Web Programming, Cloud Services and Quality of Experience (QoE) for Mobile Computing

Ustijana Rechkoska-Shikoska
University for Information Science & Technology UIST “St. Paul the Apostle”
Faculty for Computer Science and Engineering CSE
Ohrid, Republic of Macedonia
e-mail: ustijana.r.shikoska@uist.edu.mk

***Abstract*— A multi-platform application and its evaluation is presented in this work. It refers to the business process management software that allows organizations, institutions and companies, to use a system of integrated applications to manage the business and automate many back office functions related to technologies, services and human resources. The app is used for administration data, reviewing orders, creating invoices and adding new employees’ records. The app is also implemented in Azure, which is used as a platform for the cloud. A Web API and an Android application would significantly increase the Quality of Experience (QoE) of numerous mobile users, which can be implemented using the cloud services. It facilitates user’s work in terms of memory usage, hardware load, and responsiveness. QoE evaluation is done, as well. We managed to, not just create a reliable app, but also we made sure that it can expand and reach every user. Our approach improves the QoE, providing the users all necessary resources and performances, referring to Security, Easy to use concept, Time saving, and QR Code.**

Keywords- Web App development; Cloud services; QoE; Web application; Mobile computing

I. INTRODUCTION

Information and Communications Technology (ICT) leads the world of modern technologies, more and more platforms come up every day, so software developers have serious challenges choosing the right tools in order to complete their tasks. This paper gives the feedback on some tools usage and helps in further understanding the main concepts in creating a multi-platform application, giving evaluation sense as well.

The concepts of maintaining a database using SQL are presented in [14]. The procedures can help a developer in creating scalable databases and they are explained here as well. The cloud implementation is done, so the multi-platform applications can easily communicate in-between. Also, functions and procedures from [15] and [16] are implemented.

Section 2 of the paper presents the Related work in the area, Section 3 gives a general overview of an app. In the first subsection, we talk about SQL databases, ER diagram explaining the details. In addition, we talk about some of the developed functions and how they can help developers towards the maintaining the data constraints. In addition, we give a little demonstration of the power of stored procedure, and a particular procedure. Moreover, the application is deployed on the cloud. In the second subsection, a presentation of the user interface of the application and some of the core features are introduced. Sections 4 and 5 speak about testing and results, as well as QoE. Sections 6 and 7 give the reader conclusions and future work ideas.

II. RELATED WORK

In [1], precise details on the importance of databases creation and implementation are presented. Authors give a comparison on several database management systems and their differences, which can contribute in choosing the right one. On the other hand, in [2] a web based data-driven application, and all its concepts, which give a great representation how today’s applications work, is presented. The applications in [2] are developed in Java, and the concepts can be used almost in every C style language.

Authors in [4], [5] present a great result for building data driven Web App. In [3], authors talk about the basics of SQL, [5] gives an advanced look into the language and go through some of the specific individual operations that each of the most known database management systems offer. In [6], [11], [12] great examples on both developing Windows and Web applications, using the C# language are given. It takes one of the concepts and breaks it apart thoroughly, from the bottom of the Class Library to converging it with other technologies, which gives a great concept on making easy and useful steps on creating excellent applications. The authors in [7][8] give the background of the language itself, representing the Common Language Runtime (CLR), giving concise explanation how it is transformed to byte-code and more.

In [9], an introductory overview of Cloud computing, several distinctions of different Cloud platforms such as Google Web Service, Amazon web service and Windows Azure, Cloud infrastructure are presented. The authors present Mobile Cloud, a review into services and applications. The authors in [10] give an in-depth analysis into cloud design patterns, the security flaws of each setup, the stability of architecture.

Referring the related work mentioned, the App presented in this work improves the security, easy to use, time saving concepts, and QR Code.

III. DEVELOPING WINDOWS APPLICATION

Windows applications, as well as mobile applications, are forms of application program, and are designed to perform

Copyright (c) IARIA, 2018. ISBN: 978-1-61208-650-7

specific tasks directly for the user or, in some cases, for another application program. Web development implementation, as a process of building the web according to its design, enhances web specification, its implementation process, and presentation.

A. *App's architecture*

Application architecture is built in such a way to provide efficient application and implementation. The Application itself is consisted of Web API, Web App, Employee panel, Admin Panel, connected to the Database, also connected to the PCs, Mobiles, iPads etc.

The decision to use cloud architecture is made for the simple reason of scalability and accessibility - the Web Application, its' APIs, and Database, resides in the cloud. With this decision we have managed to cover all devices on the market and in the future target each one with a separate native application. The application itself is a dot.net core application which, by the .net core standards, can be developed on any machine. This is also a positive outcome, meaning that the application is optimized to use only the APIs that it needs, and also with additional requirement easily adding new ones. Whit this, concepts of scalability and meeting clients' needs are satisfied. Another feature that can come out with the cloud is using social networks, also we can optimize the virtual machines' performance by our needs, automatic jobs, have a multi-tenant, cloud-based directory, and identity management service (such as Azure Active Directory).

The database on the Cloud has its own server, instead of being an in app database, and a link to it resides in the app, which is a web application. There is a deployment of both the application and database as well. The database has its own server, instead of being an in app database, and a link to it resides in the app, which is a web application. There is an admin panel within the application, the implementation of employee panel and several other features will be included to improve the functionality. The application can be accessed from both mobile devices and PCs or laptops via Web browsers of any choice. A Web API and an Android App would significantly increase the Quality of experience of numerous mobile users, which can be implemented using the cloud services.

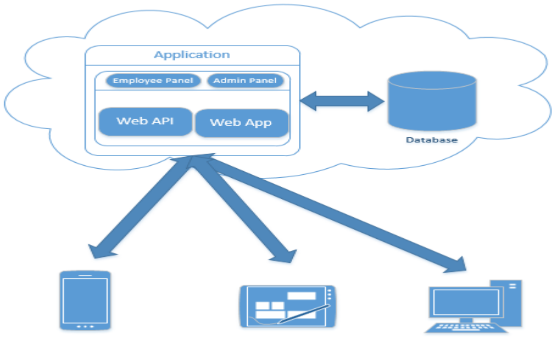


Figure 1. Application architecture

Figure 1 presents the application architecture.

B. *App's Database and SQL*

Each table in the database has its unique primary key and the specialized data in order to represent one entity. Some of the tables have composite keys, of two or more fields, with which we prevent concurrent data appearing in the database. In addition, there are unique keys on some of the table to strengthen the previous claim. The tables have constraints that are based on custom-made functions to help prevent inaccurate information being entered into the tables.

C. *Functions Developed*

Functions here serve as check constraints helpers, others as helpers to stored procedures so that large code will not be repeated, and some are used into views. One of the function checks the country calling code and the length of the phone number entered in the phone field of the Employee, Client and Supplier table. With this, we are preventing to have entries of any other country except chosen ones.

The function presented within next code lines, takes three parameters date, number of months that we want to add and the number of days that we want to add. Then, the difference between the months from zero until today are calculated, zero being 01.01.1900, and a number of month is added. After that, we add the number of day. The main key here is because the DATEDIFF function returns an integer we cannot use it as a standalone function because we a date type not an integer. That's why we have to add the DATEADD function.

```
CREATE FUNCTION [dbo].[MonthYear]
(
        @date date,
        @monthPlus int,
        @dayPlus int
)
RETURNS date
AS
BEGIN
        DECLARE @MonthYear date

        SET @MonthYear = (DATEADD(MONTH,
DATEDIFF(MONTH, 0, @date)+@monthPlus, @dayPlus))

        RETURN @MonthYear
END
```

Figure 2. Create Function

D. *Stored procedure*

There are many stored procedures for the database and they all serve for a different purpose. For instance, there are procedures for creating invoices both purchase invoices and regular, other server for updating the stock and so on. The procedure presented in the code snippet first checks if the date is later than the 20th of the current month and if the most recent generated pay is at least one month old. If that is true, the procedure calculates the sum of the total of working hours and inserts it into the pay table. Finally, it returns the generated pay. If it is false, it returns the most recent pay.

```
PROCEDURE [dbo].[CreatePay]
        @employeeID int
AS
BEGIN
        SET NOCOUNT ON;
        DECLARE @grossTotal money;
        IF(GETDATE() > dbo.MonthYear(GETDATE(),0,19) AND (SELECT TOP 1 dbo.MonthYear(Date,1,0)
        FROM Pay WHERE EmployeeID = @employeeID ORDER BY Date DESC) = dbo.MonthYear(GETDATE(),0,0))
        BEGIN
        SET @grossTotal = (SELECT SUM(Total) AS Total FROM LogEmployeeHoursTotal WHERE EmployeeID = @employeeID AND Date BETWEEN dbo.MonthYear(GETDATE(),-1,19) AND dbo.MonthYear(GETDATE(),0,19));
                INSERT INTO Pay(EmployeeID, Date, GrossPay)
                VALUES (@employeeID,GETDATE(),@grossTotal);
        END
        SELECT * FROM PayView WHERE EmployeeID = @employeeID AND Printed = 0;
END
```

Figure 3. Procedure listing

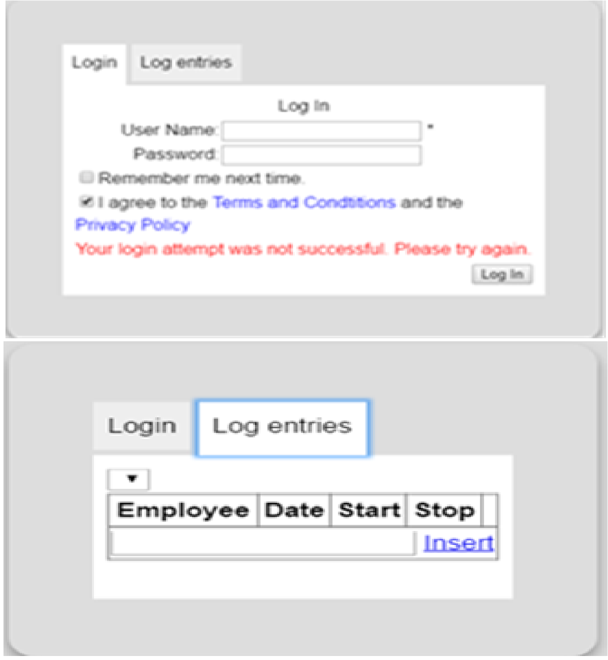
The developed application is a small enterprise resource planning software. It has a login screen, which every admin has its own username and password to log in, also log entry page where each employee checks in when comes or leaves the working place. In addition, with the app being implemented on the cloud, instead of a typical login you can also do an external one. This provides us with the opportunity to use social networking services, like Facebook, Google, or Twitter, to login or register into the app.

When the user logs in a tree menu, a window on the left appears in order the user can choose in which segment he/she wants to work in, and the dynamic content appears on the right. The menu is scrollable if it is expanded and out of screen but the dynamic con-tent stays fixed. If user goes deeper into the tree menu, he/she would get a specific operation for each node.

Here the user can see that there are textboxes where the required data can be written. The little three icons in the bottom represent the operations that users can perform, save, delete and edit previously generated pay. When the user clicks the generate pay button a popup appears showing the latest payment generated and there it can be either printed or updated.

The application has some good features like generating monthly wages automatically every 20th of the following month.

It shows all the required info for each sub-section (e.g., Employees name, address, when they took their last payment, how much was it, client specific invoices, etc.).



Log In

User Name: *

Password:

□ Remember me next time.

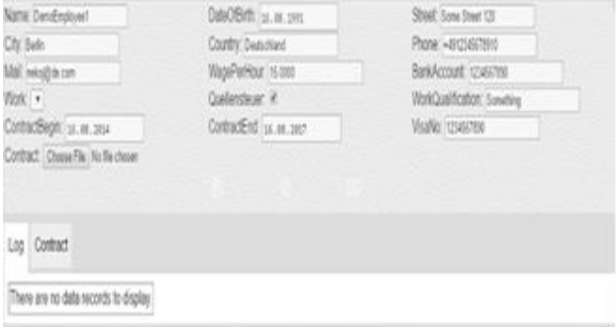
☑ I agree to the Terms and Conditions and the Privacy Policy

Your login attempt was not successful. Please try again.

Employee	Date	Start	Stop
			Insert

Figure 4. Login and Employee log screens

As it can be noticed in Figure 4, this is an user friendly App. In Figure 5, an example for the employee node is shown:



Name: DemoEmployee1

City: Berlin

Mail: neko@de.com

Work: ▾

ContractBegin: 18.08.2014

Contract: Choose File No file chosen

DateOfBirth: 18.08.1991

Country: Deutschland

WagePerHour: 15.000

Quellensteuer: ✓

ContractEnd: 16.08.2017

Street: Some Street 123

Phone: +4912345678910

BankAccount: 123456789

WorkQualification: Something

VisaNo: 123456789

There are no data records to display.

Figure 5. Employee node

It updates stocks frequently, keeps track of every out-going or incoming invoices, and all of that can be done with just a simple push of a button.

IV. Testing and Results

Several tests were performed for the application. One scenario is about performing a test in order to follow the hardware load when the application runs.

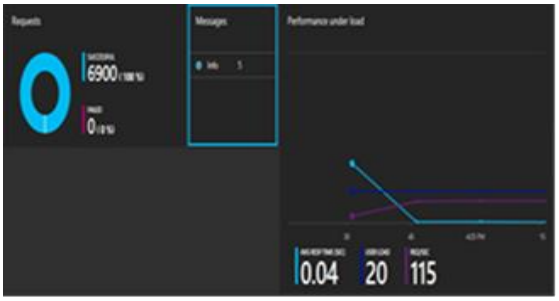


Figure 6. Testing results for performance of the application

As in Figure 6, and Figure 7 respectively, the application did astonishingly well with a 100% success. Moreover, it can be noticed that the response time is fast, meaning fast delivery of service. In addition, in the next figure it can be seen that the application does not really take much CPU time and memory while running.

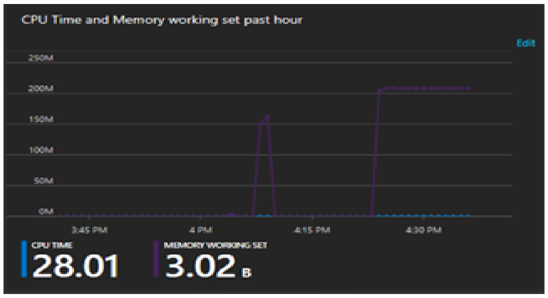
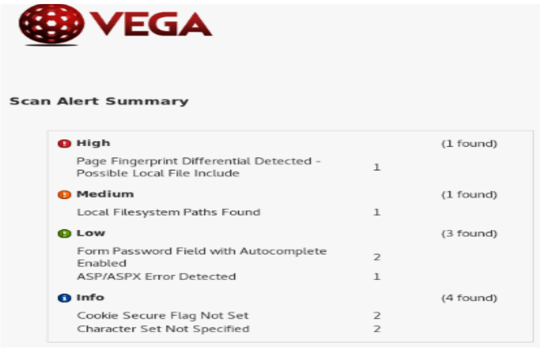


Figure 7. CPU load and Memory usage test for the app

Figure 8 is presenting the Pen testing process.

The Web Application's security was put to the test through several penetration testing tools from the Kali OS, such as Vega, OWASP ZAP, sqlmap and w3af.

There are some minor issues like low priority security, but as far as SQL injection and XSS security are concerned, the app outdone its self.

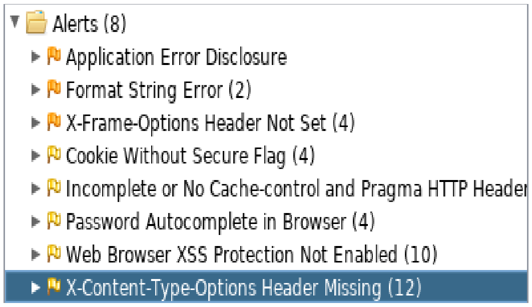


VEGA

Scan Alert Summary

High		(1 found)
Page Fingerprint Differential Detected - Possible Local File Include	1	
Medium		(1 found)
Local Filesystem Paths Found	1	
Low		(3 found)
Form Password Field with Autocomplete Enabled	2	
ASP/ASPX Error Detected	1	
Info		(4 found)
Cookie Secure Flag Not Set	2	
Character Set Not Specified	2	

(a)



Alerts (8)
- Application Error Disclosure
- Format String Error (2)
- X-Frame-Options Header Not Set (4)
- Cookie Without Secure Flag (4)
- Incomplete or No Cache-control and Pragma HTTP Header
- Password Autocomplete in Browser (4)
- Web Browser XSS Protection Not Enabled (10)
- X-Content-Type-Options Header Missing (12)

(b)

Figure 8 (a) Pen testing process – testing results from Vega (b) Pen testing process – testing tool results exposing the vulnerabilities

While Vega and OWASP ZAP found some minor security concerns, which can be easily fixed, sqlmap and w3af, which are more SQL Injection prone, found no problems regarding the previously mentioned argument.

V. Quality of Experience for Mobile Computing

Quality of experience (QoE) measures the difference between the user expectation and what the user received. This is beneficial to estimate the users' perception of the quality of the service and it depends users' satisfaction [12]. It represents how a service is accepted by the end-users. Using the QoE is beneficial to estimate the perception of the user about the quality of a particular service and it depends on the customer's satisfaction in terms of usability, accessibility, retaining ability and integrity of using specific service [13]. The results presented in the following chart were obtained according to the Quality of Experience evaluation performed on a group of representatives consisted of col-leagues from both Computer Science and Engineering, and Communications Net-work and Security Faculty, and one group consisted of professionals working in finances departments. They were given to run the application and a survey to rate the application's UI design, responsiveness, and user-friendly aspect of the app. The survey is based on the Mean opinion score (MOS). The MOS is calculated as the arithmetic mean over single ratings performed by human subjects for a given stimulus in a subjective quality evaluation test. Thus in (1):

$$MOS = \frac{\sum_{n=0}^{N} R_n}{N} \qquad (1)$$

R is the individual ratings for a given stimulus by N subjects. The results for each of the questioners are presented in Figure 9.

Referring our scenarios, 100 subjects were questioned, and the application were tested referring the timing of approximately ten minutes.

With the previous tests done, the users felt more secure handling the application, as almost 37% replied excellent to the survey question. Referring the QR code 35.5% were shown as excellent. The time saving factor was mixed and needs further assessment. The ease to use concept didn't do so poor, with almost 30% as excellent and 28.4 as very good, improvements would surely be beneficial here.

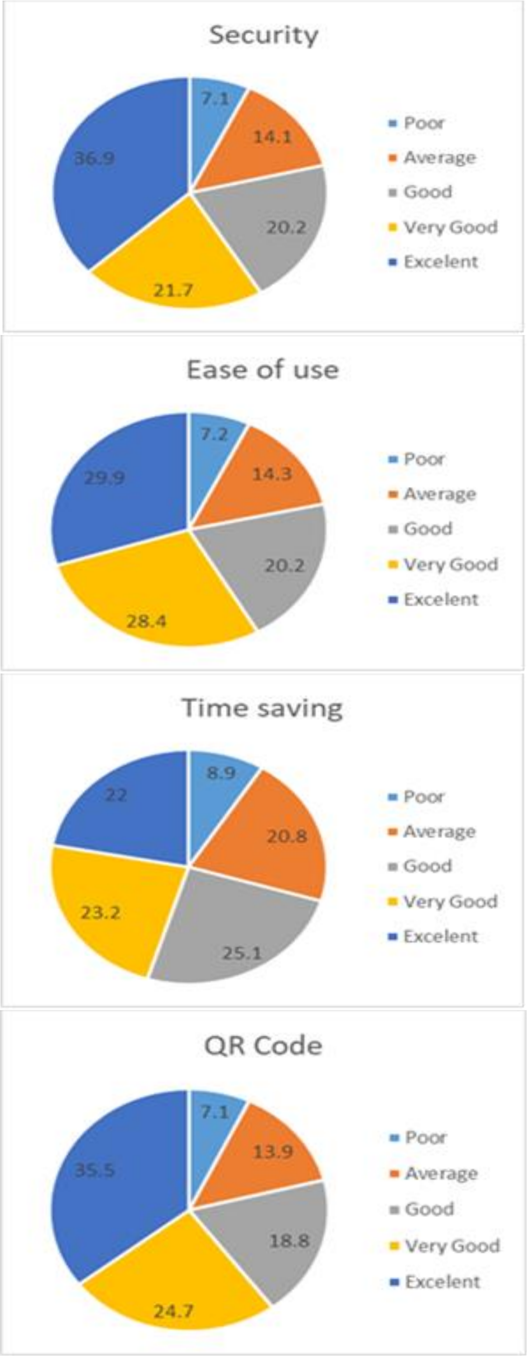


Figure 9. Survey results for the App

Easy to use: 29,9% replied as excellent, 28,4% as very good, 20,2% as good, 14,3% as average.

When it is up to time saving, 22% replied as excellent, 23,2% as very good, 25,1% as good and 20,8% as average.

For the Security concept, 36,9% replied as excellent, 21,7% as very good, 20,2% as good, 14.1% as average, which is satisfactory, since the security concept is demanded, especially speaking about web apps and cloud services, concerning database security, attacks and intrusions into the system.

About the survey results referring the QR Code, 35,5% considered as excellent, 24,7% as very good, 18,8% as good, 13,9% as average.

As it is presented in the charts, the users' satisfaction is positive. In terms of UI design and security (after presenting the test) the majority users gave an excellent review, while other thinks there can be improvements.

As for the responsiveness and ease of use the reviews were mixed, and therefore further improvement will be done.

The QoE and survey analyses show that satisfactory results are presented, of course there is always place for improvement, which is considered as future work.

VI. Conclusion and Future Work

As ICT technology goes forward, the App developing grows ever so rapidly, the developers need to adapt as well. In this work, we gave a glimpse of the tools that can help apps grow forward, and even with a simple change to readjust to the current technologies, we evaluate, as well, the App in order to show the advantages and send some recommendations to future developers. Moreover, the toolset can be expanded further thanks to the implementation of the Cloud Service. Some of the techniques and concepts that are presented can even increase the apps responsiveness, as well as the Quality of Experience that the users have. Automated some of the day-to-day tasks and made financial life easier. In a few words we managed to make sure that it can expand and reach every user.

For the future work, we will extend our work using several platforms, each one having their own app, and strengthen the current features furthermore. We will modify the web application using the Model-View-Controller pattern and the Entity Framework. A web API would greatly simplify the distribution of data on hand held devices more easily using the RESTful APIs. An Android application could to generate QR codes for employee log entries to substitute the current feature on the web app, even replace old systems and integrate IoT technologies. Furthermore, since the application will contain sensitive user data, security precautions that would be improved. In spite of that, further research in hybrid cloud will be performed, with having one private cloud and one public communicating with each other to prevent data loss, as well as further studies on how to improve current database security and web API security.

References

[1] Silberschatz, A., Korth, H and Sudarshan, S.: Database System Concepts. 6th edn. McGraw-Hill Education, 2010.

[2] Ramakrishnan, R. and Gehrke, J.: Database Management Systems. 3rd edn. McGraw-Hill Edu-cation, 2002.

[3] Elmasri, R. and Navathe, S.: Fundamentals of Database Systems. 7th edn. Pearson, 2015.

[4] Beaulieu, A.: Learning SQL: Master SQL Fundamentals. 2nd edn. O'Reilly Media, 2009.

[5] Molinaro, A.: SQL Cookbook: Query Solutions and Techniques for Database Developers. 1st edn. O'Reilly Media, 2005.

[6] Troelsen, A. and Japikse, P.: C# 6.0 and the .NET 4.6 Framework. 7th edn. Apress, 2015.

[7] Skeet, J.: C# in Depth. 3rd edn. Manning Publications, 2013.

[8] Albahari, J. and Albahari, B.: C# 6.0 in a Nutshell: The Definitive Reference. 6th edn. O'Reilly Media, 2015.

[9] Sosinsky, B.: Cloud Computing Bible. 1st edn. Wiley, 2011.

[10] Erl, T., Cope, R. and Naserpour, A.: Cloud Computing Design Patterns. 1st edn. Prentice Hall, 2011.

[11] Moon, Y.: Enterprise Resource Planning (ERP): a review of the literature. Mechanical and Aerospace Engineering, pp. 4., 2007.

[12] Fosser, E., Leister, O., Moe, C. and Newman, M.: Organisations and Vanilla Software: What Do We Know About ERP Systems and Competitive Advantage?. ECIS 2008 Proceedings. 132., 2008.

[13] W. Wu, et.al., “Quality of Experience in Distributed Interactive Multimedia Environments: Toward a Theoretical Framework,” MM '09 Proceedings of the 17th ACM international conference on Multimedia, October, 2009, pp. 481-490, DOI: 10.1145/1631272.1631338.

[14] MySQL Official Site (n.d.) Retrieved from http://www.mysql.com/, Retrieved: June 7, 2018.

[15] JSON Official Site (n.d.) Retrieved from http://json.org/, Retrieved: June, 2018.

[16] PhP Official Site (n.d.) Retrieved from http://www.php.net/, Retrieved: June, 2018.

ISBN: 978-1-61208-650-7