**MARMARA UNIVERSITY**

**FACULTY OF ENGINEERING**



**DESIGN AND PROGRAM A HEALTH NEWS RSS AGGREGATOR**

Atilla Gündoğan

**GRADUATION PROJECT REPORT**

Department of Electrical and Electronics Engineering

**Supervisor**

Asst. Prof. Salih Bayar ISTANBUL, 2022

**MARMARA UNIVERSITY**

**FACULTY OF ENGINEERING**

**DESIGN AND PROGRAM A HEALTH NEWS RSS AGGREGATOR**

**by**

**Atilla Gündoğan**

**June 18, 2022, Istanbul**

**SUMBITTED TO THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING IN**

**PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE**

**OF**

**BACHELOR OF SCIENCE**

**AT**

**MARMARA UNIVERSITY**

Shape

Description automatically generated**The author(s) hereby grant(s) to Marmara University permission to reproduce and to distribute publicly paper and electronic copies of this document in whole or in part and declare that the prepared document does not in any way include copying of previous work on the subject or the use of ideas, concepts, words, or structures regarding the subject without appropriate acknowledgement of the source material.**

Signature of Author(s)

Department of Electrical and Electronics Engineering

Certified By

Project Supervisor, Department of Electrical and Electronics Engineering

Accepted By

Head of the Department of Electrical and Electronics Engineering

# ACKNOWLEDGEMENTS

I would like to thank our beloved teacher Asst. Prof. Salih Bayar for his valuable support and advice on the project, Design and Program a Health News RSS Aggregator, both for guidance and moral support.

**June, 2022**

**Atilla Gündoğan**

# TABLE OF CONTENTS

[ACKNOWLEDGEMENTS ii](#_Toc107438936)

[TABLE OF CONTENTS iii](#_Toc107438937)

[ABSTRACT vi](#_Toc107438938)

[1. LIST OF SYMBOLS vii](#_Toc107438939)

[1.1. SQL Special Character Escape Sequences vii](#_Toc107438940)

[1.2. ASCII Characters viii](#_Toc107438941)

[2. ABBREVIATIONS x](#_Toc107438942)

[3. LIST OF FIGURES xi](#_Toc107438943)

[4. LIST OF TABLES xii](#_Toc107438944)

[5. OPERATORS AND RESERVED WORDS xiii](#_Toc107438945)

[5.1. PHP xiii](#_Toc107438946)

[5.1.1. PHP Arithmetic Operators xiii](#_Toc107438947)

[5.1.2. PHP Assignment Operators xiii](#_Toc107438948)

[5.1.3. PHP Increment / Decrement Operators xiv](#_Toc107438949)

[5.1.4. PHP Comparison Operators xiv](#_Toc107438950)

[5.1.5. PHP Logical Operators xv](#_Toc107438951)

[5.2. SQL xvi](#_Toc107438952)

[5.2.1. SQL Keywords xvi](#_Toc107438953)

[6. INTRODUCTION 1](#_Toc107438954)

[6.1. Thesis Content 2](#_Toc107438955)

[7. RESEARCH OBJECTIVE 3](#_Toc107438956)

[8. RELATED WORKS 4](#_Toc107438957)

[9. DESIGN 5](#_Toc107438958)

[9.1. Realistic constraints and conditions 5](#_Toc107438959)

[9.2. Cost of the design 5](#_Toc107438960)

[9.3. Development Standards 5](#_Toc107438961)

[9.4. Details of the design 6](#_Toc107438962)

[10. METHODS 7](#_Toc107438963)

[11. INTRODUCTION TO RSS 11](#_Toc107438964)

[11.1. Why use RSS? 12](#_Toc107438965)

[11.2. Who Should use RSS? 12](#_Toc107438966)

[11.3. Benefits of RSS 13](#_Toc107438967)

[11.4. RSS Standards 13](#_Toc107438968)

[11.5. How RSS Works 14](#_Toc107438969)

[12. WHAT IS XML? 17](#_Toc107438970)

[12.1. Why use XML? 18](#_Toc107438971)

[12.2. The Difference Between XML and HTML 19](#_Toc107438972)

[13. GITHUB 20](#_Toc107438973)

[13.1. Repository 21](#_Toc107438974)

[13.2. Branch 21](#_Toc107438975)

[13.3. Commits 21](#_Toc107438976)

[13.4. Pull Requests 21](#_Toc107438977)

[13.5. Merge 21](#_Toc107438978)

[14. PHP 22](#_Toc107438979)

[14.1. PHP MySQLi vs PDO vs MySQL 23](#_Toc107438980)

[14.1.1. MySQL 23](#_Toc107438981)

[14.1.2. PHP MySQLi 24](#_Toc107438982)

[14.1.3. PDO 24](#_Toc107438983)

[14.2. PHP SimpleXMLElement 24](#_Toc107438984)

[14.3. PHP htmlspecialchars\_decode Function 25](#_Toc107438985)

[14.4. PHP addslashes Function 25](#_Toc107438986)

[15. Apache 26](#_Toc107438987)

[16. MySQL 27](#_Toc107438988)

[17. MARIADB 28](#_Toc107438989)

[18. XAMPP 29](#_Toc107438990)

[18.1. Components of XAMPP 31](#_Toc107438991)

[18.2. What is Cross-Platform? 31](#_Toc107438992)

[19. PHPMyAdmin 32](#_Toc107438993)

[19.1. UTF8 vs UTF8MB4 34](#_Toc107438994)

[20. RESULTS AND DISCUSSION 35](#_Toc107438995)

[21. CONCLUSION 36](#_Toc107438996)

[REFERENCES 37](#_Toc107438997)

[APPENDICES 39](#_Toc107438998)

[Appendix 1 - SQL\_SCHEMA.md 39](#_Toc107438999)

[Appendix 2 - initiate\_sql.php 40](#_Toc107439000)

[Appendix 3 - sql\_conf.php 41](#_Toc107439001)

[Appendix 4 - ntv.php 43](#_Toc107439002)

[More Codes are Available on GitHub 45](#_Toc107439003)

# ABSTRACT

RSS Readers was a kind of social media in the 90s before Facebook and its competitors came along. Today RSS finds its place in the hands of avid readers who wants to keep up with the latest news and innovations. Many websites offer this service but not many do enjoy their benefits. For simplification and greater integration, we gather data from RSS XML files provided by publishers and push them to a SQL database. After pushing the data, we can pull them directly or query them to select one or more specific data with requested features. Once we get our requested data, we can create an algorithm and insert them into a Machine Learning application. We can also integrate the data into an existing application creating news panel. Also, an important note, since these are copyrighted work and shares to users’ consumption free of charge, we can't manipulate the data inside and add our own data. We can share or use it while giving references.

# LIST OF SYMBOLS

## SQL Special Character Escape Sequences

In SQL Databases like MariaDB or MySQL, we may need to query some special characters. Here is a list of SQL special character escape sequences.

**\0 :** An ASCII NUL (0x00)

**\n :** A newline

**\r :** A carriage return

**\t :** A tab

**\Z :** Control-Z

**\' :** A single quote (')

**\" :** A double quote (")

**\b :** A backspace

**\\ :** A backslash (\)

**\% :** A % character

**\\_ :** A \_ character

## ASCII Characters

ASCII is the first character set used for communications with computers over the internet and it was created at 1963 by ANSI (American National Standards Institute). Latest HTML, HTML5, character set that is UTF-8 is built on ASCII.

**&#32;** → Space ( )

**&#33;** → Exclamation mark (!)

**&#44;** → Comma (,)

**&#45;** → Hyphen (-)

**&#48;** → Digit 0 (0)

**&#125;** → Right curly brace (})

**&#126;** → Tilde (~)

**&#61;** → Equals sign (=)

**&#40;** → Left Parenthesis [(]

**&#41;** → Right Parenthesis [)]

**&#64;** → At sign (@)

**&#49;** → Digit 1 (1)

**&#95;** → Underscore (\_)

**&#96;** → Grave accent (`)

**&#42;** → Asterisk (\*)

**&#36;** → Dollar sign ($)

**&#37;** → Percent sign (%)

**&#38;** → Ampersand (&)

**&#62;** → Greater than (>)

**&#53;** → Digit 5 (5)

**&#54;** → Digit 6 (6)

**&#35;** → Number sign (#)

**&#50;** → Digit 2 (2)

**&#46;** → Period (.)

**&#47;** → Slash (/)

**&#51;** → Digit 3 (3)

**&#55;** → Digit 7 (7)

**&#56;** → Digit 8 (8)

**&#57;** → Digit 9 (9)

**&#58;** → Colon (:)

**&#59;** → Semicolon (;)

**&#60;** → Less-than (<)

**&#43;** → Plus sign (+)

**&#123;** → Left curly brace ({)

**&#124;** → Vertical bar (|)

**&#91;** → Left square bracket ([)

**&#92;** → Backslash (\)

**&#93;** → Right square bracket (])

**&#63;** → Question mark (?)

**&#34;** → Quotation mark (")

**&#52;** → Digit 4 (4)

**&#94;** → Caret (^)

**&#39;** → Apostrophe (')

# ABBREVIATIONS

**RSS:** Really Simple Syndication

**XML:** eXtensible Markup Language

**VPS:** Virtual Private Server

**VDS:** Virtual Dedicated Server

**VM**: Virtual Machine

**PHP:** Hypertext Pre-processor

**SQL:** Structured Query Language

**NOSQL:** Not Only Structured Query Language

**CPU:** Central Processing Unit

**RAM:** Random Access Memory

**XAMPP:** Cross-platform, Apache, MySQL, PHP, and Perl

**IaaS:** Infrastructure as a Service

**PaaS:** Platform as a Service

**SaaS:** Software as a Service

**ASCII:** American Standard Code for Information Interchange

**CLI:** Command Line Interface

**GB:** Giga Bytes

**GUI:** Graphical User Interface

**MHz:** Mega Hertz

**SEQUEL:** Structured English Query Language

**TCP:** Transport Control Protocol

**WWW:** World Wide Web

# LIST OF FIGURES

[Figure 1 Folder Structure of The Project 6](#_Toc107439004)

[Figure 2 phpMyAdmin Show Databases 6](#_Toc107439005)

[Figure 3 File Structure of the Project 7](#_Toc107439006)

[Figure 4 XAMPP Services Panel 8](#_Toc107439007)

[Figure 5 Visual Studio Code IDE 9](#_Toc107439008)

[Figure 6 Diagram of an example RSS feed 14](#_Toc107439009)

[Figure 7 Content of an example RSS feed 15](#_Toc107439010)

[Figure 8 RSS Feed Example 16](#_Toc107439011)

[Figure 9 Example XML Code 17](#_Toc107439012)

[Figure 10 XML in Plain Text Format 18](#_Toc107439013)

[Figure 11 Project Repository 20](#_Toc107439014)

[Figure 12 PHP Code for Reading XML content 22](#_Toc107439015)

[Figure 13 Connecting to MySQL Database Through MySQLi 23](#_Toc107439016)

[Figure 14 Usage of SimpleXMLElement 24](#_Toc107439017)

[Figure 15 Workflow of Apache Web Server 26](#_Toc107439018)

[Figure 16 Monitoring MariaDB Database Through Custom Script 28](#_Toc107439019)

[Figure 17 XAMPP MacOS Interface 29](#_Toc107439020)

[Figure 18 XAMPP Windows Interface 30](#_Toc107439021)

[Figure 19 phpMyAdmin Login Screen 33](#_Toc107439022)

[Figure 20 phpMyAdmin Interface 33](#_Toc107439023)

[Figure 21 Database Entries on phpMyAdmin 34](#_Toc107439024)

# LIST OF TABLES

[Table 1 PHP Arithmetic Operators xiii](#_Toc107439031)

[Table 2 PHP Assignment Operators xiii](#_Toc107439032)

[Table 4 PHP Increment and Decrement Operators xiv](#_Toc107439033)

[Table 3 PHP Comparison Operators xiv](#_Toc107439034)

[Table 5 PHP Logical Operators xv](#_Toc107439035)

[Table 8 SQL Keywords xvi](#_Toc107439036)

# OPERATORS AND RESERVED WORDS

## PHP

We perform operations on variables and their values with operators. They are grouped as below.

### PHP Arithmetic Operators

Table 1 PHP Arithmetic Operators

|  |  |  |
| --- | --- | --- |
| Operator | Name | Example |
| - | Subtraction | $n - $m |
| + | Addition | $n + $m |
| \* | Multiplication | $n \* $m |
| \*\* | Exponentiation | $n \*\* $m |
| / | Division | $n / $n |
| % | Modulus | $n % $m |

### PHP Assignment Operators

Table 2 PHP Assignment Operators

|  |  |
| --- | --- |
| Assignment | Equal To |
| k = s | k = s |
| k -= s | k = k - s |
| k += s | k = k + s |
| k \*= s | k = k \* s |
| k %= s | k = k % s |
| k /= s | k = k / s |

### PHP Increment / Decrement Operators

Table 4 PHP Increment and Decrement Operators

|  |  |  |
| --- | --- | --- |
| Operator | Name | Description |
| ++$c | Pre-increment | Increments first, then returns $c |
| --$c | Pre-decrement | Decrements first, then returns $c |
| $c++ | Post-increment | Returns $c first, then increments $c |
| $c-- | Post-decrement | Returns $c first, then decrements $c |

### PHP Comparison Operators

Table 3 PHP Comparison Operators

|  |  |  |
| --- | --- | --- |
| Operator | Name | Example |
| == | Equal | $k == $j |
| === | Identical | $k === $j |
| != | Not equal | $k != $j |
| !== | Not identical | $k !== $j |
| < | Less than | $k < $j |
| > | Greater than | $k > $j |
| <= | Less or equal to | $k <= $j |
| >= | Greater or equal to | $k >= $j |

### PHP Logical Operators

Table 5 PHP Logical Operators

|  |  |  |
| --- | --- | --- |
| Operator | Name | Example |
| and | And | $c and $v |
| && | And | $c && $v |
| or | Or | $c or $v |
| || | Or | $c || $v |
| xor | Xor | $c xor $v |
| ! | Not | !$c |

## SQL

This part contains SQL reserved words (keywords) which can be used both in MySQL and MariaDB. Reserved keywords have a certain functionality and should not be used to declare variables, tables, or columns.

### SQL Keywords

Table 8 SQL Keywords

|  |  |  |  |
| --- | --- | --- | --- |
| ADD | ADD CONSTRAINT | ALTER | ALL |
| ALTER COLUMN | ALTER TABLE | ANY | AND |
| CREATE TABLE | CREATE DATABASE | CREATE VIEW | VIEW |
| NOT NULL | OR | OUTER JOIN | ORDER BY |
| PRIMARY KEY | PROCEDURE | ROWNUM | RIGHT JOIN |
| SELECT | SELECT DISTINCT | SELECT TOP | SELECT INTO |
| DATABASE | DEFAULT | DESC | DELETE |
| DISTINCT | DROP | DROP CONSTRAINT | DROP COLUMN |
| DROP DATABASE | DROP DEFAULT | DROP TABLE | DROP INDEX |
| DROP VIEW | EXEC | FOREIGN KEY | EXISTS |
| FROM | FULL OUTER JOIN | HAVING | GROUP BY |
| IN | INDEX | INSERT INTO | INNER JOIN |
| INSERT INTO SELECT | IS NULL | JOIN | IS NOT NULL |
| AS | ASC | BETWEEN | WHERE |
| CASE | CHECK | CONSTRAINT | COLUMN |
| CREATE | CREATE PROCEDURE | VALUES | CREATE INDEX |
| UNION | UNION ALL | UPDATE | UNIQUE |

# INTRODUCTION

Today, readers find themself in a huge pile of information. There is just simply too many publishers, news and researches to keep up with. RSS feeds helped those readers until the 90s but then the technology started to lose its place and now the average person doesn't even know what an RSS feed is. Even though many websites, publishers, blogs and even forum sites offer this service there is not an intuitive way to use this information. In this project we gathered data from RSS XML files from publishers and inserted into a SQL Database so we can use it in other applications and projects. After inserting the data, we can pull them directly or query them to select one or more specific data with requested features.

## Thesis Content

In the following sections I will give detailed informations about the technologies I used in the project and explain how I used them. Sections will include Introduction to RSS, XML, GitHub, PHP and SQL APIs for PHP, Apache, MySQL, MariaDB, XAMMP, phpMyAdmin.

# RESEARCH OBJECTIVE

Although having too much information doesn't sound bad at all it can have a negative impact. Having too much unorganized information can make the desired info buried under unrelated topics and researches. But even too much organized information can cause problems. One of those problems is information overload.

Information overload is probably something you have felt before after hours of researching the web and you realize that there is too much data floating around and you can no longer make decisions about the topic. To prevent this is in this project we only gathered data from Turkish Heath News Publishers. We defined 6 categories including Title, Link, Media, Meta Description, Content and Publication Date. After inserting the data into a database, we can search and filter the data based on containing keywords in the content, name of the publisher, publication date, title, and content itself. Also, only the data in the form of RSS is not helpful. We need an RSS Reader or an RSS Aggregator.

The RSS data is in XML Form, and it is hard to read and not intuitive enough for the public. With the help of PHP, we will also display the database content in a web page, that will make it easier to read the data without full permission access to the database itself.

# RELATED WORKS

There is a few closed-source RSS Aggregators like Feedly and Inoreader which you can't run it self-hosted. And then there is NewsBlur, the biggest open-source RSS Reader with a plenty of features. But you must run it a dedicated server alongside Docker. Because you will need a dedicated server which may cost a lot and may add up with the number of publishers and data queries. Also, the code base is very large and thus it is hard to understand without a long read into the documentation. With the scripts in this project, you can run it cost-free alongside your website or on a very cheap website hosting.

There is a few closed-source RSS Aggregators which you can't run it self-hosted. And then there is NewsBlur, the biggest open-source RSS Reader with a plenty of features. But you must run it a dedicated server alongside Docker. Because you will need a dedicated server it may cost a lot. Also, the code base is very large and thus it is hard to understand without a long read into the documentation.

# DESIGN

## Realistic constraints and conditions

Since the RSS data contains copyrighted work we can't edit its content. We can only use it for ourselves or share it while giving credits to the authors and the publisher.

## Cost of the design

Cost of implementing the codes alongside a website installation with an admin panel installation is free. If there is no hosting at hand you can either purchase a cheap web hosting or host the scripts on a personal computer with XAMMP.

## Development Standards

While developing an app or a program we must follow the Programming Language's Syntax, requirements, and standards. Since we only used PHP in this project, we must use it accordingly in an intended way and syntax.

Standards for PHP: <https://www.php-fig.org/psr/>

Also their is industry standards of software development.

Standards for software from American National Standards Institute: <https://webstore.ansi.org/industry/software>

Standards for SQL Database:

<https://webstore.ansi.org/industry/software/technology-languages/SQL>

## Details of the design

The scripts work under 7 main folders. They communicate with require tags. There are over 30 PHP files in total, and they all help the project work as intended.

Graphical user interface, application

Description automatically generated

Figure 1 Folder Structure of The Project

With phpMyAdmin we can visualize our databases.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 2 phpMyAdmin Show Databases

# METHODS

I used several technologies and tools to make this project possible.

Graphical user interface

Description automatically generated

Figure 3 File Structure of the Project

I used GitHub to save and track my progress while I developed new features.

A screenshot of a computer

Description automatically generated with medium confidence

For development purposes I used XAMPP to run Apache, MySQL and phpMyAdmin locally. I used Apache and MySQL extensively.

Graphical user interface, application

Description automatically generated

Figure 4 XAMPP Services Panel

Apache is the most used free and open-source webserver software that allows users to deploy their websites on the internet.

I used Visual Studio Code through the entire development process. It is a free and open-source powerful IDE from Microsoft.

A screenshot of a computer

Description automatically generated

Figure 5 Visual Studio Code IDE

In the picture above you can see a lot of PHP files. All of them contributes to the project in some way or another.

PHP stands for Hypertext Preprocessor. PHP can generate dynamic page content, it can even create, open, read, write, and delete files on the server.

MySQL is a relational database which we used to push or insert our data into. We can create databases, create tables into those databases and execute read write operations. We can see our database with the help of phpMyAdmin. Below there is the screenshot of our database configuration.

**Graphical user interface, text, application, email

Description automatically generated**

# INTRODUCTION TO RSS

In this project we gather resources through RSS feeds. RSS feeds are used to distribute latest web content from one or more website (source) to the end users (client - receiver). End users can read the data contents with an RSS Reader or an Aggregator.

* Abbreviation of RSS is Really Simple Syndication.
* RSS can be called as a News feed or RSS feed.
* RSS allows fast searching / viewing of latest news and articles without bloat and make it able to gather all contents in on place organized under categories or folders.
* RSS does not follow a guideline and every web publisher can create their non-standard tags.
* It makes it easy to share and display content and data about the content.
* RSS can be configured to update automatically.
* RSS can ben personalized.
* RSS is written/shared is XML format.
* RSS data is small in size and loads fast.
* RSS is read through aggregators.
* RSS aggregator is a website or a program that collects, sorts out and display RSS feeds.

## Why use RSS?

RSS was designed to display specific data. Without RSS, readers will be required to check every website that they want to keep up with. This would consume a lot of time. RSS feeds bring the latest content in a single place so that the consumers (readers) can check the resources faster.

Because RSS data is small in size and loads fast compared to other solutions it can be used with portable gadgets like smart phones and tablets.

## Who Should use RSS?

RSS can be used by anyone especially for the avid readers who constantly want to stay up to date with the latest news. RSS used frequently used in News sites, Calendars, Companies, and even announcing site changes. Every website that updates frequently or posts new content regularly needs an RSS feed.

Generally;

For News sites the XML file contains data like content title, date of publication, description, and content itself.

For Calendars the XML file contains data like upcoming events, holidays, birthdays etc.

For Companies the XML file contains data like news, articles, and products.

For site changes the XML file contains updates and changes of pages.

## Benefits of RSS

There is a lot of benefits of using RSS. Some examples are given below.

1. You choose your news and articles you want to receive

As the benefits of RSS, you can control the publications you receive, check the Informations that interests you and are related to your researches.

1. You can remove undesirable information and unfollow feeds

You are able the separate information that you desire from others that can be classified as spam or ads if you prefer to use RSS.

1. You can increase your website's traffic if you create an RSS feed

If you implement an RSS feed to your website, you can create and manage your own channels and share it to the rest of the world.

## RSS Standards

Unfortunately, there are no official standards for RSS feeds. About half of the feeds use RSS version 0.91. About a quarter of feed use RSS 1.0 and the rest is split between RSS 2.0 and RSS 0.9x versions.

## How RSS Works

RSS is used to share content between distributors and consumers. You can add RSS feed link to services called RSS aggregators. You can even register your feed with them to automate the process.

Below you will find a typical RSS feed in block diagram.

Diagram

Description automatically generated

Figure 6 Diagram of an example RSS feed

Every tag opened must be closed at the end of the part. Inside the tags the content follows. The tag can contain attributes.

Text, letter

Description automatically generated

Figure 7 Content of an example RSS feed

RSS document file must end with .xml extension. Once uploaded to a server you can view it in the browser as plain text or use an RSS aggregator. You can register your feed with services like Feedly so that they index your feed periodically.

If you don't use an aggregator the XML file will the displayed in plain text in a browser. Once displayed in the browser, a typical RSS feed will look like figure below.

Text

Description automatically generated

Figure 8 RSS Feed Example

# WHAT IS XML?

Abbreviation XML stands for eXtensible Markup Language. XML was first used in 1998 and it was designed to store and transfer data across the internet. It is designed to be both human and machine readable. It is widely used in IT and many Internet services across the world. If facilitates the distribution of data over the World Wide Web. It is software and hardware independent so it can be utilized across devices and platform.

Graphical user interface, text, application, email

Description automatically generated

Figure 9 Example XML Code

Some important bullet points include;

* XML and HTML are markup languages like one another.
* XML was designed and implemented to store, transfer, transmit data over the WWW.
* XML content is self-descriptive and easy to understand.
* XML is a W3C Recommendation since 1998.
* XML codes does not execute, doesn’t do anything except containing data.
* XML is transferred over TCP protocol.
* XML file may contain sender, receiver information.
* XML file may have a heading and a content body.
* XML is just a form of data that is enclosed by tags.

## Why use XML?

In many IT systems the data is in incompatible format to each other. Exchanging data between different system with incompatible format is a challenge for developers and translation of data to other format is a wasteful in regards of compute power. In conversion of data some data may be lost, or some conversion problems may occur that may disturb the integrity of the data.

Some of the reasons on why we use XML;

* XML helps simplify data sharing
* XML helps transportation of data
* XML is platform independent
* XML is software independent
* XML is hardware independent
* XML improves data availability

XML stores its content in plain text format like the figure below.

Text, timeline

Description automatically generated

Figure 10 XML in Plain Text Format

## The Difference Between XML and HTML

Although both XML and HTML is a markup language, they were designed with different objectives in mind.

* The objective of XML is to carry data while describing data content.
* The objective of HTML is to display data while describing how data looks.
* XML does not have predefined tags.
* HTML only works with predefined tags (ex: <h1>, <b>, <div>).
* XML, as of its name, is extensible, meaning it can be extended or shrunk.

# GITHUB

GitHub is a code hosting platform that offers services like version control, project management and helps team members to work on the project together. Authorized members can pull, push, and merge code to add new features.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 11 Project Repository

Some GitHub essentials are;

* Repositories
* Branches
* Commits
* Pull Requests
* Merge

## Repository

We stored the code in a GitHub repository while in development. If we made any mistake and wanted to revert back to an older version, we could do it in a safely manner through git. We can store any code files, pictures, and documents. We can include a README.md file that we can explain about the project. We may also create a wiki to guide users of our project. GitHub repositories may be public or private.

## Branch

We can use a GitHub branch to develop or share different versions of the repository. We can create a development and stable repository for example. By default, a repository is initiated with a master branch while other branches were a copy of master branch at some point.

## Commits

Any changes to our code base are called commits. Commits has a description on every file that the change occurred.

## Pull Requests

Pull request are a request send to the moderators that the changes you made should be merged to the master branch. Pull request shows code modifications. After a commit, you can create a pull request and start a discussion inside GitHub.

## Merge

Once the development of new features or the bug fixes are finalized in a development repository it is merged into the master branch. Therefore, as soon as you have a pull request, you can merge any changes into the master by clicking a "Merge pull request" button.

# PHP

PHP is a backend scripting language that is mostly used for interactive (dynamic) website development. With PHP, developers can create beautiful dynamic websites and apps. It can be installed and run-on diverse platforms. PHP can interact with files on a server and run queries on many databases.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 12 PHP Code for Reading XML content

## PHP MySQLi vs PDO vs MySQL

While developing for PHP, to interact with a database we can choose from several Database APIs. Developers can choose one of them to access MySQL database, but each have their own advantages and disadvantages.

Text

Description automatically generated

Figure 13 Connecting to MySQL Database Through MySQLi

### MySQL

This was the original extension that eased the communication to MySQL database. However, it is currently deprecated and not recommended to use since PHP 7 and newer versions doesn't support it.

### PHP MySQLi

MySQLi stands for MySQL Improved. It is known to have better integration and features than its predicator. MySQLi has both Object-Oriented and Procedural API. MySQL is strictly dedicated to MySQL and MariaDB and can offer better support for them over PDO.

### PDO

PDO supports most used database systems and provides a uniform method of access to those databases. PDO has support for 11 different database systems. With PDO switching databases are easier but will require to test all SQL queries and responses.

## PHP SimpleXMLElement

With SimpleXML we can get and interact with XML data. From PHP 5, SimpleXMLElement is a core part of PHP. SimpleXML facilitates the process of getting an element's content, name, and attributes in an XML document. SimpleXML allow us to display XML document's data structure.

Text

Description automatically generated

Figure 14 Usage of SimpleXMLElement

## PHP htmlspecialchars\_decode Function

Some XML documents may include HTML special characters. To decode or convert them back to chatacter we use htmlspecialchars\_decode function.

## PHP addslashes Function

The addslashes function returns a string with backslashes added before some characters. We used this function to query single quotes, double quotes and backslahes to the database. Without this the SQL database can't query those characters.

# Apache

Apache is a widely used cross-platform web server to deliver web content. It is a free and open-source webserver that is developed and maintained under the Apache Software Foundation. An Apache server delivers web pages, images, sends documents and files.

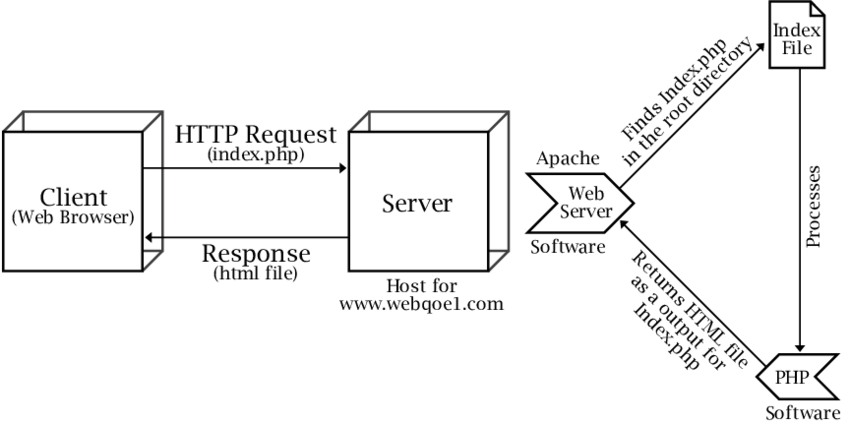


Figure 15 Workflow of Apache Web Server

* Apache receives the request of access.
* Apache runs security checks on HTTP request
* Apache allows the visitor to access the web page.
* Apache can run extra modules.
* Apache serves the documents requested.

# MySQL

MySQL is one of the most popular, widely used database. MySQL is a relational database management system (RDBMS) based on Structured Query Language (SQL). It is developed and maintained by Oracle.

In relational database the data is stored on excel like tables. In relational model tables consists of rows and columns. The logic between elements follows a strict path. The data can be emails, encrypted passwords, purchases and much more. A RDBMS is a compined toolset of software tools that is used to interact with tables and databases.

MySQL is an integral part of the modern web. It is used in many softwares, backend infrastructures and frontend web apps.

# MARIADB

MariaDB is a free and open-source Relational Database Management System (DBMS). MariaDB is close relative of MySQL, and both use the same query format. It was created from a fork of MySQL. MariaDB's founders are the developers who designed MySQL in the first place. Because MariaDB is binary-compatible with MySQL, developers can switch between them easily.

Originally, XAMMP included MySQL database, but because of licensing required to use MySQL it was switched to MariaDB.

Table

Description automatically generated

Figure 16 Monitoring MariaDB Database Through Custom Script

# XAMPP

XAMPP is a widely used cross-platform webserver, that can be run on Windows, Linux and MacOS. It is a free and open-source software package that is used to create a local development environment. It is developed and maintained by Apache Friends Community.

Graphical user interface

Description automatically generated

Figure 17 XAMPP MacOS Interface

XAMPP is an abbreviation of Cross-Platform for X, Apache for A, MariaDB for M, PHP for P and Perl for P respectively. XAMPP contains command-line executables and developers can add extra functionality by modifying ini files.

Graphical user interface

Description automatically generated

Figure 18 XAMPP Windows Interface

## Components of XAMPP

XAMPP is a stack of software tools and technologies. It provides a base for development and testing on a local machine. We may also use it to host our website, but we do not recommend it. XAMPP include following components;

* Apache
* MariaDB
* PHP
* Perl
* phpMyAdmin
* Webalizer
* Mercury
* Tomcat
* FileZilla
* XAMPP Control Panel
* OpenSSL

## What is Cross-Platform?

Cross-Platform means that the software is platform independent. It will run on more than one platform or operating environment. Cross-platform increases utility and user range for the software.

# PHPMyAdmin

phpMyAdmin is a free and open-source software tool for interacting with MySQL and MariaDB. phpMyAdmin's main functionality is to interact and handle the administration of MySQL or MariaDB databases. It can show databases, tables, rows, and columns. It can also query new data or drop (delete) existing ones. phpMyAdmin supports multiple users with custom permissions. Operations can be performed through the online admin interface panel.

phpMyAdmin Features;

* Easy to understand admin panel
* Support for MySQL and MariaDB features
* Browse databases and tables
* Create, copy, delete, and alter databases, tables
* Server configurations
* Execute, edit, and save SQL queries
* Manage MySQL or MariaDB user accounts and privileges
* Export data to other formats
* Searching data globally or in a database

Graphical user interface

Description automatically generated

Figure 19 phpMyAdmin Login Screen

Once we login we will be greeted with the interface below.

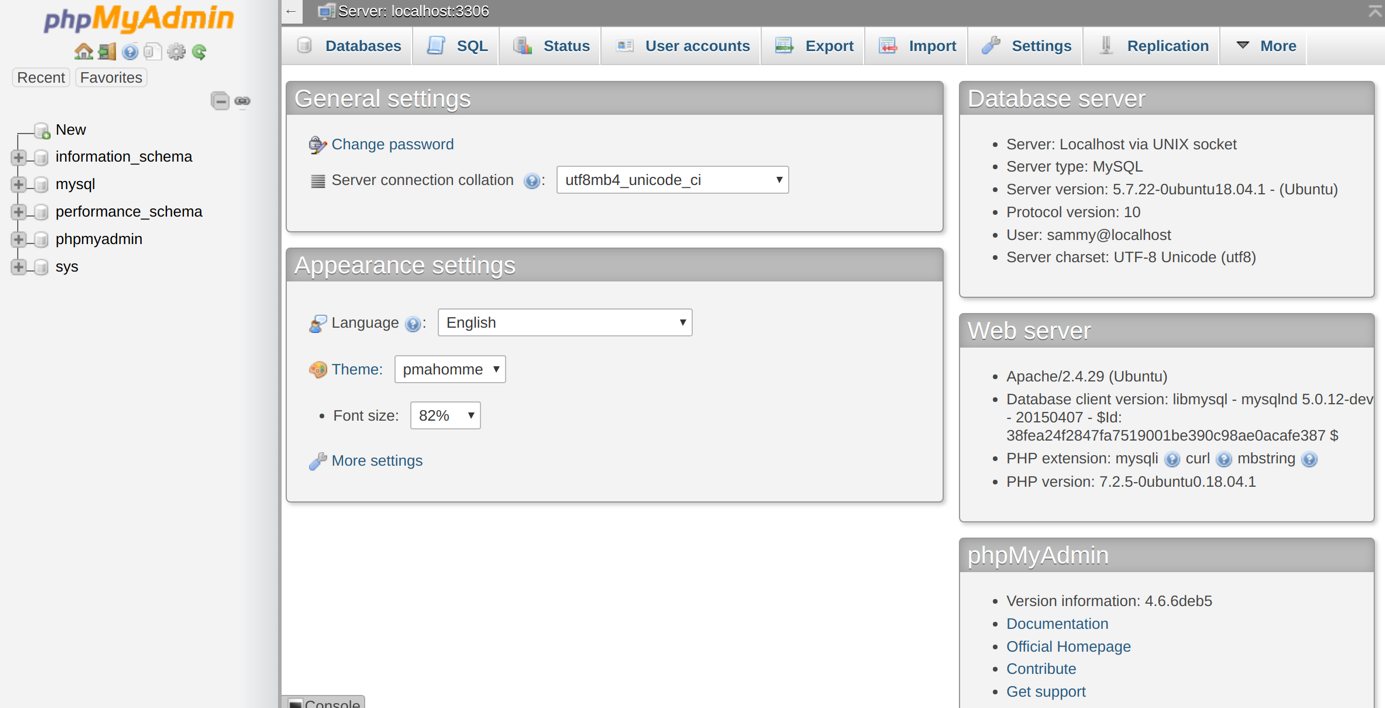


Figure 20 phpMyAdmin Interface

Our datas will be in a table inside a database. As of this project the database name is "saglikdb" and the table name is "entries".

Graphical user interface

Description automatically generated with low confidence

Figure 21 Database Entries on phpMyAdmin

## UTF8 vs UTF8MB4

According to MySQL 8.0 Reference Manual UTF8MB3 (Classic UTF8 for MariaDB) encoding is only using three bytes per character while UTF8MB4 is using four bytes per character. Because of this characteristic UTF8MB4 can be used to store not only Latin language characters but also symbols, emojis and Turkish characters. UTF8MB3 will be removed in the future releases.

# RESULTS AND DISCUSSION

We have reached the desired result but we can make it work a lot better if we use a NOSQL database like MongoDB or Firebase since it will get rid of some limitations and give us some new features to look deeper into our data. We can also make use of Google’s Machine Learning platform and do more calculations to increase our productivity. We did not implement an SSL Certificate and we may need to add it so when we expose our server to the public we can protect our services.

# CONCLUSION

Utilizing RSS sources and XML files we have successfully created a Heath News RSS Aggregator. With RSS we can choose to view the news and articles we want, the publications that interest us and are relevant to our work. With RSS we remove unwanted information. RSS services uses XML as a language and XML doesn’t have a standard. RSS Readers doesn’t read and display all the data. Using PHP, we can get access to every data and use it across our projects. At the end we made a lightweight program that integrated RSS data to SQL Database. In the later projects we can use this data extensively and create a Machine Learning Application. This project will also serve as a Search Engine as much as it is now serving as an RSS Aggregator. By using user data end queries, we can create a simple Google alternative.

# REFERENCES

American National Standards Institute. *SQL Database Standards.* Retrieved from https://webstore.ansi.org/industry/software/technology-languages/SQL

W3Schools. HTML ASCII Reference: The ASCII Character Set. Retrieved from https://www.w3schools.com/charsets/ref\_html\_ascii.asp

MySQL. MySQL 8.0 Reference Manual. What Is MySQL? Retrieved from https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

Apache Friends. About XMAPP. Retrieved from https://www.apachefriends.org/about.html

MariaDB Server Documentation, Using MariaDB Server, SQL Statements and Structures, SQL Language Structures, Reserved Words. Retrieved from https://mariadb.com/kb/en/reserved-words/

PHP Framework Interop Group, PHP Standards Recommendations, Retrieved from https://www.php-fig.org/psr/

Tutorials Point, Basics of Computer Science Abbreviations, Retrieved from https://www.tutorialspoint.com/basics\_of\_computer\_science

ResearchGate, Björn Johansson, A XML sample document including energy consumption from https://www.researchgate.net/figure/A-XML-sample-document-including-energy-c onsumption\_fig3\_221529510

ResearchGate, Mathias Uslar, A sample CIM/XML file modeling idle power (q) and effective power (p) at a corresponding substation, Retrieved from https://www.researchgate.net/figure/A-sample-CIM-XML-file-modeling-idle-power- q-and-effective-power-p-at-a-corresponding\_fig1\_220701370

GeeksForGeeks, What is the difference between MySQL, MySQLi and PDO, Retrieved from https://www.geeksforgeeks.org/what-is-the-difference-between-mysql-mysqli-and- pdo/

Wikipedia, Apache HTTP Server, Feature overview, Retrieved from https://en.wikipedia.org/wiki/Apache\_HTTP\_Server

StackOverflow, What is the difference between utf8mb4 and utf8 charsets in MySQL, Retrieved from, https://stackoverflow.com/questions/30074492/what-is-the- difference-between-utf8mb4-and-utf8-charsets-in-mysql

Techopedia, Cross Platform, What Does Cross Platform Mean, Retrieved from, https://www.techopedia.com/definition/17056/cross-platform

phpMyAdmin, Bringing MySQL to the web, Features, Retrieved from, https://www.phpmyadmin.net

r

# APPENDICES

## Appendix 1 - SQL\_SCHEMA.md

**Markdown file for SQL Schema, Query Format and Sample Query**

1. ### TABLE SCHEMA
3. id
4. title
5. link = guid = atom:link
6. media = enclosure = media:content = image
7. meta\_description = description
8. content = news = content:encoded
9. pubDate

12. ### QUERY FORMAT
14. $query = "INSERT INTO $tableName
15. (title, link, media, meta\_description, content, pubDate) VALUES
16. ($title, $link, $media, $meta\_description, $content, $pubDate);";

19. ### SAMPLE QUERY
21. ```sql
22. INSERT INTO Entries
23. (title, link, media, meta\_description, content, pubDate) VALUES
24. ('Title comes here!', 'https://lintothesource.com', 'https://image.com/img.jpg', 'Meta description comes here!', 'The content goes here!', 'Wed, 01 Jun 2022 10:11:00 +0300');
25. ```

## Appendix 2 - initiate\_sql.php

**PHP file for initiation of SQL Database and Table**

1. <?php
3. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/connect\_mysql.php');
4. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/disconnect\_mysql.php');
5. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/create\_db.php');
6. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/create\_table.php');
7. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/config/sql\_conf.php');
9. $conf = new SqlConf();
11. $dbname = $conf->getDatabase();
12. $tableName = $conf->getTableName();
14. $conn = connectMysql();
16. if (empty (mysqli\_fetch\_array(mysqli\_query($conn,"SHOW DATABASES LIKE '$dbname'"))))
17. {
18. echo "DB does not exist";
19. echo PHP\_EOL;
21. createDatabase($conn, $dbname);
22. }
24. else {
25. echo "DB does exist";
26. echo PHP\_EOL;
28. $conn -> select\_db("$dbname");
30. if ($result = $conn->query( "SHOW TABLES LIKE '".$tableName."'" ))
31. {
32. if($result->num\_rows == 1)
33. {
34. echo "Table exists";
35. echo PHP\_EOL;
36. }
38. else {
39. echo "Table does not exist";
40. echo PHP\_EOL;
42. createTable($conn, $tableName);
43. }
44. }
46. }
48. disconnect($conn);
50. ?>

## Appendix 3 - sql\_conf.php

**PHP file for Global SQL Configuration**

1. <?php
3. mysqli\_report(MYSQLI\_REPORT\_ERROR | MYSQLI\_REPORT\_STRICT);
5. class SqlConf {
7. private $user = "root";
8. private $password = "password";
10. private $hostname = "localhost:3306";
12. private $dbname = "saglikdb";
13. private $tableName = "entries";
15. //USER
16. public function setUserName($user) {
17. $this->user = $user;
18. }
20. public function getUserName() {
21. return $this->user;
22. }
24. //PASSWORD
25. public function setPassword($password) {
26. $this->password = $password;
27. }
29. public function getPassword() {
30. return $this->password;
31. }
33. //HOSTNAME
34. public function setHostname($hostname) {
35. $this->hostname = $hostname;
36. }
38. public function getHostname() {
39. return $this->hostname;
40. }
42. //DATABASE
43. public function setDatabase($dbname) {
44. $this->dbname = $dbname;
45. }
47. public function getDatabase() {
48. return $this->dbname;
49. }
51. //TABLE
52. public function setTableName($tableName) {
53. $this->tableName = $tableName;
54. }
56. public function getTableName() {
57. return $this->tableName;
58. }
60. }

63. //USAGE =>
65. //include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/config/sql\_conf.php');
66. //$conf = new SqlConf();
67. //echo 'DB Name: ' . $conf->getDbName().' Table Name: ' . $conf->getTableName();
68. //echo 'User Name: ' . $conf->getUserName().' Password: ' . $conf->getPassword() . ' Hostname: ' . $conf->getHostname();
70. ?>

## Appendix 4 - ntv.php

**PHP file for Retrieving Data From RSS and Querying to SQL**

1. <?php
3. /\*
4. title
5. published -> pubDate
6. link href=
7. content
8. \*/
10. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/connect\_mysql.php');
11. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/disconnect\_mysql.php');
12. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/sql/insert\_multiple.php');
13. include\_once(dirname( dirname(\_\_FILE\_\_) ) . '/config/sql\_conf.php');
15. function ntv($key, $value) {
17. $context = stream\_context\_create(
18. array(
19. 'http' => array(
20. 'max\_redirects' => 5
21. )
22. )
23. );
25. $conf = new SqlConf();
27. $query = "";
28. $dbname = $conf->getDatabase();
29. $tableName = $conf->getTableName();
30. $conn = connectMysql();
31. $conn -> select\_db("$dbname");
33. try {
35. $content = file\_get\_contents($value.'?'.mt\_rand(), false, $context);
36. $a = new SimpleXMLElement($content);
38. foreach($a->entry as $entry) {
40. $title = $entry->title;
41. $link = $entry->link->attributes()->{'href'};
42. $content = $entry->content;
43. $pubDate = $entry->published;
45. $title = addslashes(htmlspecialchars\_decode($title, ENT\_QUOTES));
46. $link = addslashes(htmlspecialchars\_decode($link, ENT\_QUOTES));
47. $content = addslashes(htmlspecialchars\_decode($content, ENT\_QUOTES));
48. $pubDate = addslashes(htmlspecialchars\_decode($pubDate, ENT\_QUOTES));
50. $result = $conn->query("SELECT \* from $tableName WHERE link='$link'");
52. if ($result->num\_rows > 0) {
53. echo "DATA exists";
54. echo PHP\_EOL;
56. echo "$key $title";
57. echo PHP\_EOL;
58. }
59. else {
60. echo "DATA does not exist\n";
62. $query .= "INSERT INTO $tableName
63. (title, link, media, meta\_description, content, pubDate) VALUES
64. ('$title', '$link', NULL, NULL, '$content', '$pubDate'); ";
66. }
68. }
70. if($query != "")
71. {
72. insertMultiple($conn, $query);
73. }
75. echo PHP\_EOL;
77. } catch (Exception $e) {
78. echo "Exception at $key $e";
79. echo PHP\_EOL;
80. }
82. disconnect($conn);
84. }

87. ?>

## More Codes are Available on GitHub

**All the source codes and detailed documentations are available on a GitHub Repository. To access the repo please visit** [**https://github.com/At1llaG/PHP-RSS**](https://github.com/At1llaG/PHP-RSS)**.**