

KG1902428 Database Driven Website Report

by Kallum Gordon

FILE	17858_KALLUM_GORDON_KG1902428_DATABASE_DRIVEN_WEBSITE_REPO RT_294270_104086850.DOCX (81.62K)		
TIME SUBMITTED	25-JAN-2021 06:55AM (UTC+0000)	WORD COUNT	1098
SUBMISSION ID	143255691	CHARACTER COUNT	5860

Databases and Application Development

Database-driven website

Author: Kallum Gordon

Prepared for: Gaynor Thomas

10th January 2021

1.0 Introduction

A Dungeons & Dragons character website has been produced which communicates with a database and phpMyAdmin using php. From the website the user can search and view records, or enter their own record which is then entered into the database.

Table of Contents

1.0	Introduction	2
2.0	The website	3
3.0	The Database	3
4.0	Security features within the website	3
4.0.1	conninfo.php	3
4.0.2	SQL injection and prepared statements	3
4.0.3	Additional prepared statements	4
4.1	Security features that should be implemented if the website went live	4
4.1.1	MyAdmin password	4
4.1.2	A login page	4
4.1.2	Captcha challenge	4
4.1.3	Buffer overflow attack prevention	4
4.1.4	More prepared statements	4
4.1.5	SSL (secure sockets layer) encryption	4
4.1.6	Spambots	5
5.0	References	5

2.0 The website

The website should be opened on index.php and provides a basic UI and purple background for navigation. From any page the user can select to search for data, or enter a new record.

3.0 The Database

In total, the characters table has 17 sections and 20 pre-entered records. The classes table has two sections and 16 classes to select from. This could be expanded upon to allow the user to enter their own classes in the future and also provide the ability to multi-class, which is a key D&D feature.

4.0 Security features within the website

4.0.1 conninfo.php

The connection information file establishes admin information and prevents just anyone accessing the database information. It connects by checking the server, the username, the password, and the database. It will produce an error message if the connection failed.

4.0.2 SQL injection and prepared statements

SQL injection (injecting code) could occur within the search bar of the database. Entering a special string such as 'OR 1=1 --' will print out all of the records found within the database. [1] This is a huge risk and a breach of privacy of all users whose records are contained within the database and can be taken to a court of law. [2] This has been combatted using prepared statements on all of the radio button options. Therefore, searching for relevant information works, but using SQL injection to pull all of the records at once does not.

4.0.3 Additional prepared statements

Prepared statements have also been handled on the side of phpMyAdmin. Critical fields have been flagged so they cannot be left null (empty). This prevents fields being left empty that need to be filled to provide the bare minimum amount of information to be able to analyse the entered D&D character and use it in a game.

4.1 Security features that should be implemented if the website went live

4.1.1 MyAdmin password

A simple one on the side of phpMyAdmin. It is ideal to change the username and password so they aren't common. For the best results the password should be randomized and adhere to the generally accepted tried and true long password which includes random letters, capital letters, numbers, and symbols. This relates to my website as this is especially important if the website was actually live as it would be very easy to crack.

4.1.2 A login page

This is a critical part of a website like this as it protects from unverified users. This is the first thing to be done given more time. This relates to my website as users will be using it to enter records to a database and the database could easily be destroyed if random users are left to their own devices.

4.1.2 Captcha challenge

A captcha challenge could be implemented to challenge those who want to add new data to the database from the website. This would theoretically catch bots out and mitigates DDoS attacks. However, it can be considered intrusive which can result in a negative experience for the user. This relates to my website as bots could attack it and bring it down if it went live. [3]

4.1.3 Buffer overflow attack prevention

A buffer overflow attack is theoretically possible and would overflow the buffer limit to insert malicious code to overwrite the stack. This is low-level programming. This can be solved by using no-op operations.

4.1.4 More prepared statements

Prepared statements could be added when entering data into the database. It has been done when searching for information but more could be done to ensure total protection. This relates to my website as data security is important, legally and personally. It also protects data which is critical as having the data be deleted due to SQL injection would mean the database would have to be developed again from scratch or using an existing backup.

4.1.5 SSL (secure sockets layer) encryption

SSL encrypts data transmitted through the internet. Attackers who intercept the data will see random undecipherable characters. To transfer the data it commences a "handshake" between two devices ensuring they are the intended recipient. SSL then digitally signs the data, ensuring it is not intercepted before reaching its intended recipient. This is relevant to my website as it would prevent data from being transmitted in plaintext that could be intercepted. This is especially important in a website that communicates with a database containing personal information and potential login information. [4]

4.1.6 Spambots

Spambots harvest information found on the internet and keep it for nefarious acts such as sending spam to harvested emails, or selling emails on to interested parties. These can be combatted by using methods such as munging, where an email is modified so that a human can read it, but a spambot cannot. Spambots can also be used to submit irrelevant information to a database which can overload it. [5] Another way to prevent spambots is requiring the person submitting information to the database to verify their credentials via email. Traps can also be set up for spambots by including a link to a bot trap website within your website, the goal of these websites is to overload a bot with thousands of unused email addresses.

5.0 References

- [1] w3schools, "SQL Injection," w3schools.com, [Online]. Available: https://www.w3schools.com/sql/sql_injection.asp. [Accessed 10th January 2021].
- [2] Computerphile, "Running an SQL injection attack," Computerphile, 15th June 2016. [Online]. Available: <https://www.youtube.com/watch?v=ciNHn38EyRc>. [Accessed 12th January 2021].
- [3] RED BUTTON DDoS Experts, "CAPTCHA," RED BUTTON DDoS Experts, [Online]. Available: <https://www.red-button.net/ddos-glossary/captcha/>. [Accessed 15th January 2021].
- [4] Cloudflare, "What is SSL?," Cloudflare, [Online]. Available: <https://www.cloudflare.com/en-gb/learning/ssl/what-is-ssl/>. [Accessed 15th January 2021].
- [5] Cloudflare, "What Is a Spam Bot? | How Spam Comments and Spam Messages Spread," Cloudflare, [Online]. Available: <https://www.cloudflare.com/en-gb/learning/bots/what-is-a-spambot/>. [Accessed 15th January 2021].

KG1902428 Database Driven Website Report

GRADEMARK REPORT

FINAL GRADE

55/100

GENERAL COMMENTS

Instructor

WEBSITE

Database structure and design 10/15 - Generally good for the characters/classes normalisation, another field you might want to normalise is physique, as you have several repeating values in that field. You don't have a log in so have not included a user table.

Range of PHP code demonstrated

Search is good and it is possible for the user to search on multiple fields. Add a record is OK, although there is no validation - you can just press submit and a blank record goes to the database. Good use of pull-down menu for the classes. The rest of the implementation is limited, you have not included login, edit or delete 20/40 -

Overall integration of website -

Website is difficult to read with a very dark purple background. Use a lighter colour to assist with this. The layout is very basic, although you have used an include file. You could include a few pictures, centre the text etc to make it a nicer layout. Also you could line up your form entry text boxes and so on. 7/15

REPORT

Security issues identified 10 /15 -

You have identified the main issues such as SQL Injection/Spambots etc and it is good to see that you have researched a little beyond what was covered in class

Consideration of security issues for website 8/ 15 -
Good consideration of prepared statements and their use, You do not always link the issue identified to your website - for example, do you think the website you produced is vulnerable to spambots - you briefly mention that bots could attack it - a little more is needed here - which pages are vulnerable do you think? Do you need to implement SSL on your website?

Overall a reasonable attempt, although the website functionality is fairly limited.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5
