Brief for Website Development

Client: Rhys Davies

Problem

Rhys' is a 15, and is currently teaching himself digital music production. We both agreed that a site where he could collate and advertise his music/albums would be advantageous

Solution: Create a website with a functional CMS, Music/Album gallery, and basic blogging capabilities

Stakeholders

- Rhys Davies (Client)
- Me
- Rhys' parents (to an extent)

Research

- Content Management Systems
- Music streaming
- Efficient resource storage online (blobs/Binary Large Objects)

Notes

While there is a standard for a relation database, the DB implemented for this will be only present, really to get the credits, as MongoDB is far better suited for storing blog-posts, and binary data than a relation database, which is ideal for values traditionally representable in a table. This system is not ideal for stuff with nested values (I'm aware of embedded tables).

The NoSQL database will be used to store blobbed picture files, post pages, and binary data. Users will be stored in a MySQL db, solely for the purpose of the database standard.

Context Consideration

How is the client going to keep the website up to date

The website will have a basic CMS implemented, which will primarily be focused on adding music, with short blog-type posts in-between. There will need to be functionaly for albums, album art, ID3 tags and other music related functionality

How long is the website going to last

Not sure at this point, but likely for at least a couple of years. Rhys is currently a year older than me, and is unlikely to need a new site.

How often does it need to be updated

Not too regularly, but Rhys may use it for blogging type updates.

What does the target audience need from it

Access to well collated and organised music that Rhys has made, as well as giving Rhys a decent jumping off point into the industry

How to test it/keep info private

There won't be much client-side stored info, but the normal practices will still hold true. I will (try) to get an SSL certificate using LetsEncrypt, so that information from my site can't be used in CSF (Cross Site Request Forgery), or other attacks primarily relying on unencrypted communications. As there won't be user-accounts, except for the administration side of things, I'll just need to make sure that the system is secure