SEGURIDAD Y ENCRIPTACION

Integrantes :
Báez Muñoz Ana Karen
Chávez Martínez José Guadalupe
Escobar Díaz Ulises Iván
Ruiz Tique Erick Alberto

Descripción breve

- Encriptar es el proceso por el cual se cifra un texto usando una clave , esta clave es un código de signos que se interpretan según determinadas reglas para que no pueda ser entendido por nadie.

- Desencriptar sería el proceso de transformar el mensaje encriptado a texto legible usando la clave generada al encriptar el mensaje, sin esta clave el mensaje sería imposible de volverlo a transformar a texto normal.

Resumen

- Encriptar es el proceso por el cual se cifra un texto usando una clave, esta clave es un código de signos que se interpretan según determinadas reglas para que no pueda ser entendido por nadie.
- Desencriptar sería el proceso de transformar el mensaje encriptado a texto legible usando la clave generada al encriptar el mensaje, sin esta clave el mensaje sería imposible de volverlo a transformar a texto normal.

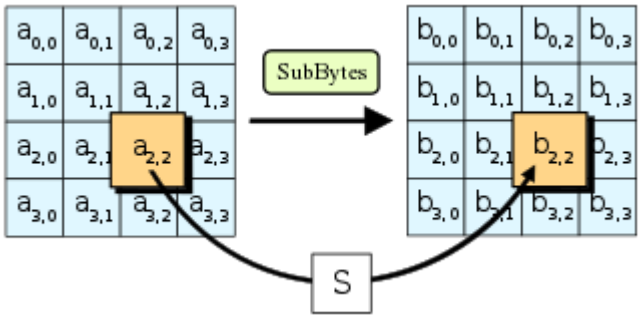
Requerimientos

Para desarrollar este proyecto usaremos unas herramientas determinadas:

Algoritmo AES

- Usaremos un algoritmo llamado AES – este algoritmo es uno de los más usados para esta función.

Advanced Encryption Standard (AES) es uno de los algoritmos de cifrado más utilizados y seguros actualmente disponibles. Es de acceso público, y es el cifrado que la NSA utiliza para asegurar documentos con la clasificación "top secret".



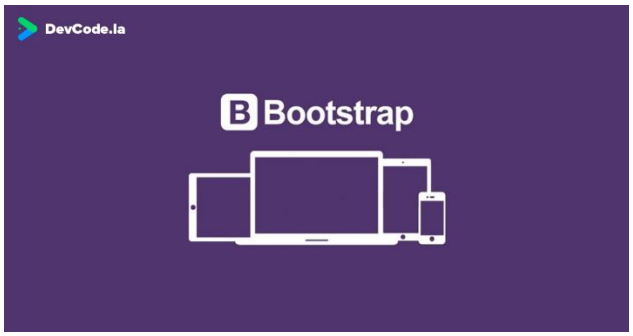
Librería CryptoJS

CryptoJS es una colección creciente de algoritmos criptográficos estándar y seguros implementados en JavaScript utilizando las mejores prácticas y patrones. Son rápidos y tienen una interfaz simple y consistente.

- Usaremos CryptoJs es una librería de algoritmos utilizados en criptografía y escritos en javascript.

Bootstrap

Es un framework de interfaz de usuario, de código abierto, creado para un desarrollo web más rápido y sencillo, su objetivo principal es crear sitios responsive. Permite que la interfaz de usuario de un sitio web funcione de manera óptima en todos los tamaños de pantalla, ya sea en teléfonos de pantalla pequeña o en dispositivos de escritorio de pantalla grande.



Problemática

Podemos considerarlo como el proceso por el cual la información legible se transforma, mediante un algoritmo, en ilegible, y se necesita de una “llave” especial para decodificarla. Este mecanismo nos permite aislar nuestra información de extraños y minimizar las consecuencias indeseadas que ellos pueden generar. El cifrado puede ser usado para proteger todo tipo de archivos; desde mails, información digital, documentos, videos, fotos, claves bancarias y personales, mensajes de texto, hasta el disco duro de una computadora.

Dicho esto, entonces, podemos pensar que cifrar o encriptar nuestra información no resultaría tan difícil, sobre todo si se encuentra en un disco duro al que solo uno tiene acceso. Sin embargo, y a pesar de lo común que se ha vuelto esto hoy en día, no existe en los usuarios individuales una verdadera conciencia de la importancia que reviste el cifrado como método de prevención y resguardo. Peor aún, hasta en el mundo corporativo, muchas empresas desestiman la importancia de asignar recursos para la implementación de verdaderas políticas de seguridad de la información.

Objetivos generales

El cifrado o la encriptación de mensajes sirve para hacer las comunicaciones más seguras, y lo mismo se puede decir a la hora de aplicarlo a Internet. La primera funcionalidad para conseguirlo es la de la confidencialidad de los mensajes, ya que, al no ir al descubierto, cuando tú le envías algo a otra persona, los algoritmos criptográficos de la aplicación ayudan a que no se pueda leer fácilmente si alguien lo intercepta en el camino.

Siendo la confidencialidad la primera de las ventajas que ofrece el cifrado de mensajes, la segunda podríamos decir que es la integridad. El encapsular un mensaje dentro de un sobre de cifrado, lo digo así para hacerse una mejor imagen mental, ayuda que todo lo que haya cifrado se mantenga correcto y completo.

También hay algoritmos criptográficos que proporcionan mecanismos para verificar la identidad de la persona que envía un mensaje. Además, hay métodos de cifrado que también ayudan a vincular un documento o transacción a una persona o sistema de gestión concretas.

Objetivo Especifico

Abarcar varios procedimientos, métodos y enfoques para proteger los datos confidenciales del acceso de terceros y realizar comunicaciones digitales seguras entre dos o más usuarios. La idea básica del cifrado es que los datos se convierten a un formato ilegible utilizando una clave antes de que se produzca un intercambio de información entre el remitente y el destinatario, o se almacenen los datos. El formato convertido se denomina texto cifrado y el formato legible se denomina texto sin formato. Sólo aquellos que conocen la clave (código) correcta para el algoritmo de cifrado tienen acceso al texto sin formato y pueden codificarlo en su forma original. Por lo tanto, el código debe mantenerse secreto o distribuirse de manera especial para que los datos puedan transmitirse o almacenarse de forma segura. El cifrado forma parte de la criptografía en lo que se refiere a la implementación técnica y a la seguridad de los diferentes métodos de cifrado.

Diseño

Los pasos a seguir son los siguientes:

-Abrimos nuestro cmd y creamos el proyecto a través del comando «ng new nombre-proyecto».

-Luego vamos instalamos las dependencias que usar nuestro proyecto:

-npm install crypto-js –save

-npm install bootstrap –save

-Iniciamos visual code atraves del cmd «Code .»

-Iniciamos el servidor para ver que la aplicación corre perfectamente «ng serve»

```
C:\Users\colosus\angular>ng new
? What name would you like to use for the new workspace and initial project? encriptar
? Would you like to add Angular routing? No
? Which stylesheet format would you like to use? CSS
```

Ilustración 1

```
C:\Users\colosus\angular\encriptar>npm install crypto-js --save
```

Ilustración 2

```
C:\Users\colosus\angular\encriptar>npm install bootstrap -save
```

Ilustración 3

```
C:\Users\colosus\angular\encriptar>ng serve
? Would you like to share anonymous usage data about this project with the Angular Team at
Google under Google's Privacy Policy at https://policies.google.com/privacy? For more
details and how to change this setting, see http://angular.io/analytics. No
0% compiling
Compiling @angular/core : es2015 as esm2015

Compiling @angular/common : es2015 as esm2015

Compiling @angular/platform-browser : es2015 as esm2015

Compiling @angular/platform-browser-dynamic : es2015 as esm2015

chunk {main} main.js, main.js.map (main) 57.7 kB [initial] [rendered]
chunk {polyfills} polyfills.js, polyfills.js.map (polyfills) 141 kB [initial] [rendered]
chunk {runtime} runtime.js, runtime.js.map (runtime) 6.15 kB [entry] [rendered]
chunk {styles} styles.js, styles.js.map (styles) 9.71 kB [initial] [rendered]
chunk {vendor} vendor.js, vendor.js.map (vendor) 2.7 MB [initial] [rendered]
Date: 2020-03-26T21:42:39.683Z - Hash: 05f36bdcc153b0d7f1cb - Time: 15478ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
: Compiled successfully.

Date: 2020-03-26T21:42:40.894Z - Hash: 05f36bdcc153b0d7f1cb
5 unchanged chunks

Time: 767ms
: Compiled successfully.
```

Ilustración 4

Copiamos el local host en el navegador y nos abrirá el proyecto, si todo va bien veríamos esto:

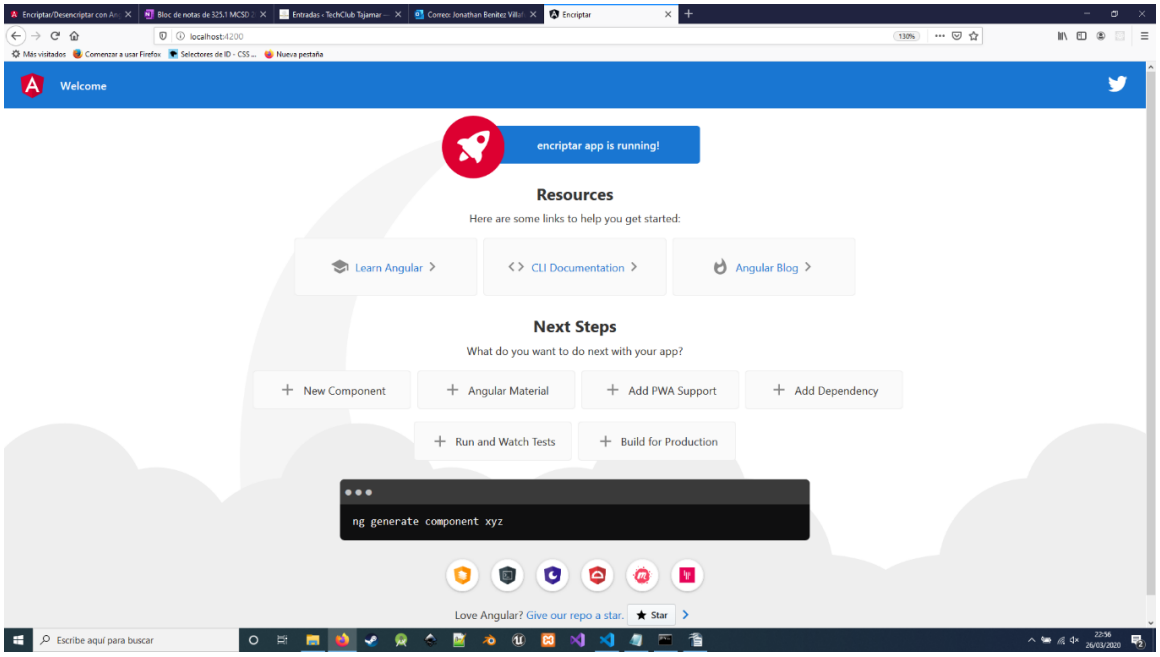


Ilustración 5

Ahora agregamos los módulos en el angular.json

```
  "styles": [
    "src/styles.css",
    "node_modules/bootstrap/dist/css/bootstrap.min.css"],


  "scripts": ["node_modules/crypto-js.js",
  "node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"
  ]
}
```

Ilustración 6

Con esto tenemos el proyecto configurado para usar los módulos agregados.

Ahora vamos a importar unas herramientas en el app.module.ts para poder usar formularios en nuestra app.

Hacemos un import de :

import {FormsModule, ReactiveFormsModule} from '@angular/forms';

y tambien lo agregamos en el imports de @NgModule:

imports: [

  BrowserModule, FormsModule, ReactiveFormsModule

],

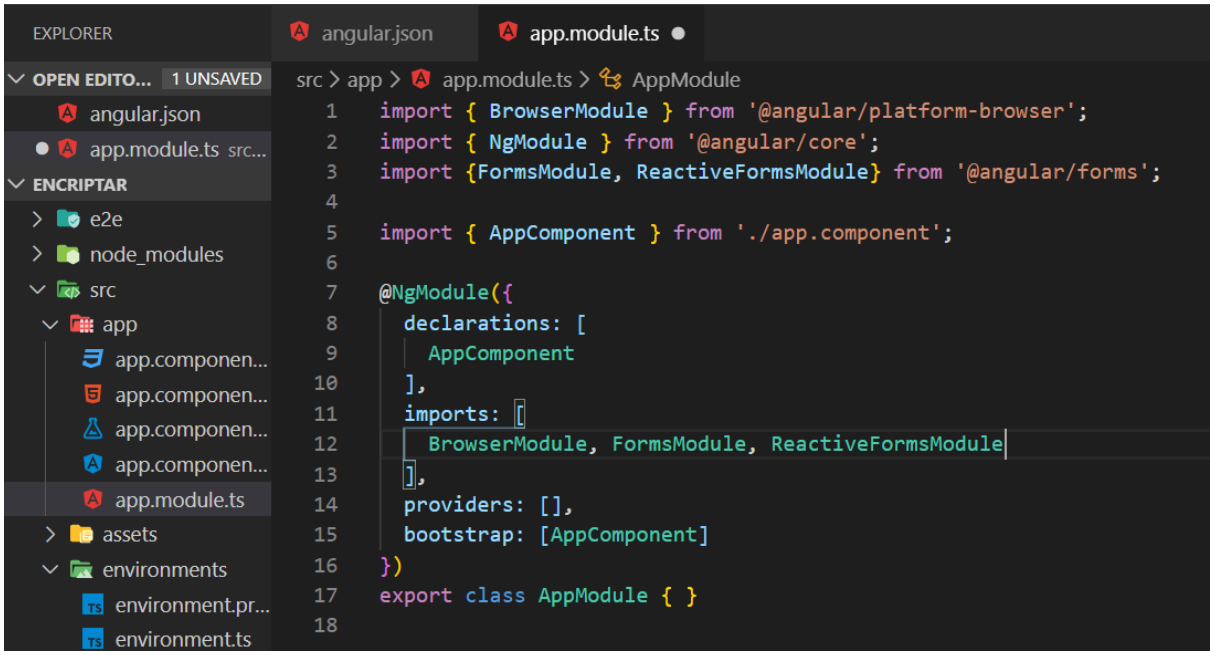


Ilustración 7

Nos creamos un componente que va contendrá la lógica y la vista de la aplicación

Ilustración 8

Una vez creado el componente vamos a la vista del componente «cifrar.component.html»

y creamos un formulario que contendrá 2 partes:

La primera parte es la de cifrado que contendrá las siguientes partes:

- Una caja para el texto que vamos a cifrar
- Una caja para meter la clave
- Un textarea para mostrar el texto ya cifrado
- Un botón para llamar a la función de convertirTexto pasándole un valor «encriptar»

La segunda parte es la de descifrado y contendrá las siguientes partes

- Una caja para meter el texto que queremos descifrar
- Una caja para meter la clave usada en el cifrado y que usaremos en esta caja para descifrar el mensaje
- Un textarea que nos mostrara el mensaje descifrado
- Un botón para llamar a la función convertirTexto pasándole un valor «desencriptar»

Ilustración 9

Ahora vamos a ver en detalle que funcionalidad tiene cada etiqueta

En esta etiqueta definimos la clase que es de tipo formulario y ngModel que capturara el contenido que hay en la caja, en este caso sería el texto que vamos a introducir para cifrar:

<input type=»text» class=»form-
control» placeholder=»Ingresa texto a encriptar» [(ngModel)]=»enctexto»>

Esta es la misma que la anterior descrita y servirá para capturar la clave que usaremos para cifrar el texto:

<input type=»password» class=»form-
control» placeholder=»Ingresa llave de encripción» [(ngModel)]=»encPass»>

Aquí usaremos un textarea para mostrar el texto cifrado, con {{textoEncriptado}} pintamos el resultado de cifrar el texto:

<textarea class=»form-
control» readonly rows=»3″>{{textoEncriptado}}</textarea>

Por ultimo tenemos el botón que contiene el evento (click)=»convertirTexto(‘encriptar’) , esto lo que hace es llamar a la función «convertirTexto» pasándole un parametro (‘encriptar’):

<button type=»button» class=»btn btn-success float-right» (click)=»convertirTexto(‘encriptar’)»>Encriptar</button>

Parte Desencriptar:

En esta etiqueta vamos a introducir el resultado de encriptar el texto y lo capturamos con ngModel

<input type=»text» class=»form-control» placeholder=»Ingrese el texto que quieres desencriptar» [(ngModel)]=»destexto»>

En esta etiqueta vamos a introducir la clave que hemos usado para cifrar y que ahora usaremos para descifrar el texto cifrado, capturamos la clave con ngModel

<input type=»password» class=»form-control» placeholder=»Ingrese la llave para desencriptar» [(ngModel)]=»desPass»>

En esta etiqueta se mostrare el texto ya descifrado y lo pintaremos con {{textoDesencriptado}}

<textarea class=»form-control» readonly rows=»3″>{{textoDesencriptado}}</textarea>

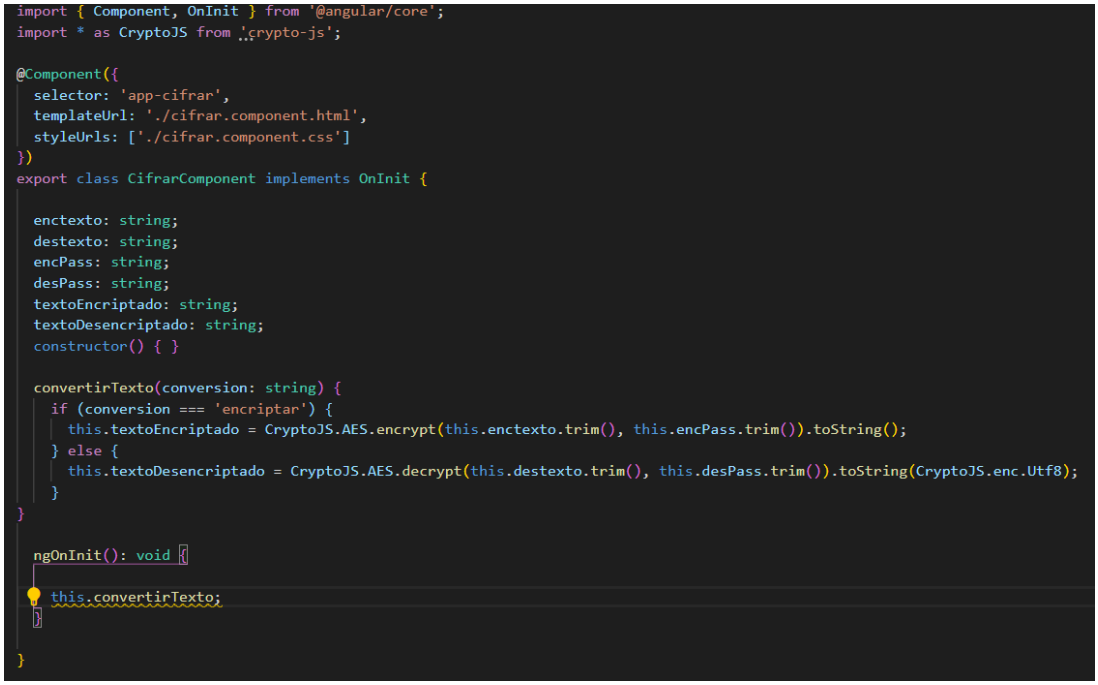
Por ultimo tenemos el botón que contiene el evento (click)=»convertirTexto(‘desencriptar’) , esto lo que hace es llamar a la función «convertirTexto» pasándole un parámetro que en este caso será el de (‘desencriptar’)

<button type=»button» class=»btn btn-success float-right» (click)=»convertirTexto(‘desencriptar’)»>Desencriptar</button>

El resultado visual de este formulario es el siguiente:

Ilustración 10

Ahora vamos a programar la parte lógica, nos iremos al component.ts donde este alojado nuestro html.



```
import { Component, OnInit } from '@angular/core';
import * as CryptoJS from 'crypto-js';

@Component({
  selector: 'app-cifrar',
  templateUrl: './cifrar.component.html',
  styleUrls: ['./cifrar.component.css']
})
export class CifrarComponent implements OnInit {

  enctexto: string;
  destexto: string;
  encPass: string;
  desPass: string;
  textoEncriptado: string;
  textoDesencriptado: string;
  constructor() { }

  convertirTexto(conversion: string) {
    if (conversion === 'encriptar') {
      this.textoEncriptado = CryptoJS.AES.encrypt(this.enctexto.trim(), this.encPass.trim()).toString();
    } else {
      this.textoDesencriptado = CryptoJS.AES.decrypt(this.destexto.trim(), this.desPass.trim()).toString(CryptoJS.enc.Utf8);
    }
}

  ngOnInit(): void {

  this.convertirTexto;
  }

}
```

Ilustración 11

Vamos a seguir los pasos:

Primero hacemos un import de CryptoJs para realizar los procesos de encriptar y desencriptar el texto

```
import * as CryptoJS from 'crypto-js';
```

Definimos el tipo de datos de los Ngmodel que hemos usado para capturar los datos introducidos en las cajas en este caso todo lo que vamos a introducir es texto asique será de tipo String

```
 enctexto: string;
  destexto: string;
  encPass: string;
  desPass: string;
  textoEncriptado: string;
  textoDesencriptado: string;
```

Lo siguiente es las funciones para realizar el proceso de encriptar y desencriptar

```
convertirTexto(conversion: string) {
    if (conversion === 'encriptar') {
      this.textoEncriptado = CryptoJS.AES.encrypt(this.enctexto.trim(), this.encPass.trim()).toString();
    } else {
```

```
    this.textoDesencriptado = CryptoJS.AES.decrypt(this.destexto.trim(), this.de
sPass.trim()).toString(CryptoJS.enc.Utf8);

  }}
```

Vamos a ver en detalle esta función

La función se llama ConvertirTexto que es llamada cuando hacemos el click en el botón, definimos el tipo de dato que le pasamos como parámetro que en este caso es string

```
convertirTexto(conversion: string) {
```

A continuación, creamos un if donde vamos a comparar el parámetro pasado en este caso comparamos que botón hemos pulsado preguntado si conversión es igual a encriptar o desencriptar

```
if (conversion === 'encriptar') {
```

Si es encriptar ejecutara esta línea de código:

Esta línea lo que hace es escoger el texto de la caja que queremos encriptar y la clave que usaremos para encriptar, mediante el método CryptoJs.AES.encrypt , usamos el módulo cryptojs definimos el algoritmo que usaremos en este caso Aes y con encrypt procedemos a encriptarlo todo.

```
    this.textoEncriptado = CryptoJS.AES.encrypt(this.enctexto.trim(), this.encPa
ss.trim()).toString();
```

Y si el botón pulsado es el de desencriptar se ejecutará la siguiente línea

La funcionalidad es la misma lo único que cambia es decrypt donde cojera el texto encriptado de la caja y la clave usada para encriptar y desencriptara el mensaje

```
} else {

    this.textoDesencriptado = CryptoJS.AES.decrypt(this.destexto.trim(), this.de
sPass.trim()).toString(CryptoJS.enc.Utf8);

  }
```

Ahora vamos a ver si es funcional:

Metemos un texto

Metemos una clave que usaremos para encriptar y también para desencriptar

¡¡Pulsamos el botón y vemos el resultado en la caja, !!bien tenemos el texto encriptado!!!

Demostración 1

Ahora vamos a Desencriptar

Metemos en la caja el texto que hemos encriptado

Metemos la clave usada en el cifrado para poder descifrarlo, tiene que ser la misma sino será imposible descifrar el texto

Demostración 2