ to SQL increases as compared to the other two frameworks. However, overall Entity Framework consumes more memory. While in CPU usage LINQ to SQL performs well as Compared to the other two frameworks.
- We also observed from the Table 2.that Transact Query consumes less memory among the all while in CPU usage this performs worst. Therefore, if a system is needed, which has more insertion features, and memory is the Top concern, we have to use Transact Query in the application

After experimenting on Single Row Insertion, the experiment on Multiple Rows Insertion was carried out. The results given in the Table 2. Show that

- In Single Row Insertion only LINQ to SQL Framework memory usage increases with the increase in number of columns while in Multiple Rows Insertion with the increase in number of columns all the three frameworks memory usage increase. However, LINQ to SQL consumes more memory as compared to the other two.
- In terms of CPU usage LINQ to SQL performs well in case of less number (i.e. 5) of columns while in case of
  More number of columns (i.e. 10) Transact Query performs well as compared to the other two.
- Overall we observed from the Table 2. That in case of Single Row Insertion Entity Framework performs well in terms of CPU usage while in case of Multiple Row Insertion with less number (i.e. 5) of columns LINQ to SQL performs well and with more number (i.e. 10) of columns Transact Query performs well.

For Select Query evaluation, we selected 5000 rows with five columns from the database by these three methods, Transact Query, LINQ to SQL, and Entity Framework. The results are given in Table 2. From the Table 2. It is clear that in Select Query

- The Transact Query consumes less memory with less number (i.e. 5) of columns while with more numbers (i.e. 10) of columns LINQ to SQL consume less memory as compared to the other two.
- In terms of CPU usage LINQ to SQL performs well as compared to the other two irrespective the number of columns.

For Select with Join Query the results given in Table 2 show that

- The Transact Query consumes less memory and CPU usage as compared to other two frameworks while Entity Framework consumes more memory as compared to other two.

For Select with Order Query the results in Table 2. Show that the Transact Query consumes less memory and CPU usage as compared to other two frameworks whereas the Entity Framework consumes more memory and CPU usage as compared to Transact Query and LINQ to SQL.

After experimenting on Select with Join Query and Select with Order query. The Select with Filters Query experiment was carried on one filter, two filters and three filters respectively. The results in Table 2. For Select with Filters show that

- In terms of memory usage for select with one filter or two filters, the LINQ to SQL consumes less memory as compared to other two frameworks.

Table 2. CPU and memory usage results of eleven queries

Query Name	Memory Usage (KB)			CPU Usage (ms)		
	Transact Query	LINQ to SQL	Entity Framework	Transact Query	LINQ to SQL	Entity Framework
Insert Query Single Rows (5 Columns, 10 Columns)	0.9	2.75	13.64	1365.5	236.5	339
	0.6	10.55	11.58	1715	394.5	862
Insert Query Multiple Rows (5 Columns, 10 Columns)	0.215	900.275	11.57	12706.5	11662	23039
	2.725	910.015	17.694	6716	13352.5	19407.5
Select Query (5 Columns, 10 Columns)	10.255	954.025	3556.364	612	227	1176
	1670.45	1266.025	5047.94	611	596	775
Select Query with Join	1.705	104.4	5047.98	157	532	340
Select Query with Order	4.155	5320	16321.42	83	26066.6	28687.5
Select Query with Filters	7113.81	5342.93	16327.69	1265.5	508	1738.5
	7136.67	5328.455	16308.75	929	921.5	1593.5
	828	4339	16328.615	632	1196	1807
Select Query with Filter and Join	2538.755	829.65	837.875	847	463	542
Select Query with Filters, Joins and order	2537.82	838.135	849.175	862.5	1058	744.5
Delete Query with Filter	0.4	353.315	232.75	769	7149.5	15810
Update Query with Filter	0.38	338.56	727.33	243.5	7699	47428
Update Query with Filter and Order	0.825	348.325	727.59	345.5	5024	61866.5

- We observed from the Table 2. That when the number of filters is increased to 3. The Transact Query consumes less memory as compared to other frameworks.
- In terms of CPU usage for select with one filter or two filters the LINQ to SQL performs well as compared to Transact Query or Entity Framework.
- It is observed that from the Table 2, The Transact Query consumes less CPU usage as compared to other frameworks.

In case of using Select with both join and Order Query, the results from Table 2. Show that

- The LINQ to SQL consumes less memory as compared to other two frameworks.
- In terms of CPU usage the Entity Framework performs well as compared to other two frameworks.

After details experiment on Select Query. The experiment on Delete with Filter Query was carried. The results of Delete with Filter Query from Table 2. Shows that

- In terms of CPU usage and memory usage the Transact Query consumes less as compared to other frameworks.

The results of the Update with Filter Query in Table 2. Shows that

- The Transact Query consumes less memory and CPU usage as compared to other frameworks.

Finally, the experiment on Update with Filter and Order Query was carried out. The results from Table 2. Show that

- The Transact Query consumes less memory and CPU usage as compared to Entity Framework and LINQ to SQL.
- From Table 2. it is also observed that the Transact Query consumes less memory and CPU usage for Delete with Filter Query, Update with Filter Query as well as Update with Filter and order query.

Fig. 2. Describes overall brief comparison of memory usage of the selected 11 queries. It is clear from the Fig. 2. That Transact Query performance in terms of memory usages is outclass.

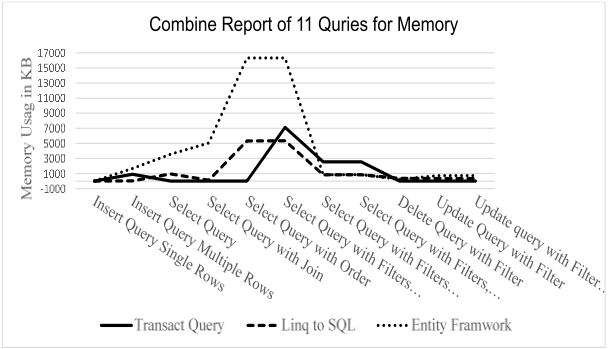

Figure 2. Combine graph of 11 queries for memory usage usage.

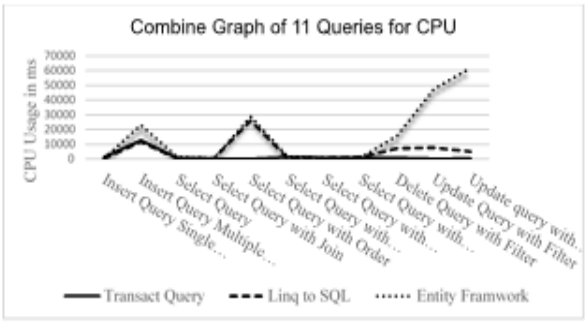

Figure 3. Combine graph of 11 queries for CPU usage

The Fig. 3. Illustrates brief comparison of all the 11 queries CPU usage. It is clear from the Fig. 3. That the Transact Query performance is better than other models in terms of CPU.

For security evaluation of ORM Frameworks, all the attacks from Table 1. are applied. The results from Table 3. Show that

Table 3. SQL injections results

SQL Injection	LINQ to SQL	Entity Framework	Transact Query
Tautologies	0	0	1
Union Queries	0	0	1
Piggybacked Queries	0	0	1
Stored Procedure	0	0	1
Alternate encoding	0	0	1
Inference	0	0	1
Malformed Queries	1	1	1

- All SQL Injections attacks are passed successfully in Transact Query while LINQ to SQL and Entity Framework show resistance to the attacks.
- From Table 3. It is also clear that the Malformed Attack is passed in LINQ to SQL and Entity Framework.

RECOMMENDATIONS AND CONCLUSION

From the above results we are giving following recommendation to the practitioners.

MEMORY USAGE RECOMMENDATION

- If Single Row Insertion or Multiple Rows Insertions are needed, use Transact Query.
- When an application uses the Select Statement, for fewer columns use Transact Query, but for more columns use LINQ to SQL.
- In case using Select with Join or Order we recommend to use Transact Query.
- In case using Select with Filter Query, we recommend to use LINQ to SQL.
- For Delete Queries and Update Queries, we recommend to use Transact Query.

CPU USAGE RECOMMENDATIONS

- In case of Single Row Insertion or Multiple Row Insertion use LINQ to SQL.
- Our results show that for Select Statement, the best framework is LINQ to SQL.
- For Select with Join Query or Select with Order Query the best option is Transact Query.
- For Select with Filter Query, the best option is LINQ to SQL while for Join with Select and Filter, our recommendation is still LINQ to SQL.
- In case of Select with Filter, Join and Order Query, the results show that the Entity Framework is the best option.
- In the case of Delete Query use LINQ to SQL.
- For Update Query, our recommendation is Transact Query, Similarly, if Filter and Order with Update Query are used, still our recommendation is Transact Query.

SECURITY RECOMMENDATIONS

- We concluded that the Transact Query makes the database vulnerable; it cannot prevent the SQL Injection Attacks by default. If a Transact Query is directly used in an application without any different technique for preventing SQL Injection, it will pass the SQL Injection. So our recommendation is not to use the Transact Query without any extra measure to prevent SQL Injections attacks.

CREDIT AUTHOR STATEMENT

Attaullah: Conceptualization, Methodology, Software, Writing- orginial draft preparation **Muhammad Usman**: Supervision **Muhammad F. Abrar:** Co-Supervision, Data curation. **Najeeb Ullah***:* Writing-Reviewing and Editing **Ibrar A. Shah**: Software Validation **Muhammad F. Nadeem:** Visualization

TAKE CREDIT AUTHOR KEYWORDS FROM: Conceptualization, Methodology, Writing- Original draft preparation, Supervision, Co-Supervision, Data curation, Writing- Reviewing and Editing, Software Validation, Visualization, but not limited to the above keywords

COMPLIANCE WITH ETHICAL STANDARDS

It is declared that all authors don't have any conflict of interest. Furthermore, informed consent was obtained from all individual participants included in the study.

REFERENCES

[1] Joshua R. Dick, Kenneth B. Kent, Joseph C. Libby, "A quantitative analysis of the .NET common language runtime", *Journal of Systems Architecture*, Volume 54, Issue 7, Pages 679-696, 2008.

[2] M. H. Lutz and P. A. Laplante, "C# and the .NET framework: ready for real time", *IEEE Software*, vol. 20, no. 1, pp. 74-80, Jan.-Feb 2003.

[3] J. Richter, "Microsoft .NET Framework Delivers the Platform an Integrated Service-Oriented Web," *MSDN Magazine*, 2000.

[4] ROSS MISTRY. Stacia Misner, "Introducing Microsoft SQL Server 2014", *Microsoft Press*, Washington, 2014.

[5] Dr V.Sivakumar, T.Balachander, Logu, Ramu Jannali, "Object Relational Mapping Framework Performance Impact", *Turkish Journal of Computer and Mathematics Education*, Vol.12 No.7, Pages 2516-2619, 2021.

[6] Václav Kaczmarczyk, Zdeněk Bradáč, Jakub Arm, Ondřej Baštán, Zuzana Kaczmarczyková, "A Simple and effective ADO.NET-based ORM layer", *IFAC-PapersOnLine*, Volume 52, Issue 27, Pages 228-234, 2019.

[7] Abdulkadir Karachi, "Performance Comparison of Managed C# and Delphi Prism in Visual Studio and Unmanaged Delphi and C++ Builder Languages", *Journal of Computer Applications*, Volume 26-No.1, 2011.

[8] Yener, Oguz, Tugce and Adil, "Performance Comparison of PHP-ASP Web Applications via Database Queries", *In Proceeding of the International Conference on Engineering & MIS*, 1-3. 10.1145/2832987.2833054, 2015.

[9] Atul Mishra, "Critical Comparison of PHP and ASP.NET for Web Development", *International Journal of Science & Technology Research*, Volume 7, 2014.

[10] Natela & Merab, "Reforming the Trees C# and F# Comparison", *IV International Conference of "Problems of Cybernetics and Informatics*" (PCI'2012), 2012.

[11] Li, Yishan & Manoharan, Sathiamoorthy, "A performance comparison of SQL and NoSQL databases", *In Proc. 15-19th IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, 10.1109/PACRIM. 6625441, 2013.

[12] Laaziri, Majida & Benmoussa, Khaoula & Khoulji, Samira & Larbi, Kerkeb &Yamami, Abir, "A comparative study of Laravel and symfony PHP frameworks", *International Journal of Electrical and Computer Engineering*, 9. 704-712.10.11591 /ijece.v9i1. pp 704 712, 2019.