

Proyecto: Sistema de gestión de blogs

Se requiere una aplicación web que nos permita gestionar escritos (blogs) de interés sobre programación por usuarios. La aplicación deberá cumplir con los siguientes requerimientos.

1. Registrar usuarios. Un interesado se debe registrar ingresando un usuario, contraseña y correo electrónico. Debe haber un proceso de activación. Por ejemplo, una vez que el usuario se haya registrado, este debe recibir un correo electrónico con un enlace para activar su cuenta. Registrar usuarios. Un interesado se debe registrar ingresando un usuario, contraseña y correo electrónico. Debe haber un proceso de activación. Por ejemplo, una vez que el usuario se haya registrado, este debe recibir un correo electrónico con un enlace para activar su cuenta.
2. Proveer un portal de acceso, donde los usuarios puedan acceder a la plataforma, si se autentican, usando usuario y contraseña, exitosamente. Esto debe cumplir con los requerimientos mínimos de seguridad.
3. Ofrecer la opción para recuperar la contraseña en caso de olvido para usuarios.
4. Ofrecer la opción para la gestión (CRUD) de blogs para los usuarios autenticados. Es decir, un usuario autenticado puede crear nuevos blogs, y actualizar y/o dar de baja de blogs existentes creados por él. En la creación y/o actualización de un blog el usuario deberá ingresar el título del blog, y su cuerpo, y si este es público o privado.
5. Un usuario autenticado puede buscar blogs usando una palabra clave para el nombre. Esta búsqueda debe mostrar todos los blogs públicos de otros usuarios y blogs (públicos o privados) del usuario autenticado que hace la búsqueda.
6. Un usuario autenticado puede comentar cualquier blog público.

B. Desglose general del proyecto para formato Scrum

1. Theme: Proyecto final Mision TIC - Uninorte

2. Initiative: Sistema de gestión de blogs

3. Epics

- 3.1. Portal de acceso (con Storie 4.1)
- 3.2. Registro de usuarios (con Storie 4.2)
- 3.3. Administración de blogs (CRUD) (con Storie 4.3)
- 3.4. Búsqueda y filtro de blogs por nombre y tema (con Storie 4.4)

4. Stories

- 4.1. **Como** Usuario, **busco** un portal de blogs **para** leer lo que escriben otras personas y dejar mi opinión acerca de lo que otros escriben.

4.2. **Como** Usuario, **quiero** registrarme en un portal de blogs **para** hacer parte de una comunidad interesada en escribir sobre lo que gusta, piensa o sueña y dejar mi opinión acerca de lo que otros escriben.

4.3. **Como** Usuario, **quiero** crear un post en mi blog, **para** editarlo, borrarlo y definir si permito o no que otros usuarios puedan leer lo que escribo.

4.4. **Como** Usuario, **quiero** buscar blogs y post públicos por nombre y tema sin tener que registrarme ó si estoy registrada **quiero** ver la información de blogs públicos o privados dentro de la comunidad.

5. Tasks (y para cada tarea hay que definir un conjunto de subtareas)

5.1. Para Historia de usuario de Daniela

1. Crear home portal de blogs
2. Acceso a login de usuario
3. Acceso a registro de nuevo usuario
4. Características para cualquier usuario
 - 4.1. Puede comentar en post públicos.
 - 4.2. Búsqueda limitada a post públicos.

5.2. Para Historia de usuario de Miguel

1. Registro con usuario y contraseña (obligatorios)
2. Validación y activación de usuario a través de correo electrónico
3. Recuperación contraseña para usuarios registrados a través de correo electrónico
4. Login en cuenta de usuario (usuario y contraseña)
5. Características para usuarios registrados
 - 4.1. Comentar post de otros usuarios.
 - 4.2. Buscar blogs y post por nombre y tema en post públicos y privados.
 - 4.3. Todo lo que incluye la historia de Jorge.

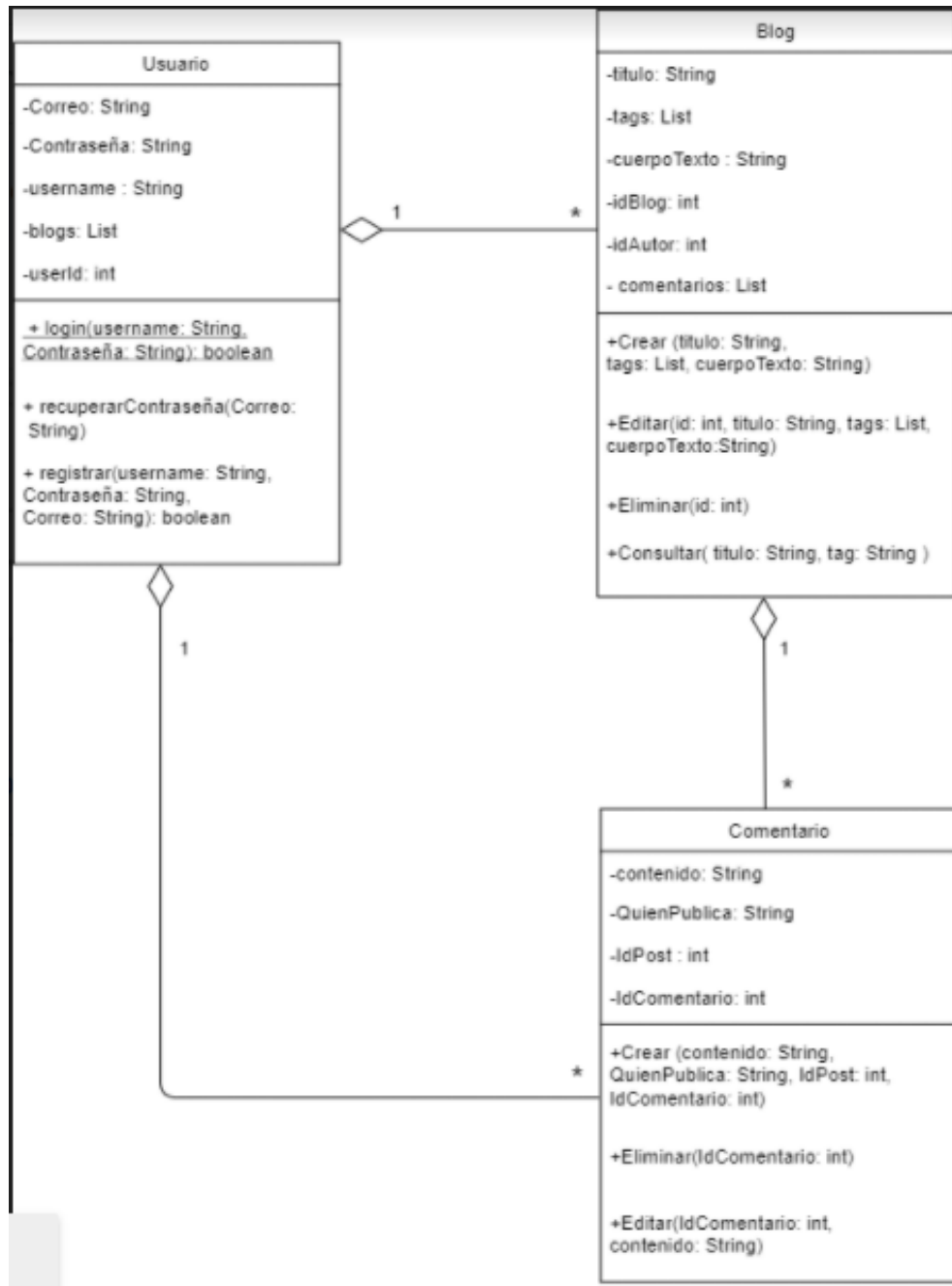
5.3. Para Historia de usuario de Jorge

1. Crear un nuevo post (Create)
 - 1.1. Agregar título
 - 1.2. Agregar cuerpo de texto
 - 1.3. Definir si es público (Todos los pueden ver) o privado (Solo los usuarios registrados y logueados lo ven) para ser indexado y mostrado en resultados de búsqueda según el estado del usuario que realiza la búsqueda (logueado o no logueado)
 - 1.4. Publicar
2. Actualizar un post (Update)
 - 2.1. Editar el título
 - 2.2. Editar el cuerpo de texto
 - 2.3. Editar si es público o privado
3. Borrar un post (Delete)
 - 3.1. Borrar un post (Pasos y cómo debe hacerlo el usuario)

5.4. Para Historia de usuario de Angela

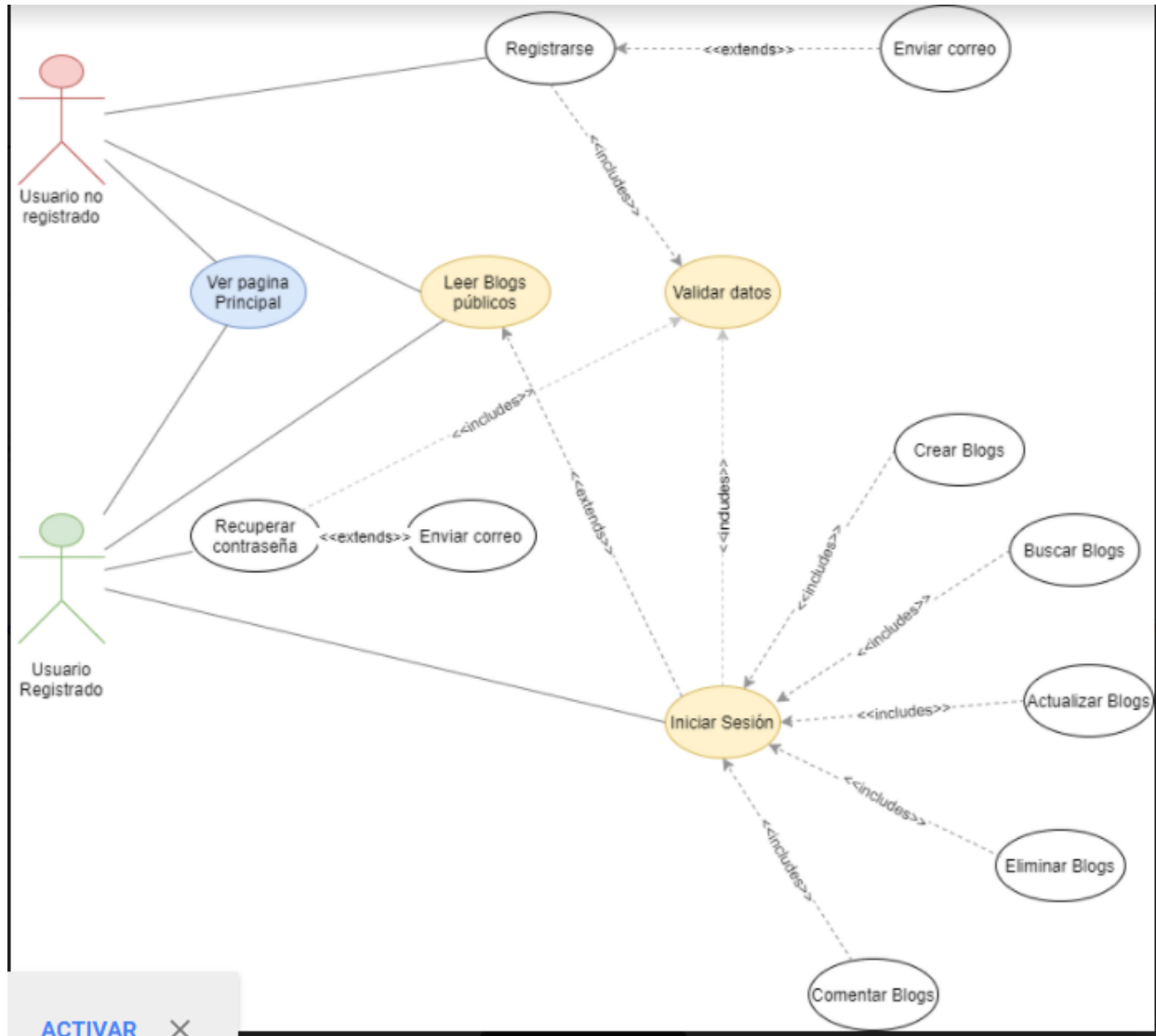
1. Campo de búsqueda por nombre o tarea
2. Despliegue de resultados para usuarios registrados
3. Despliegue de resultados para usuarios no registrados.

D. Clases



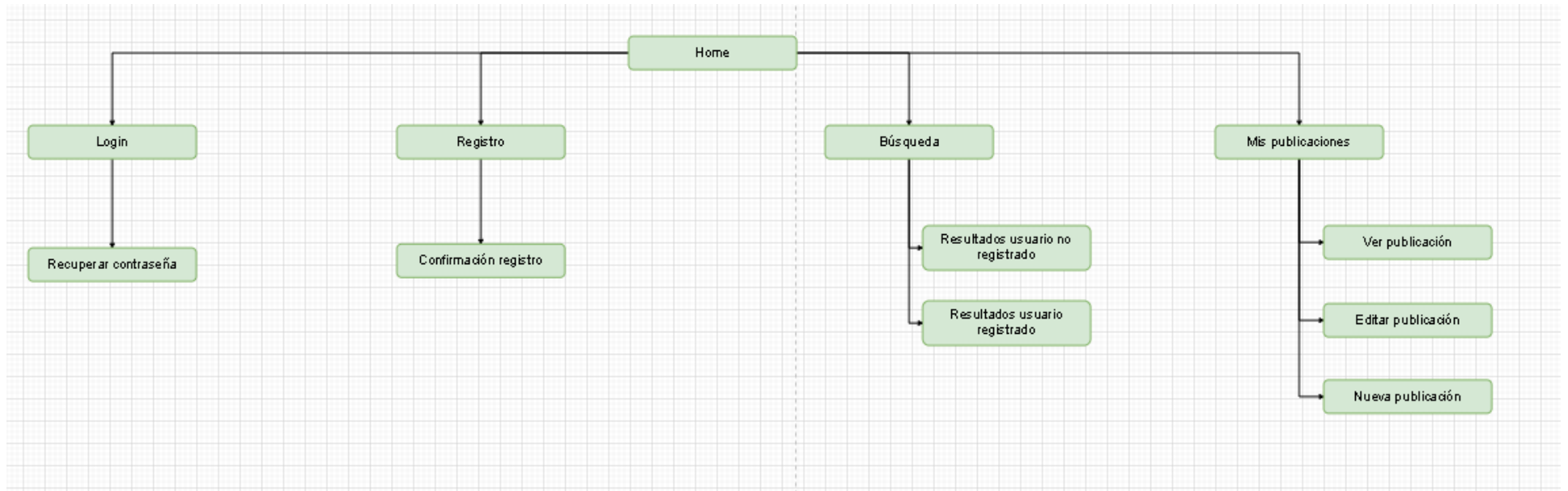
LINK: <https://app.diagrams.net/#G14p9U1wTtdOyXnJwNIS5pIrtIdILdY3I->

E. Casos de uso



<https://drive.google.com/file/d/1VjO4O9INLoBXO18y8NFnotrL92rodidX/view?usp=sharing>

Mapa de navegabilidad.










SPRINT 3 - DOCUMENTACIÓN

<i>NOMBRE DE LA FUNCIÓN</i>	<i>RUTAS ASOCIADAS</i>	<i>PARÁMETROS QUE RECIBE</i>	<i>MÉTODO DE RECEPCIÓN</i>	<i>DESCRIPCIÓN DEL OBJETIVO</i>	<i>MOCKUPS ASOCIADOS</i>
index()	‘/’	No aplica.	‘GET’ ‘POST’	Acceder a los blogs públicos para leer y/o comentar. En caso de ser usuario registrado podrá ingresar y visualizar los blogs asociados a dicho perfil.	index.html
registro()	‘/registro’	No aplica.	‘GET’ ‘POST’	El usuario podrá registrarse como usuario nuevo y luego podrá acceder al blog.	registro.html
Recuperar()	‘/recuperacion1’	No aplica.	‘GET’ ‘POST’	Primer paso para la recuperación de la contraseña, en la cual se solicita el correo.	recuperacion1.html
verificacion()	‘/verificacion’	No aplica.	‘GET’	Segundo paso de la recuperación de contraseña: Pagina para notificar al usuario la llegada correo electrónico para verificar el cambio de contraseña.	paginaVerificacionCorreo.html
Recuperar2()	‘/recuperacion2’	No aplica.	‘GET’ ‘POST’	Tercer paso para la recuperación de la contraseña (solicita la nueva contraseña luego de la verificación).	recuperacion1.html
resultados_sinsesion()	‘/resultados_sin_sesion’	No aplica.	‘GET’ ‘POST’	Resultados de la búsqueda de blogs públicos para usuarios sin registrarse o sin sesión iniciada.	resultados_sinsesion.html

resultados()	‘/resultados’	No aplica.	‘GET’ ‘POST’	Resultados de la búsqueda de blogs públicos para usuarios con sesión iniciada.	resultados.html
buscar()	‘/buscar’	No aplica.	‘GET’ ‘POST’	Página de búsqueda cuando ya se ha iniciado la sesión. El usuario podrá ingresar palabra clave.	paginaBusqueda.html
MisBlogs()	‘/MisBlogs’	No aplica.	‘GET’ ‘POST’	Mostrar los blogs publicados por un usuario logueado.	MisBlogs.html
header()	‘/header’	No aplica.	‘GET’ ‘POST’	Página para mostrar el detalle de un blog, los títulos, contenido, comentarios.	detalleBlog.html
blogs_sinSesion()	‘/blog_sinSesion’	No aplica	‘GET’ ‘POST’	Vista de blogs para usuarios no logueados.	detalleBlog_SinSesion.html
Editar()	‘Editar’	No aplica	‘GET’ ‘POST’	Página para edición de un blog particular	edicionBlog.html

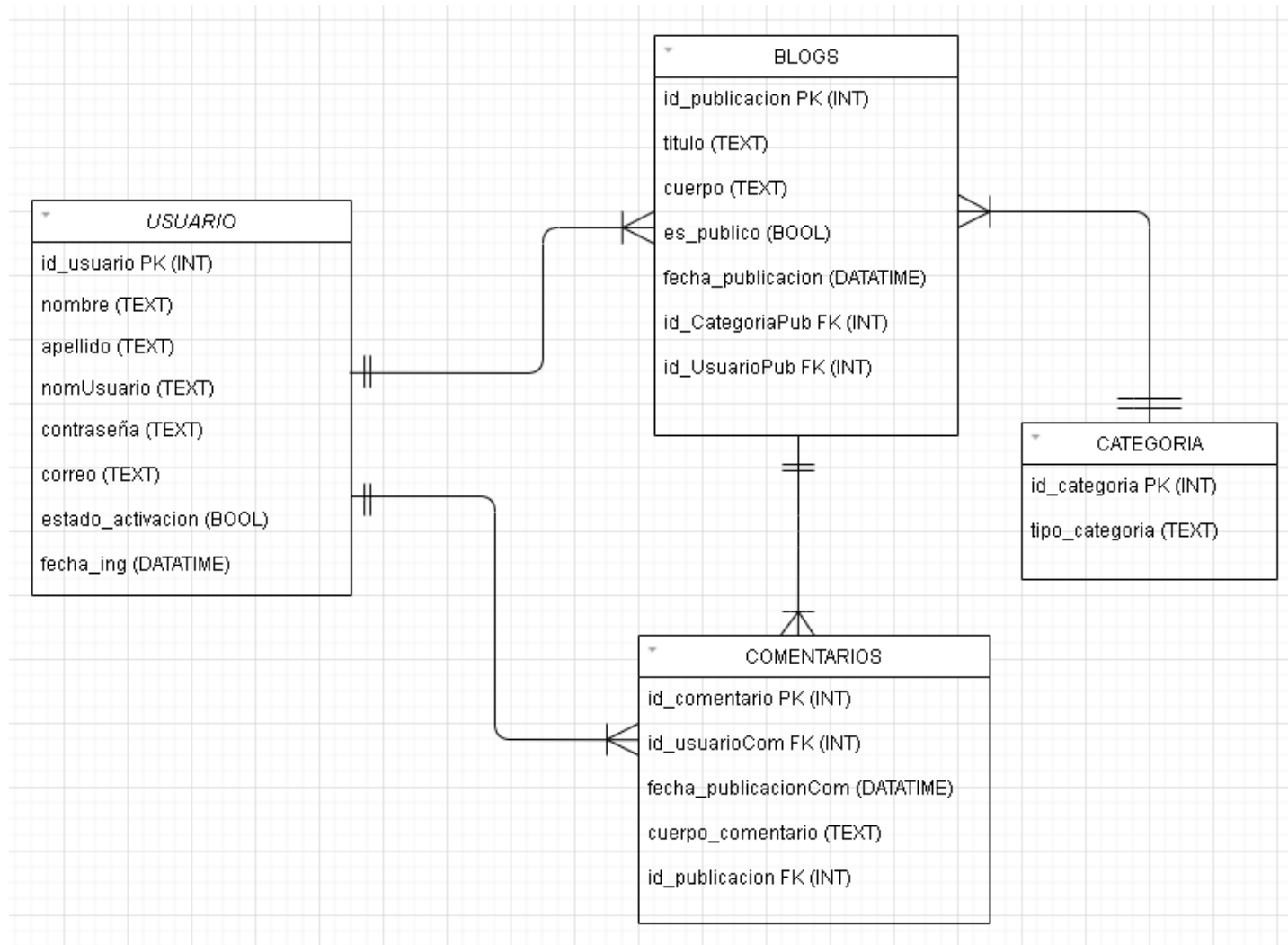
FUNCIONES GENERALES.

ValidateUser()	‘/’	<ul style="list-style-type: none"> ❖ username (String) ❖ password (String) 	POST	<p>Recibe un usuario y una contraseña y valida que coincidan con valores fijos para dar acceso. Posteriormente, esta validación se hará con la base datos, así se podría verificar la existencia del usuario y validación del password.</p>	<p>index.html</p> 
isEmailValid()	‘/registro’ ‘/Recuperar’	<ul style="list-style-type: none"> ❖ email (String) 	‘GET’	<p>Recibe un email y verifica que esté apropiadamente formateado y que realmente exista.</p>	<p>registro.html</p>  <p>recuperacion1.html</p> 
Equals()	‘/Recuperar2’ ‘/registro’	<ul style="list-style-type: none"> ❖ Nueva contraseña (String) ❖ Verificar nueva Contraseña (String) 	‘POST’	<p>Recibe una nueva contraseña junto con su verificación y se encarga de validar que estos dos campos sean iguales.</p>	<p>recuperacion2.html</p> 

isUsernameValid()	'/registro'	❖ User(String)	'POST'	Verifica que el formato de usuario sea válido. (que el usuario contenga valores alfanuméricos y algunos caracteres especiales)	<p>registro.html</p> 
isPasswordValid()	'/registro' '/Recuperar'	❖ password	'POST'	Valida que la contraseña a registrar cumpla con las características (como mínimo 8 caracteres, una mayúscula y un número)	<p>registro.html</p>  <p>recuperacion2.html</p> 
login_required()	'/resultados' '/buscar' '/MisBlogs'	❖ resultados() ❖ buscar() ❖ MisBlogs()	'GET' 'POST'	Verifica que haya usuario logueado buscando el id del user que se identifica cuando se loguea, es decir, nadie ha iniciado sesión entonces	<p>Resultados.html</p> <p>paginaBusqueda.html</p>

	'/blog' '/editar'	❖ Header() ❖ Editar()		hace un redirect a la pagina para iniciar sesión, impidiendo que el usuario sin cuenta acceda a paginas que son para usuarios registrados y logueados.	MisBlogs.html detalleBlog.html edicionBlog.html
--	--------------------------	--------------------------	--	--	---

Diagrama Entidad – Relación.



Buenas prácticas de programación.

TÉCNICA	IMPLEMENTACIÓN
Parametrizar consultas	Consultar Blogs. Crear Blogs. Editar Blogs. Eliminar Blogs.
Validar todas las entradas	Validación de expresiones regulares como el usuario, correo, contraseña.
Control de identidad y autenticación	Gestión de sesiones: Inicio de sesión, estado de la sesión iniciada, cierre de sesión. Creación de clases y funciones.
Proteger los datos	Algoritmo cifrado: Cifrado de contraseña, Hash de Contraseñas Librerías usadas: from werkzeug.security import check_password_hash, generate_password_hash
Recuperación de contraseña	Nota: en caso de tener problema con el correo http://127.0.0.1:8000/Recuperar2/UserName se debe cambiar de puerto: http://127.0.0.1:5000/Recuperar2/UserName

