MyCloud

<p>

MyCloud is a project I’ve create for use at school. It let’s me access my media and documents where ever I am and is great to store my notes on when changing between computers and devices. It is a web-based application with a LAMP server backend with all files and data being held in the MySQL database.

</p><p>

As well as documents, MyCloud has the ability to hold, play and share music and photos, it had an inline music player which could be used simply by selecting an artist and pressing play, songs are automatically buffered and played one after the other.

</p><p>

Photos can be sorted by albums and shared with a URL, if enabled, to someone without access to your user account. Documents could also be shared the same way.

</p><p>

I abandoned MyCloud because it was a little slow in the backend and resource intensive, although I learnt a lot about web programing while creating it, it still had its memory leaks and I deiced I would be better off recreating the abilities of my website through my new projects.

</p><p>

I wrote MyCloud within 10 days, it was a challenge to see if I could make something like this within a week, which I did, but after using it for a day or two I decided to add iPhone and iPad compatibility, along with more sharing options, The web application was all ios accessible, and also had a apple webapp that could run on the devices as an application.

</p>

Travlr 3

* 2 databases
* server sent events
* framework
* server, caching file metadata

<p>

Travlr3 is a web application platform, built to provide a framework and interface for web apps written in PHP and Node.js. It manages user accounts, user permissions, administration, communication with the client, windows and many more aspects of an application, which allows all the apps it contains to have a standard structure and function.

</p><p>

The platform can be run 3 different ways, firstly in a browser, the entire platform itself has a HTML, CSS, Javascript front end. Secondly as a Mac application, with the Webkit rendering engine. Lastly I created a striped down Linux based operating system that could run the same way.

</p><p>

Applications had to be placed from the internal applications directory and have some metadata put into the database such as the app title, icon and app identification code, used to refer to the app between client and server. This means faster lookups for applications.

</p><p>

Applications all have access to the main framework and its resources and functions, this includes 15 JavaScript functions and 18 PHP classes, these both provide easer methods for regularly used objects and functions in web apps and also links applications into the system.

</p><p>

The available classes contain objects that can communicate with the desktop UI via server sent events, a HTML form object that performs validation and prevents brute force attacks against forms in the applications and system, database objects, cashing methods and many more

</p><p>

Windows are a critical part of how Travlr 3 operates, when an application is launched the launch function within its app.php file is run, which creates a new window object, applications are allowed to have as many windows as they want and an application will be “running” as long as any windows are open.

</p><p>

Application windows can communicate with each other – every window is given an object, this object contains all the information about it, the size, title, hidden state, url, and extra data attached to the window. These objects can be accessed by any scripts from the same application and can be used to attach data to other scripts.

</p><p>

An application can alter its own windows using functions from the window objects, for example. getWindow(‘confirmDialog’)->close(); From any script within the application will close the confirmDialog window that same application has created, all necessary communication to the client is sent through server sent events.

</p>

<p>

There are two different database platforms used for Travlr3, MySQL and SQLite. All write heavy situations, such as window data and running applications are stored in the MySQL database where as the data that is not altered quite as often lives in a SQLite file database where read operations are very quick.

</p><p>

A user’s personal files do not live in the database; they are stored in a file system directory, this is for two reasons, firstly: lookup speed, although the lookup speed of MySQL is very quick it bogs down quite quickly with large sets of data and many rows, a file system does not slow down with larger files and its hierarchical structure allows easy lookups.

</p><p>

The second reason for using a physical file system for files is for easy access to a files ID3 tags, using the PHP ID3 class library Travlr analyzes each new file and then stores the tags for each file in the database, this allows searching by ID3 tag without having to analyze each file, which is quite time consuming. For example: you could search all of your files for and song by a specific artist and the files would all be found within the time it takes to do a small database lookup.

</p>

Sales

<p>

The sales web application was created for use in an in school assignment for commerce. In groups we had to create a product or service and sell that product or service over a week.

</p><p>

We were told only one hour before having to start sales that we were to keep record of all sales. We were selling food products and knew that it would be difficult to keep track of everything as we sold it.

</p><p>

So I created a simple web app that we used on a few iPods and an iPad, which helped us, keep track of our sales and stock. As we made a sale the associated item just had to be tapped on one of the devices.

</p><p>

At the beginning of each day the “start” price was set to our current profit, we could then see how we were going though out the day. Every sale was stored with a precise time and date. There was also a counter for each product.

</p><p>

At the end of our task we had to canalize the sales that we received, this was simple as the application had an SQLite database file sitting with all of our group’s sale information.

</p>

School bookings

<p>

I was unable to have images for this project as the style sheets are hosted by the school and are being changed, breaking the website. I will add images when available. I built this system when I was in year 7 and my mate in year 8 did video tutorials for it.

</p><p>

The school bookings system was a website created for my school. It was an online booking system for the parent teacher interviews, which involved logging in and selecting the timeslots wanted to see each teacher. This allowed people to book times with full knowledge of what times are available and also took all the work off the teachers as before the night they simply logged in themselves and printed off a sheet of who is coming when.

</p><p>

The website was written in PHP with a MySQL database, it used the schools website as a base for the front end so it had the same look and feel as the rest of the website.

</p><p>

The way interviews were organized before hand were individually with each teacher and a sheet of paper, going around and finding available times. It was a great step forward for the school to have a digital replacement for this method.

</p><p>

The website had to include a way for teachers to set certain times that they would not be available on the night. For example dinner breaks or not being able to be there on time. This was included and users were able to easily see when there were times available.

</p><p>

Both teachers and parents are able to print off a schedule of the night once the bookings have been completed.

</p>

Calculator scripts

<p>

I created a few scripts throughout junior school as we covered certain topics in advanced maths. Two of these were the prime and Fibonacci number tester and generators. These either gave you the nth number in the sequence, or checked to see if the number fit in the sequence, depending on what “mode” you were on.

</p><p>

Another script I created was for circles, it had text boxes for area, diameter, radius and circumference and if the user filled in one of the boxes the rest were filled with the appropriate numbers.

</p>

Travlr1

<p>

The first “travlr” was a project that I had been thinking about for a year or so before I started building it. I had always liked the idea of a centralized place for my web applications, as I had started to built up quite a collection and keeping track of everything was getting difficult.

</p><p>

You may notice that there was no “Travlr2”. Travlr2 was a very short lived project, soon into designing and creating it I had decided to rebuild things from scratch, it was slow, inefficient and overall not worth continuing work on. I started fresh with Travlr3

</p><p>

The plan was to have a system that managed files, updates, security and UI of web applications. A larger web application that contained and ran all others, this was in the form of icons on the “home page“ along with a utilities page which managed plugins, colours etc.

</p><p>

Plugins were applications but instead of having a UI that was launched from the home page they had scripts that ran in certain situations, an example plugin was a pass wording script, which allowed you to choose applications that required extra authentication and set a master password for these applications. Another was one that changed the main background colour of the system.

</p><p>

Updates of both web apps and the travlr system itself were all managed, looking at external servers to check if new versions were available and installing them when possible.

</p><p>

There was a file browser included which allowed the user to view, delete and upload files. Along with opening a file in another application (given the app supports this.) This allowed the apps to be used not just as stand alone web apps but to really communicate and work together.

</p><p>

This system was used without any specific database server or platform. All information was stored in flat files. Searching the applications directory and verifying the applications configuration files found applications when needed. This meant very slow operation, every time a page was loaded or new data was fetched many file system operations had to take place. Dramatically slowing down operations.

</p>

SiriProxy port

<p>

You may have heard of siri proxy. It is a ruby application, which intercepts communication between the iPhones “Siri”, and apples servers used for the text to speech service. This allows custom responses to the voice control software.

</p><p>

I loved the idea of this, and spent some time playing around getting it to work on my home computer and phone then creating some scripts that interacted with my computer like “open \*somewebsite\* on my computer”. This was great fun the only issue was that I did not know ruby well enough to be able to do anything complex with it.

</p><p>

I decided to create a framework for PHP, which allowed it to do the same thing the ruby script, was. This involved modifying the ruby application to forward requests and responses onto PHP scripts with all the appropriate information, and creating functions for the PHP scripts to handle responses.

</p><p>

Once complete I was able to create much more advanced responses and tasks for my phones Siri, This was a project I very much enjoyed doing.

</p>

Welcome,

My name is Jye Lewis and this is my portfolio of work. Under the projects menu you will find some of my personal projects with information and images. Under websites there is a list containing some of the websites I have worked on.

About

<p>

My name is Jye Lewis, I am 15 years old and live in Sydney, Australia.

</p><p>

I have used and am competent on Mac, Windows and Linux machines, I have been programing and building websites since I was 7 years old and can fluently code in PHP and Node.JS/JavaScript. I can cutup websites well and quickly. I started learning PHP at 7 and Node at 15 (although I already had a strong understanding of JavaScript.) I also know python though do not tend to use it as much as the other languages. I have a good understanding of the bash CLI and its associated applications; I can and have setup Linux and web, file and database servers.

</p><p>

I work a part time job at <a href=”http://www.digeratisolutions.com.au/”>Digerati Solutions</a> where I cutup websites, examples are under the websites menu. I have been working at digerati since the beginning of 2012.

</p><p>

I have used multiple different SQL based database systems (including MySQL and Microsoft SQL Sever) and also some noSQL servers (including MongoDB and Redis)

</p>