The Project

The Payment Card Industry Data Security Standard (PCI DSS) was developed to increase security and protection on payment card account data as well as globalize data security. PCI DSS outlines technical and operational requirements necessary to protect account data. On top of that, PCI DSS also can provide assistance in protection against threats and keep all elements in the payment ecosystem secure. The goal of our compliance quiz is to give small businesses the opportunity to quickly test their compliance with PCI DSS guidelines in a simple and easy-to-understand format. The compliance quiz seeks to help small businesses identify any noncompliant aspect of their business so that they can address compliance issues before problems arise. This quiz is not a replacement for the full PCI DSS guidelines and participants should refer to the full PCI DSS guidelines if they have questions or want clarification on any of the guidelines.

Guidelines
https://docs-prv.pcisecuritystandards.org/PCI%20DSS/Standard/PCI-DSS-v4_0.pdf

Project Overview

Requirements

- A machine to run the server
- Apache
- MariaDB
- PHP
- (optional) phpMyAdmin

Summary

The CyberGuard360 dynamic quiz utilizes the LAMP full stack (Linux, Apache, MariaDB, PHP) with the addition of phpMyAdmin as a frontend to manage the latter two aspects. In the development of this project, the LAMP server was a Raspberry Pi running Raspian without a desktop environment. GitHub was used to make changes to the source code and push changes to the server. To access this, SSH was used.

Possible Issues

- If the server is going to be set up at a remote location, setting up port forwarding is important to allow for SSH and accessing the website and phpMyAdmin

- One issue we ran into was that the server was only accessible when the router firewall sent traffic to the server itself instead of just the IP address. This will most likely depend on the router and firewall used.
    - A good way to check if the forwarding for the server is working is to use https://portchecker.co/checking

Apache

Summary

Apache was rather easy to set up. It is designed to work right out of the box with little to no configuration. The only changes possibly needed are changing the apache config to listen out for a domain name if utilized.

File Layout

For the file layout of the website, we used the MVC (model view control) design pattern. The model deals with the database calls and page controls, the view handles the visuals of the website, and the control is the main index that runs everything. This allows for ease of use and modularity of the website.

Possible Issues

- The website may not be accessible if not port forwarded correctly. Apache uses port 80 by default. If your router does not allow you to forward port 80, a popular alternative is to configure Apache to use port 8080 and forward that port on your router instead.
    - A good way to check if the forwarding for the server is working is to use https://portchecker.co/checking
- If you are using a domain instead of your public IP address, you may need to configure Apache to accommodate the domain

MariaDB

Summary

MariaDB is one of the most important aspects of this project. It handles login information, the questions presented to the user, answer storage, and survey tracking. The four tables in the database are as follows:

- Accounts: Stores login information of users to allow them to log in. This table is referenced by the survey table to keep track of what surveys belong to who.

- Answers: Stores a user’s answer to a given question. This table references the survey_id and question_id columns of the survey and questions tables respectively
- Questions: This is where the questions are pulled from and presented to the user. They are categorized by guideline and have yes_id and no_id fields to determine where the user goes after they answer yes or no to a question. This table is referenced by the Answers table to determine what question a user’s answer corresponds to
- Survey: Keeps track of Surveys for every user. Mainly utilized by the Answers table to keep track of answers. This table references the account_id column of the accounts table to give the survey ownership.

Possible Issues

- If the database is being hosted remotely, computers running stacks such as XAMPP to assist in development may struggle with connecting to the database. This may require a few changes
    - Instead of using localhost, you may need to use the –bind-address option in the config to use 0.0.0.0 or your private IP address to allow port forwarding. Your router firewall will also need to be configured to port forward 3306.
        - A good way to check if the forwarding for the server is working is to use https://portchecker.co/checking
    - The MariaDB account being used to access the database may need remote permissions. This can be done by granting permissions to the account and referencing as `'username'@'%'` with % acting as a wildcard for the IP address.
- Perceived database errors may also be a result of improper PHP code or incorrect MariaDB queries from the database model php files. See below for solutions.

PHP

Summary

PHP is the main language used for this project. It controls the flow of web pages as well as encapsulates the html. As mentioned here, the project structure follows the MVC design pattern.

Possible Issues

- PHP programming errors can be hard to find if you are unfamiliar with using it. If you are editing and testing the code on your machine, utilizing a program such as xdebug can prove beneficial
- Adding `[PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]` to the PDO object can also give you more information about any problems that may occur during testing

phpMyAdmin

Summary

While not required, phpMyAdmin is a great tool for viewing and modifying databases as well as MariaDB accounts and their permissions. If you are unfamiliar with commands for MariaDB and want an easier way of looking at and modifying the database, phpMyAdmin is great. It also allows for easily batch adding entries into tables in addition to being able to run commands from the website.

Possible Issues

- It is not recommended to install phpMyAdmin before MariaDB or PHP, as this can cause issues in the setting up of MariaDB. It is best to install phpMyAdmin last after you have installed everything else.