* **Migratios:** Hecho.
* **Seeders:**Hecho
* **Controlers**: Creados todos
* *AmigosController:*

1. getAmigos: Devuelve la lista de los amigos del usuario conectado.
2. deleteAmigos: Elimina el registro.
3. getAmistad: Envia solicitud de amistad.

* *UserControler:*

1. getUserByName: Busca listado de usuarios que contengan el parámetro que le pasemos. Luego se le podrá mandar solicitud de amistad.
2. deleteUser: Borra el usuario activo. De esta manera “sales” de la red social.
3. updateUser: Actualiza los datos de tu perfil.

* *PublicacionesController:*

1. getPublicaciones: Coge las publicaciones del usuario y las fotos relacionadas con el mismo.

* *NotificacionesController:*

1. Index: coge todas las notificaciones.

HomeController: tiene dos métodos index e inicio. Ambos hacen lo mismo. Llevan a la página “inicio”.

* **Vistas:**
* *ListaUsuarios*: Muestra los usuarios que hemos buscado con getUserName
* *Perfil y Usuarios:* Ambos son un formulario que coge los datos para actualizar los datos del usuario activo. Hay que elegir uno de los dos. Al que elijamos le pasamos el JS para validar los datos antes de enviarlos (validarUpdate). HAY QUE PONER UN ENLACE EN ALGUN SITIO PARA PODER MOSTRARLO.
* **JavaScript:** Esta en public/js
* validarUpdate.js: Valida los datos que cogemos para actulizar el perfil de usuario.
* Amigos.js: Estoy trabajando en ello para recibir los datos con jquery y Ajax para mostrar los amigos del usuario activo y mostrarlo cada vez que se carge la pagina. De esta manera no haría falta buscar los amigos del usuario en cada una de las acciones de los distintos controllers y volver a mandarlos.

Habría que ver si hay que seguir esta filosofía en otros apartados.

**https://styde.net/uso-del-metodo-rendersections-con-query-ajax-y-formato-json/**