Report on Database Access and Hosting, and recommended system for the student housing database.

Word Count: 1447

Table of Contents

[Glossary of Terms 3](#_Toc530347073)

[Introduction 4](#_Toc530347074)

[Hosting a Database. 4](#_Toc530347075)

[Methods to access a database 5](#_Toc530347076)

[Recommended System 6](#_Toc530347077)

[References 9](#_Toc530347078)

# Glossary of Terms

API – Application Programming Interface, allows creation of programs that access features of an operating system of program.

Client – Computer or workstation that gets information and applications from a server.

CSS – Cascading Style Sheets, used for describing the presentation of a web page written in HTML.

Extranet – Companies intranet available to outside companies, customers and staff.

HTML – Hypertext Mark up Language, used to create web pages and web applications.

Intranet – An internal network accessible only to a company’s staff using internet style communication.

JAVA – A general purpose programming language

LAN – Local Area Network

PHP – Server side scripting language used for web development.

Scripting Language – A language used to create programs for a specialised environment.

SQL – Structured Query Language, the language used to ‘talk’ to databases

VPN – Virtual Private Network, allows a private network to be accessed over a public network, such as the Internet.

# Introduction

This report provides an overview of how a database can be hosted and accessed, and the technologies required to achieve this, followed by a detailed system proposal.

In many businesses data is shared and accessed remotely from multiple locations. To provide these services a database needs to be hosted (usually on a server) and access to the server provided. As well as access to the database from a desktop computer, access can also be delivered for other, portable devices, such as smartphones and laptops. This is provided by using either a network connection (wireless or wired) or over the internet, both of these use a similar way of accessing the database, with small differences between them (Kroenke, 2005).

# Hosting a Database.

Connecting to the database follows the client-server model that is akin that from computer networking. The setup is the same, a server providing services to client computers, but with the addition of the database server. Whether the database is accessed over the internet or network, the server remains the controller of the database, better known as the backend of the system. This is also known as a two tier model. This provides access to the database for many PCs/workstations, all connected by a network (Wide Area Network (WAN)/ Local Area Network(LAN)). The client is the PC and provides user interface and local processing. The server provides functionality that the client does not have, including access to the database (Ward, 2008). This model allows connection to the database from computers located on the same network as the database. This access can be on site directly or via an intranet; or over the internet via VPNs and extranets.

An alternative model for hosting a database is known as the three tier model. This has an extra level between the client and the server. This extra layer is known as an application server and provides the client with database access by requesting resources from a server or many servers. This type of hosting is prevalent when connecting to a database over the internet via a web browser.

Connecting to a database over the internet there includes differences when compared to connecting over a LAN. During connection a web browser is used (on the client side) which understands HTML, this then connects to the database using web servers, that use dynamic web pages in HTML that are returned to the browser (Ward, 2008). HTML web pages are updated from the server so that updates to the pages can be applied (Kroenke, 2005). Although HTML offers only a limited feature set, the use of scripting languages, such as PHP and Java, allows for a better user experience by allowing for improved display, form creation and access to the database. Information is sent from the web browser to the web server, from there to the application server and finally the database server.

Equipment for hosting a database can be housed at the University using servers, database software, scripting languages and staff knowledge. Also companies that rent out equipment and support for a monthly fee. Amongst these are companies that provide a web-based solution for hosting a database (with all the equipment housed at their end) and allows users to upload web pages and database information to their servers.

# Methods to access a database

To access a database, a ‘front-end’ is used. This is an interface that is used to fetch and display the data stored on the database (Taylor, 2001). The front-end works separately from the database (or back-end) and connects to the database through an API (application program interface) or direct access through a secure connection (Taylor, 2001). Either of these connections uses a client/server set-up.

The front-end or client software is not dependant on the database type (for example MySQL or Oracle), this provides different opportunities for connecting to the database server. A front-end can be an application for a desktop computer, laptop computers or mobile devices, and also over the internet as a web page.

The front end pertains to everything that the user of a database sees, and this includes designing screens (if on a web-page using HTML and CSS) and programming (languages such as Python, and scripting languages such as JavaScript) to provide a way for the user to send instructions to the database server, such as searching for data, creating views (or structured reports) and adding records to the database.

# Recommended System

For hosting the database I suggest a web based solution hosted by an external company on virtual servers known as a ‘Cloud Database’ or Database as a service. Justification for this decision is due to the infrastructure for the database is provided by the hosting company negating need for buying equipment and finding a location for it. This is more economical due to less capital expenditure, avoiding increased staffing and extra electrical costs (from heating, ventilation and air conditioning) (Rouse, 2017). Database maintenance becomes the responsibility of the hosting company.

Additional benefits from cloud hosting include, quick scalability allowing for more bandwidth and database capacity to be added quickly, performance guarantees through a service level agreement, better failover support – uninterrupted access to the database is provided and if the companies servers should fail then their backup servers would takeover. Providing database hosting through the cloud can help simplify connections to the database over the internet (Rouse, 2017). The hosting company will also provide security for the database, including Firewall protection, Intrusion detection, Antimalware protection, Vulnerability scans and encrypted storage, backups and VPN (Virtual Private Network). They also provide user authentication and user access levels.

The database solution I propose uses a collection of utilities known as a Software Stack. The stack I have selected is known as WAMP (Windows/Apache/MySQL/PHP). Windows signifies the operating system, Apache the server software, MySQL the database and PHP the scripting language for applications. Dynamic web pages will be incorporated to create a front end for the database so it can accessed over the internet. Web pages use two languages to provide content, HTML (Hyper Text Mark-up Language) which describes the structure of a web page and CSS (Cascading Style Sheets) which describes the presentation (colours, text size etc.) of a web page.

This stack is mostly located on the virtual servers, except the PHP element. On our end we will have to deliver the database schema and data, and any web-pages and scripting. These are then uploaded to the server which uses containerisation to run the database. Containerisation is a type of virtual server that allows an application to be put into a container with its own operating environment. This in essence creates a database with lower overheads for the operator and allows us to change only the containers we need.

I chose the Windows operating system because it is highly recognised and widely used in a business environment and although with containerisation it will not have an impact on the end users, it will make explaining what is happening easier.

Apache is the server application for the database, and I chose this because it is an open source solution, meaning that it is free to use, and has a large support community. Again, with containerisation, we will not have to manage the server software but we would have to pay for any licences for other server applications.

MySQL is the database software. I selected this because it is open source and has the benefits mentioned for Apache. It uses SQL to query the database, and would require knowledge of this along with a scripting language to access and manage the database itself (Creating tables, adding records, extracting data).

PHP is the scripting language that will be used to make the web pages more user friendly rather than using HTML and CSS alone. As the database query language (SQL) can be imbedded into PHP, the database can be accessed and maintained through the web page (Banerjee, 2009).

When using PHP there are several security concerns that if not addressed can leave the database vulnerable. SQL injection is an attack on a web page in areas that allow user access, where a line of SQL code is inputted that could allow database access and allow attackers to view data, amend data and create and delete tables/data (Shirey, 2012). This can be prevented by making sure all SQL statements are validated, verified and cleaned before being entered into the database (Shirey, 2012). Other risks of PHP include remote file inclusion and cross site scripting. Using PHP directives properly can prevent remote file inclusion, and escape functions in PHP code to ignore characters that signify HTML or Javascript codes (such as < or >) during a cross scripting attack (Shirey, 2012).

# References

Banerjee, P. S., 2009. *A Note on Web application with Reference to ASP.NET and AJAX.* [Online]   
Available at: https://www.codeproject.com/Articles/37504/A-Note-on-Web-application-with-Reference-to-ASP-NE  
[Accessed 05 11 2018].

Gundavaram, S., 1996. *CGI Programming on the World Wide Web.* [Online]   
Available at: https://www.oreilly.com/openbook/cgi/ch01\_01.html  
[Accessed 2018 11 09].

Kroenke, D. M., 2005. *Database Concepts.* 2nd ed. New Jersey: Pearsoon Education Inc..

Rouse, M., 2017. *cloud database.* [Online]   
Available at: https://searchcloudapplications.techtarget.com/definition/cloud-database  
[Accessed 17 11 2018].

Shirey, D., 2012. *Top 10 PHP Security Vulnerabilities.* [Online]   
Available at: https://www.sitepoint.com/top-10-php-security-vulnerabilities/  
[Accessed 18 11 2018].

Taylor, A. G., 2001. *Database development for dummies.* Foster City: IDG Books Worldwide Inc..

Ward, P., 2008. *Database Management Systems.* 2nd ed. London: Middlesex University Press.

WODEHOUSE, C., N.D. *Choosing the Right Software Stack.* [Online]   
Available at: https://www.upwork.com/hiring/development/choosing-the-right-software-stack-for-your-website/  
[Accessed 07 11 2018].