Mary, Pablo y Patricia

PROYECTO RED SOCIAL

FUNCIONES REALIZADAS

1. Utilización de Ajax:
   -Utilizamos AJAX para la solicitud de amistad:
   En templates en la plantilla PerfilOtro.html.twig en la parte de javaScript esta declarado todo.
   -La respuesta del servidor la trae de este controlador
   En controller en el controlador UsuarioController.php -> #[Route('/solicitud-amistad/{id}', name: 'solicitud_amistad')] y #[Route('/gestionar-solicitud-amistad', name: 'gestionar_solicitud_amistad', methods: ['POST'])]

   -Utilizamos AJAX para la búsqueda de usuarios:
   En templates en la plantilla Usuarios.html.twig en la parte de javaScript esta declarado todo.
   -La respuesta del servidor la trae de este controlador
   En controller en el controlador UsuarioController.php -> #[Route('/buscar-usuarios', name: 'buscar_usuarios', methods: ['POST'])]

2. Utilización funciones DOM, , incluyendo creación dinámica de nodos:
   -Utilizamos DOM para Las funciones de mostrar error, limpiar error y validar campos:
   En templates en la plantilla registrarse.html.twig y restablecer_contraseña.html.twig en la parte de javaScript esta declarado todo.

   -Utilizamos DOM para realizar el buscador de usuarios:
   En templates en la plantilla Usuarios.html.twig en la parte de javaScript esta declarado todo.
   En controller en el controlador UsuarioController.php -> #[Route('/buscar-usuarios', name: 'buscar_usuarios', methods: ['POST'])]

   -Utilizamos DOM para el contenido del post:
   En templates en la plantilla Perfil.html.twig en la parte de javaScript esta declarado todo.
   En controller en el controlador controladorGeneral.php -> #[Route('/miPerfil/{id_usuario}', name: 'miPerfil')]

Mary, Pablo y Patricia

3. Validación de campos del lado cliente con JS:
   -Utilizamos la función validar campos para poder validar todos los campos y que el usuario no pueda equivocarse:
   En templates en la plantilla registrarse.html.twig y restablecer_contraseña.html.twig en la parte de javaScript esta declarado todo.

4. Eliminar objetos con drag and drop a papelera con JS:
   -Utilizamos Drag and drop para eliminar post de usuarios con el admin:
   En templates en la plantilla admin.html.twig en la parte de javaScript esta declarado todo.

5. Uso de objetos de almacenamiento de datos del lado cliente(localStotage):
   -Utilizamos localStotage para guardar automáticamente en localStorage los borradores que dejan los usuarios al escribir en la caja de texto para subir posts, los cargará al abrir la página, y los eliminará cuando se envíe el formulario:
   En templates en la plantilla Perfil.html.twig en la parte de javaScript esta declarado todo.

6. Animación:
   -Utilizamos funciones de animación para a la hora de dar like o dislike haga una animación:
   En templates en la plantilla post.html.twig en la parte de javaScript esta declarado todo.
   En controller En el controlador ControladorGeneral.php -> (#[Route('/post/{id}/reaccionar', name: 'reaccionar_post', methods: ['POST'])])

7. Buscador:
   -Utizamos eventos, xhr, xml y dom para la búsqueda de usuarios y que sin completar el nombre ya salga:
   En templates en la plantilla Usuarios.html.twig en la parte de javaScript esta declarado todo.
   En controller en el controlador UsuarioController.php -> #[Route('/buscar-usuarios', name: 'buscar_usuarios', methods: ['POST'])]

Mary, Pablo y Patricia