Pre-coursework: M.V.C design

M.V.C design

M.V.C stand for *Model View and Controller* Far from popular belief M.V.C is not a framework but a design pattern. It provides a way to structure our applications to allow us to process data in a certain way.

Most Frameworks out there use the M.V.C patterns in their design. These Frameworks include

1. Django
2. Ruby on Rails
3. Cake PHP
4. Laravel
5. Zen Framework
6. ios

Website flow

When Interacting with a website several features come into play.

1. The **Client** - This is your Local Browser ie *Chrome* / *FireFox* / *Internet Explorer*. Let's take an example we put in the Moringa School URL to our browser URL tab. The browser sends a request to a server that hosts the Moringa School Website.
    1. The **Server** - The server is a connector to the client and the data source. It does not store any information but sends responses from the data source back to the client.
    2. The **Database** - This is the long-term storage location for our website data. It gets queries from the server and relays information that it gets.

M.V.C architecture

1. The **Model** - This represents anything that interacts with the database. This involves

    a) Adding and retrieving data to and from the database.

    b) Processing Data from that comes to and from the database. The Model only communicates directly to the **controller**.

2. The **View** - The view represents anything the user sees on the *client*. This is in form of plain HTML, CSS and JavaScript. It displays information it gets from the Controller.

3. The **Controller** - Considered the middleman in the structure it is responsible for handling and processing *GET* and *POST* requests made by the User. It also interacts with the model to retrieve and send information to the database. It's also responsible for telling the views what to do and display.

The M.V.C pattern is useful to allow us, developers, to prevent repeating ourselves and provides us with a structured programming flow.