# A+ Envío Correo

En este report vamos a comentar como hemos llevado a cabo el envío de correo electrónico. Para ello, hemos creado dos cuentas de correo, una que es la que envía el correo y otra que lo recibe.

Para acceder al envío del correo, simplemente debemos acceder al apartado **Contact** desde el menú principal.

Las credenciales de estas cuentas para comprobar que el envío se ha efectuado correctamente son:

Cuenta que envía el correo: aisscreamcontact@gmail.com Pass: acas41997

Cuenta que recibe el correo: aisscreamacas@gmail.com Pass: acas41997

El objetivo del A+ consiste en autenticarnos en el servidor de correo Gmail y realizar el envío de un correo cifrado. Con ello simulamos un formulario de contacto que permite a todos los usuarios contactar con el administrador del sistema.

Para realizar este A+ hemos creado una vista de edición para enviar el correo, seguido de un controlador con dos métodos, uno para mostrar el formulario y otro para realizar el envío. Por último, un servicio que nos permite autenticarnos en Gmail y enviar el correo.

Vamos a comenzar explicando el código del servicio, ya que es el más interesante:

Para ello usamos un objeto de tipo JavaMailSenderImpl con el que podamos enviar el correo. Cambiamos el valor de algunos atributos como son:

- Port: hace referencia al puerto que se va a utilizar para enviar el correo, al usar el puerto 587 estamos usando el puerto asignado a TLS para enviar el correo cifrado.
- Host: hace referencia al nombre del host de Gmail al que debemos enviar los datos para que se encargue posteriormente del envío de correo.
- Username: hace referencia al nombre de usuario de Gmail que se va a encargar de enviar el correo.
- Password: hace referencia a la contraseña correspondiente al usuario de Gmail que se va a encargar de enviar el correo.
- Una serie de propiedades:
    - mail.transport.protocol con valor smtp. Indica el protocolo que se va a usar para enviar el correo.
    - mail.smtp.auth con valor true. Indica que el correo se va a enviar autenticándose.
    - mail.smtp.starttls.enable con valor true. Indica que se quiere activar el envío de correo seguro.

Una vez hemos cambiado los valores de los atributos necesarios en el objeto. Necesitamos crear el mensaje. Para ello tenemos dos opciones.

- MimeMessage: permite crear un email construido con html y al que se le puedan añadir adjuntos. Para hacer esto podemos usar el MimeMessageHelper en el cual, al

cambiar el valor de la propiedad text, si queremos que el mensaje sea html, colocamos como segundo atributo true.

- SimpleMailMessage: permite crear un email simple.

Aunque nuestro mensaje es simple, hemos usado el primero por si en un futuro tuviéramos que usar html, sería tan sencillo como agregar un true como segundo parámetro de setText.

Por último, antes de enviar, indicamos a quién le debe llegar el correo, quién lo debe enviar, el texto del mensaje y el asunto.

Hay que destacar que el try/catch que hemos añadido en el servicio, aunque sabemos que no va a saltar nunca por cómo funcionan las transacciones, es necesario ponerlo porque el código que está contenido lo necesita, si no, aparece un error.

Con esto, ya hemos explicado el funcionamiento del servicio.

En el controlador, como hemos indicado anteriormente tenemos dos métodos, uno GET que muestra el formulario de envío de correo; y otro que se ejecuta al enviar los datos del correo.

En este último, enviamos el correo y volvemos a la página principal si todo ha ido bien, por el contrario, si se ha producido algún error mostramos el mensaje por pantalla.

También se ha creado un formulario llamado ContactForm que tiene como propiedades el asunto y el cuerpo; estas propiedades son obligatorias de forma que no te dejará enviar el correo si alguna de ellas están vacías.

Por otra parte, hemos creado una vista sencilla en la cual hemos metido un formulario con las etiquetas personalizadas correspondientes.

Por último, vamos a destacar que hemos añadido dos dependencias necesarias en el archivo pom.xml para que todo esto funcione como se espera.

```
<dependency>
    <groupId>javax.activation</groupId>
    <artifactId>activation</artifactId>
    <version>1.1</version>
</dependency>
```

Esta dependencia es usada por JavaMail para gestionar adjuntos.

```
<dependency>
    <groupId>javax.mail</groupId>
    <artifactId>mail</artifactId>
    <version>1.4</version>
</dependency>
```

Esta dependencia es la usada a la hora de enviar el email.