

Instituto Politécnico Nacional

Unidad Profesional Interdisciplinaria   
de   
Ingeniería Y Ciencias Sociales Y Administrativas

Materia: Seguridad Informática

Alumnos: Clímaco de la Cruz María José

Espinosa Morales Uriel

González Hernández Jorge Raúl

Luis Salcedo Alejandro Daniel

Reyes Medina Francisco

Rojas Pérez Esteban Alexander

Romero Rueda Stephania Montserrath

Ciclo: 2020/1

Profesor: Sandoval Gonzalez Victor L.

Secuencia: 3NM60

Tabla de contenido

[Paquetes 3](#_Toc25627873)

[View 3](#_Toc25627874)

[Util 3](#_Toc25627875)

[Servlet 3](#_Toc25627876)

[Value Object 3](#_Toc25627877)

[Data Access Object 3](#_Toc25627878)

[Servlet 4](#_Toc25627879)

[Autenticar 4](#_Toc25627880)

[Crear Usuario 4](#_Toc25627881)

[Recuperar 5](#_Toc25627882)

[Blog 5](#_Toc25627883)

# Paquetes

## View

Se encargan de recibir información directamente del usuario a través de formularios y mostrar información según los procesos lógicos realizados por los **servlet.**

## Util

Contiene clases que servirán para realizar comprobación de los datos recibidos por los servlet, con el fin de validar que los datos que ingresa el usuario son correctos según lo que debe aceptar el sistema, se comunica con ValueObject.ErrorValueObject para mostrar mensajes al usuario.

## Servlet

Contiene clases que van a recibir peticiones de los **views,** se van a comunicar con las clases del paquete **DAO** para poder hacer altas, bajas, consultas y cambios en la **BD** y poder dar una respuesta a los views.

## Value Object

Contiene clases con objetos que van a almacenar y manipular datos del **Usuario, Blog, Product y Error.** Se comunica con los JSP para mostrar información almacenada en estos objetos, recibida de la **BD** y del paquete **Util** en caso de mostrar mensajes.

## Data Access Object

Contiene clases encargadas de comunicarse con la **BD** para realizar consultas, altas, bajas y cambios y, posteriormente mandar información a los **Value Object** para mostrar información en pantalla.

# Servlet

## Autenticar

header.jsp contiene un formulario que va a recibir como parámetros **email** y **password**, esta información es validada en el **jsp** para verificar que el correo y la contraseña tienen el formato correcto, esta información es enviada a **AutenticarServlet** y esta clase vuelve a comprobar la información a través de **SecurityUtility,** posteriormente se envía a **UsuarioDAO,** a la función

**public** UsuarioValueObject authenticate(String email, String password)

Dentro de esta función se hace una petición a través de la función

**public** String Cifrado (String urlString, String jsonString)

para obtener el password cifrado y posteriormente ser enviado junto con el email a la función

**public** String Post\_JSON(String urlString, String jsonString)

que es la encargada de realizar la comunicación con la BD a través de un servicio externo, posteriormente esta función regresa una respuesta al servlet, “true” o “false”, dependiendo de si la autenticación fue exitosa o no, en caso de que haya sido exitosa, los datos del usuario obtenidos de la BD son enviados a la clase

**public** **class** UsuarioValueObject

para crear una sesión en la aplicación y mostrar datos del usuario en los **jsp,** que son enviados desde UsuarioValueObject.

## Crear Usuario

register.jsp contiene un formulario que va a recibir toda la información necesaria del usuario, esta información es validada en el **jsp** para verificar que el correo y la contraseña tienen el formato correcto, esta información es enviada a **RegistrarUsuarioServlet** y esta clase vuelve a comprobar la información a través de **SecurityUtility,** posteriormente se envía a **UsuarioDAO** el usuario que se intenta registrar**,** a la función:

**public** UsuarioValueObject findById(String email)

Dentro de esta función se hace una petición a través de la función:

**public** String Post\_JSON(String urlString, String jsonString)

que es la encargada de realizar la comunicación con la BD para hacer una consulta, a través de un servicio externo, posteriormente esta función regresa una respuesta al servlet una respuesta dependiendo del resultado de la consulta, sirve para ver si el usuario ya se encuentra registrado. En caso de no encontrar coincidencias, el Servlet guarda los datos ingresados en el UsuarioValueObject y llama a la función

**public** UsuarioValueObject create(UsuarioValueObject user)

Para hacer el registro del usuario en la BD y posteriormente se regresa una respuesta al jsp

## Recuperar

recuperar.jsp contiene un formulario que va a recibir como parámetros **email, qiz1, qiz2 y qiz3**, esta información es validada en el **jsp** para verificar que el correo tiene el formato correcto, esta información es enviada a **RecuperarPasswordServlet** y esta clase vuelve a comprobar la información a través de **SecurityUtility,** posteriormente se envía a **UsuarioDAO,** a la función

**public** UsuarioValueObject recuperar(String email, String qiz1, String qiz2, String qiz3)

Dentro de esta función se hace una petición a través de la función:

**public** String Post\_JSON(String urlString, String jsonString)

que es la encargada de realizar la comunicación con la BD a través de un servicio externo, posteriormente esta función regresa una respuesta al servlet dependiendo de si la consulta fue exitosa o no, en caso de que haya sido exitosa, los datos del usuario obtenidos de la BD son enviados a la clase

**public** **class** UsuarioValueObject

Posteriormente se obtiene del objeto usuario el password cifrado y es enviado a la funcion

**public** String Cifrado (String urlString, String jsonString)

para obtener el password descifrado y posteriormente ser enviado al jsp para ser mostrado al usuario

## Blog

**GetBlogInfoServlet**

header.jsp contiene un form-action que llama a la clase GetBlogInfoServlet, esta clase se encarga de crear una lista en el objeto BlogEntryValueObject para almacenar todos los registros en el blog, posteriormente se llama a la función

**public** List<BlogEntryValueObject> findByCriteria(String criteria) **throws** DAOInitializationException

Dentro de esta función se hace una petición al servidor externo que se comunica con la base de datos para hacer la consulta de todos los registros existentes en la tabla blog

Todos estos elementos son almacenados en la lista creada con anterioridad a través del BlogEntryValueObject, posteriormente este objeto llama al jsp y le envía todos los datos que tiene almacenados para ser mostrados al usuario

**RegisterBlogEntryServlet**

blog.jsp contiene un form-action que llama a la clase RegisterBlogEntryServlet, esta clase se encarga de crear un objeto BlogEntryValueObject para almacenar los valores que se van a insertar a la BD, se llama a la función

**public** **void** create(BlogEntryValueObject blogEntry)

Dentro de esta función se hace una petición al servidor externo que se comunica con la base de datos para hacer inserción, utiliza para la inserción el email, obtenido de la sesión creada al momento de autenticar, y el valor ingresado en el textarea

Una vez que se obtiene una respuesta de inserción satisfactoria se da respuesta al servlet que a su vez se encarga de llamar a **GetBlogInfoServlet** para mostrar todos los registros en el blog, incluyendo el que acaba de ser creado.