Proyecto de software 2017
Informe final

Integrante: Elías Biagioni

1. Fundamentación de la elección del framework.

El framework elegido para el desarrollo del sistema fue Laravel. Fundamentalmente, la elección se realizó debido al gran uso actual del mismo para el desarrollo en PHP, obtenido de un ranking de usos de frameworks en 2017/18, y la gran cantidad de facilidades que ofrece a la hora de programar y por lo tanto, la notable reducción y claridad a la hora de escribir código.

En un principio empecé a informarme sobre el desarrollo usando dicho framework y Symfony, el cual estaba posicionado segundo en el ranking. Laravel utiliza dependencias de Symfony y otros frameworks, por lo que el primero me pareció muy completo y empecé a investigar más en profundidad sobre este.

2. Referencias de donde se sacaron materiales.

Laravel posee una documentación online muy extensa, donde explica, muy claro y detalladamente, cada una de las funcionalidades que posee. También posee muchos ejemplos. De todas formas, el puntapié inicial lo di mirando un tutorial de YouTube (codigofacilito), donde el autor del video te presenta el framework desde cero, describiendo cada uno de los directorios y componentes más importantes del mismo. Lleva a cabo del desarrollo de un blog y, en paralelo, te guía por la documentación para empezar a saber en dónde buscar en caso de tener problemas o para realizar una programación correcta y aprovechando al máximo las facilidades del framework. También, en algunos casos donde ocurrían diversos errores, utilice el sitio stackoverflow para ver la resolución de los mismos, ya que eran errores con los que se habían enfrentado otros desarrolladores.

3. Módulos realizados durante la cursada, aprovechados en la utilización del framework.

En general, se volvió a escribir la mayoría del código ya que, como se mencionó antes, Lavarel proporciona componentes que facilitan el desarrollo del sistema. Algunas funcionalidades fueron reutilizadas, pero fueron las mínimas como por ejemplo, algunas funciones que utiliza el bot de telegram.

4. Mecanismo provisto para el manejo de seguridad y routing.

Laravel provee un componente completo para el manejo de sesiones, desde el registro de un usuario hasta el inicio (autenticación) y mantenimiento seguro de la sesión durante la navegación por el sitio.
También posee seguridad sobre CSRF (Cross-Site Request Forgery) ante cada petición HTTP utilizando el método POST. Utiliza un parámetro de más, por ejemplo en un formulario, el cual es un token (VerifyCSRFToken) extenso generado al azar, y validado una vez que llega la petición al servidor. En caso de no poseer dicho token o no coincidir con el generado en el servidor, dará error.
Utiliza un ORM para el mapeo de objetos a bases de datos relacionales, llamado Eloquent, el cual a su vez utiliza PDO y parametriza las consultas para evitar SQL Injection.
Provee protección en el manejo de rutas. Estas últimas se definen en el archivo routes/web.php, utilizando la clase provista Route y definiendo, de acuerdo al tipo de petición (GET, POST, PUT, etc), una función (la cual puede ser escrita ahí mismo o una llamada a una función de un controlador especifico) que atiende la petición y los posibles middlewares. Estos middlewares son filtros HTTP que permiten o no que la petición sea llevada a cabo de manera exitosa. Se ejecutan antes de pasar la petición HTTP a la ruta correspondiente. Se pueden agrupar rutas, por ejemplo las rutas de tipo *resource*, lo cual agrupa todas las rutas necesarias para las operaciones CRUD.

5. Mecanismo provisto para operaciones CRUD.

Para el manejo de CRUD, provee los llamados controladores de recursos, los cuales, una vez generados, poseen las funciones requeridas para cada una de las operaciones (create, read, update, delete). A su vez, para cada recurso, te permite generar rutas de recursos donde cada una de ellas llama a una función específica del controlador acorde a la acción que se quiera llevar a cabo. Para operaciones donde se trabaja con el objeto Request, por ejemplo la función créate, es posible crear requests específicos con validaciones para cada uno de los parámetros que llegan en el mismo, lo cual es una ventaja para asegurar de que los parámetros que llegan al servidor, son los que el mismo acepta.

6. Forma de manejar el MVC.

Los modelos son clases donde se definen la tabla en la base de datos con la que se va a interactuar, los atributos que son mostrados ante una consulta y las funciones para el manejo de relaciones con otros modelos y para generar consultas especificas acorde a las necesidades. Se utiliza el ORM Eloquent para la persistencia de los modelos. Por defecto se encuentran en la carpeta app, pero se les puede asignar otra carpeta a gusto del programador. En mi caso, los modelos se encuentran en la carpeta app\Models

La vista se desarrolla bajo la utilización de un gestor de plantillas, llamado Blade, que viene incluido en Laravel. Utiliza una sintaxis parecida a twig pero fue necesaria la construcción de las vistas nuevamente para adaptarlas a dicho gestor. Estas se encuentran por defecto en la ruta resources\views pero también se puede crear carpetas nuevas y ubicarlas dándole cierto orden.

Los controladores poseen funciones que llevan a cabo la lógica de negocio. En general, dichas funciones son llamadas o ejecutadas por las rutas descriptas anteriormente luego de una petición. Los controladores pueden realizar consultas sobre los distintos modelos, realizar alguna acción específica y retornar una vista al cliente. Los mismos se encuentran ubicados por defecto en app\Http\Controllers.