Una arquitectura está basada en un conjunto de tecnologías sobre la cual se va a construir nuestra aplicación. Hasta hace unos años la más predominante era **LAMP**, compuesta por **Apache** como servidor web, **MySQL** como gestor de bases de datos y **PHP** como lenguaje de programación, todo ello ejecutándose en un sistema operativo basado en **Linux**. Estas tecnologías están muy enfocadas a desplegar y servir aplicaciones web rápidamente, por lo que consiguió captar a la mayor parte de la comunidad de desarrolladores web. También contribuyó a su éxito que muchos CMS populares como Wordpress o Drupal para los blogs, u otras tecnologías como Mediawiki, Moodle o vBulletin utilizaran principalmente este *stack* de tecnologías. También existe una variante llamada **WAMP** para entornos **Windows**.

Sin embargo, los tiempos cambian y cada vez aumenta más la demanda de aplicaciones reactivas que se actualicen en tiempo real. PHP desde los comienzos fue pensado para ser un lenguaje que sirviera una página web entera cada vez que llegaba una petición, y eso sumado a su naturaleza síncrona lo hace un lenguaje demasiado pesado para aplicaciones en tiempo real.

En los últimos años una de las arquitecturas que más auge ha experimentado es el *stack* **MEAN**, compuesta por **MongoDB** como base de datos, **Node.js** como entorno de ejecución del lado del servidor, **Express** para facilitar tareas tediosas como el enrutamiento y **Angular** como *framework* que incorpora el patrón MVC y que acelera la creación de aplicaciones reactivas. Node.js lleva JavaScript al lado del servidor, convirtiéndose en una alternativa fantástica dada la naturaleza asíncrona del lenguaje, y su capacidad para manipular la información rápidamente. También es relevante mencionar lo bien que se complementa con Mongo, ya que los datos son almacenados en una notación aparentemente idéntica (utiliza BSON que es una especie de JSON binario, pero la forma de trabajar en la práctica es la misma).

Por mencionar otra tecnología que está adquiriendo peso en los últimos años, reseñaría a **Go**, un lenguaje de programación que ha ganado mucha fuerza en la programación web del entorno del servidor. A diferencia de PHP y Node.js, Go es un lenguaje compilado con una sintaxis similar a C y Python, pero con un enfoque mucho más moderno. Una ventaja respecto a Node.js es que puede distribuir los procesos en diferentes canales, en vez de tenerlo todo en un único hilo de ejecución. Esto se nota especialmente en el rendimiento de tareas más complejas que implican más recursos del servidor, haciéndolo un lenguaje increíblemente veloz.

Para el siguiente apartado he seleccionado el *framework* **Laravel**, uno de los más conocidos en la comunidad de PHP. Esta tecnología viene acompañada de todo un ecosistema que facilita enormemente la creación de aplicaciones con este lenguaje, y establece unas directrices bastante claras sobre cómo se debe escribir código. Para el manejo de datos utiliza su propio ORM, llamado **Eloquent**, que facilita la creación de **modelos** (la M de MVC). Con un ORM es posible tratar los datos de una base relacional como objetos.

Para las **vistas** (la V de MVC), Laravel incluye un motor de plantillas llamado **Blade** que estiliza el código HTML permitiendo incrustar los datos pasados por el controlador y otras plantillas. Por otro lado, los **controladores** (la C de MVC) se pueden instanciar fácilmente gracias a un potente enrutador que permite vincular rápidamente las rutas con su respectivo controlador.

En definitiva, este framework proporciona una estructura de proyecto bien definida, que aligera la creación de aplicaciones web, pudiendo obtener resultados de forma más rápida. Es por todo esto y la buena reputación que se ha labrado Laravel por lo que lo he elegido para comentarlo. Este framework es ya uno de los más usados en el desarrollo web, y con el reciente lanzamiento de Laravel 6 parece que va a seguir siendo así durante un buen tiempo.