UNIVERSIDAD MARIANO GÁLVEZ

Geoffrey Estiven Hernández Franco

7690-14-3807

Administración de Tecnologías

Tarea

Se crea una clase de con el Crypt para Encriptar y des encriptar

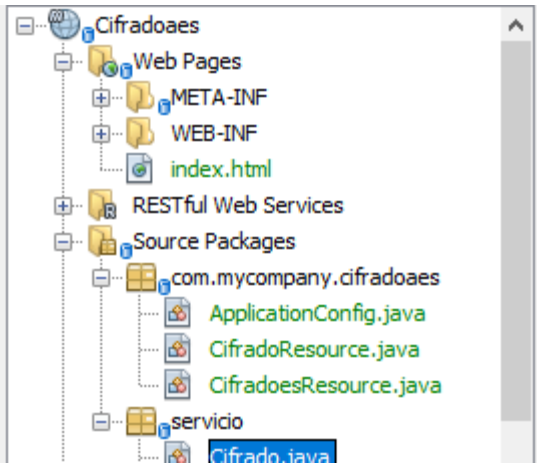
```
34  public static String encrypt(String llave, String iv, String texto) throws Exception {
35          Cipher cipher = Cipher.getInstance(tipoCifrado);
36          SecretKeySpec secretKeySpec = new SecretKeySpec(llave.getBytes(), algoritmo);
37          IvParameterSpec ivParameterSpec = new IvParameterSpec(iv.getBytes());
38          cipher.init(Cipher.ENCRYPT_MODE, secretKeySpec, ivParameterSpec);
39          byte[] encrypted = cipher.doFinal(texto.getBytes());
40          return new String(encodeBase64(encrypted));
41  }
42
43  /**
44  Función para desencriptar un String mediante algoritmo AES por bloques con los siguientes pa
45   @param llave tipo String a utilizar
46   @param iv el vector de inicialización
47   @param encrypted el texto a desencriptar previamente encriptado con la misma llave y codif
48   @return el texto descrifrado en modo String codificado en base64
49   @throws Exception excepciones que puede devolver: NoSuchAlgorithmException, NoSuchPaddingE
50  */
51  public static String decrypt(String llave, String iv, String encrypted) throws Exception {
52          Cipher cipher = Cipher.getInstance(tipoCifrado);
53          SecretKeySpec secretKeySpec = new SecretKeySpec(llave.getBytes(), algoritmo);
54          IvParameterSpec ivParameterSpec = new IvParameterSpec(iv.getBytes());
55          byte[] enc = decodeBase64(encrypted);
56          cipher.init(Cipher.DECRYPT_MODE, secretKeySpec, ivParameterSpec);
57          byte[] decrypted = cipher.doFinal(enc);
58          return new String(decrypted);
59  }
```

Se Crean una clase para serializar la data tipo json que ingrese con sus métodos de acceso.

```
import java.io.Serializable;

/**
 *
 * @author ghernandez
 */
public class Cifrado implements Serializable{
    private String clave="";
    private String vector="";
    private String texto="";
```

Se crean las clases para los servicios web



Y se crea el método para consumir los procesos que se desean

```
@POST
@Consumes(MediaType.APPLICATION_JSON)
@Produces(MediaType.APPLICATION_JSON)
@Path("/cifrado")
public Response validacion(String cifrado) {
    Cifrado data = new Cifrado();
    Gson gs = new Gson();
    Crypt crip = new Crypt();
    SalidaCifrado salida = new SalidaCifrado();
    data = gs.fromJson(cifrado, servicio.Cifrado.class);

    try {
        String salidal = crip.encrypt(data.getClave(), data.getVector(), data.getTexto());
        System.out.println("salida de Encriptado = " + salidal);
        salida.setIncriptado(salidal);
        String salida2 = crip.decrypt(data.getClave(), data.getVector(), salidal);
        salida.setDesencriptado(salida2);
        System.out.println("salida de deseEncriptado = " + salida2);
    } catch (Exception ex) {
        Logger.getLogger(NewMain.class.getName()).log(Level.SEVERE, null, ex);
        System.out.println("ERROR!!!!!!!");
    }

    return Response.ok(salida, MediaType.APPLICATION_JSON).build();

}
```

Se crea crea un json con el cual realizaremos la consulta tipo post con la url para consumir el servicio.

http://localhost:8080/Cifradoaes/webresources/Tarea/cifrado

{

"clave":"92AE31A79FEEB2A3",

"vector":"0123456789ABCDEF",

"texto":"Tarea"

}

Se consume desde Soapui

El cual nos devuelve un json con la frase cifrada y de cifrada