FERIA DE VALENCIA

Proyecto de Boletines Informativos

Manual de usuario

Alumno: Carlos Gimenez Marquez
Ciclo: Grado Superior DAW 1º
Fecha: 2025

Índice

1. Información General

Boletines informativos y sistema de suscripción para la Feria de Valencia.

Objetivo:
Permitir a los usuarios recibir información personalizada sobre eventos de la Feria de Valencia mediante boletines MJML enviados automáticamente. Incluye formularios de participación y suscripción, almacenamiento de datos y control de envíos.

2. Funcionalidades para el usuario

A. Visualización del boletín (MJML)

- Los usuarios reciben boletines por correo electrónico.
- Estos boletines están diseñados con MJML y adaptados para móviles y Outlook.
- Pueden contener botones que redirigen a formularios de suscripción o sorteos.

B. Participación en sorteos (Instancia formulario-web)

- Formulario accesible mediante el boletín del evento 2 Ruedas.
- Al hacer clic en “Formulario”, se accede al formulario alojado en la instancia AWS formulario-web.
- Campos: nombre, apellido 1, apellido 2, teléfono, correo, ciudad, provincia, dirección(opcional).
- Los datos se almacenan en la base de datos para gestión de participaciones.

C. Suscripción a boletines (Instancia formulario-suscripción)

- Formulario para registrarse y seleccionar preferencias: Cómics, Motos.
- Datos almacenados: nombre, apellidos, teléfono, correo, gustos.
- Node.js lee estos datos y realiza el envío personalizado.

D. Envío de correos masivo por CSV

- También se pueden subir archivos .csv con datos de suscriptores.
- El sistema lee automáticamente los datos del CSV y realiza los envíos.

3. Diseño MJML

Para el diseño de los boletines usaremos el lenguaje MJML, en mi caso vamos a ver los pasos para empezar a diseñar con este lenguaje en la herramienta VisualStudio Code.

Primero tendríamos que instalar Node.js en nuestro ordenador.

A. Instalación Node.js

Hay dos métodos para instalar Node.js.

Instala Node.js en tu ordenador.

- Ve a https://nodejs.org y descarga la versión LTS (recomendada).
- Instálala siguiendo los pasos habituales.

- Para comprobar que está instalado, abre una terminal (CMD o PowerShell) y escribe:

```
node -v
npm -v
```

Si ves números de versión en ambas, está correcto.

Instala MJML globalmente usando npm.

- Abre la terminal (CMD, Powershell o la integrada en VS Code) y ejecuta:

```
npm install -g mjml
```

- Esto instala MJML para que puedas usarlo en cualquier proyecto.

B. Prepara tu proyecto en VS Code

- Abre VS Code.
- Crea una carpeta con tu proyecto y abrela.
- Dentro de esa carpeta, crea un archivo nuevo llamado, por ejemplo, boletin.mjml.

Una vez realizado estos pasos, podrás empezar a escribir código con lenguaje MJML.
Para tener el correo listo para enviar tendrias que convertir el MJMl a HTML mediante la terminal, usando el siguiente comando:

```
mjml boletin.mjml -o boletin.html
```

Nota:
Si quieres que la edición sea más sencilla, puedes instalar la extensión MJML.

- Instala la extensión oficial MJML.
- Tendrás resaltado de sintaxis y vista previa en vivo.
- Es muy importante que las imágenes que uses sean en linea y no local

4. Instancia AWS formulario-web y formulario-suscripción.

Para poder hacer los formularios de suscripciones y participaciones vamos a trabajar con AWS Academy que nos permite crear instancias en las cuales podemos alojar servicios.

Crearemos una instancia con el nombre que queramos, en este caso “formulario-web” y “formulario-suscripción”, seleccionaremos el sistema operativo e imágenes con las cuales vamos a trabajar. Es recomendable usar Ubuntu ya que es más permisivo y podemos trabajar más cómodos con este sistema. Seguidamente asignaremos una dirección IP elástica.

A. Instalación de paquetes

Para acceder a nuestra instancia podemos hacerlo de dos maneras.

Online

Accederemos a nuestra instancia conectándonos a ella usando AWS y usaremos la terminal online que nos proporciona la herramienta.

SSH

Para acceder a mediante SSH a la instancia es muy importante haber creado un par de claves para nuestra instancia, para tener permisos para acceder a ella, usaremos el siguiente comando:

```
ssh -i /ruta/tu-llave.pem usuario@ip-publica-de-tu-instancia
```

Apache2 (Servidor Web)

Para instalar apache en nuestra instancia actualizaremos la lista de paquetes e instalamos Apache2:

```
sudo apt update
```

```
sudo apt install apache2 -y
```

Para usarlo, deberemos iniciar el servicio apache y habilitarlo para que arranque automáticamente.

```
sudo systemctl start apache2
sudo systemctl enable apache2
```

Para verificar que apache esta funcionando correctamente lo comprobaremos mediante un comando en la terminal.

```
sudo systemctl status apache2
```

MySQL Server

Seguiremos los primeros pasos igual que en Apache2, actualizaremos la lista de paquetes e instalaremos el servicio.

```
sudo apt install mysql-server -y
```

Iniciaremos el servicio para empezar a trabajar con él.

```
sudo systemctl start mysql
sudo systemctl enable mysql
```

Un paso opcional pero recomendado es asegurar la instalación de MySQL, para mejorar la seguridad, estableciendo contraseñas en root y eliminar usuarios anónimos. etc.

```
sudo mysql_secure_installation
```

Verificamos que MySQL está funcionando.

```
sudo systemctl status mysql
```

B. Instancia formulario-suscripción

Este formulario es más extenso ya que se encarga de realizar el envío automático de los correos.

Para esta instancia instalaremos los mismos paquetes que la instancia anterior, pero añadiremos el siguiente paquete:

Node.js

Este paquete nos permite enviar correos automáticamente usando javascript, para instalarlo debemos introducir los siguientes comandos:

```
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt install -y nodejs
```

Comprobamos que el paquete está instalado.

```
node -v
npm -v
```

Para empezar a usar Node.js deberemos crear nuestro proyecto. Crearemos una carpeta e iniciaremos un proyecto Node.js.

```
mkdir mi-proyecto-node
cd mi-proyecto-node
npm init -y
```

Para enviar los correos, instalaremos librerías. El paquete más popular es nodemailer. Instálalo así:

```
npm install nodemailer
```

Una vez hayamos completado estos pasos, sigue la creación de un archivo .js (por ejemplo sendEmail.js) para enviar correos. A continuación te mostraré un ejemplo básico de un archivo .js.

```
const nodemailer = require('nodemailer');

// Configura el transportador SMTP, aquí con Gmail
const transporter = nodemailer.createTransport({
  service: 'gmail',
  auth: {
    user: 'tu-correo@gmail.com',
    pass: 'tu-contraseña-de-aplicación' // Usa contraseña de aplicación,
  }
});

const mailOptions = {
  from: 'tu-correo@gmail.com',
  to: 'destinatario@ejemplo.com',
  subject: 'Correo automático desde Node.js',
  text: 'Este es un correo enviado automáticamente.'
};

transporter.sendMail(mailOptions, function(error, info){
  if (error) {
    return console.log(error);
  }
  console.log('Correo enviado: ' + info.response);
});
```

Para comprobar que está todo bien configurado ejecuta el script con el siguiente comando:

```
node sendEmail.js
```

Nota:

- Para Gmail, crea una contraseña de aplicación.
- Asegúrate que la instancia EC2 pueda hacer conexiones salientes SMTP.
- Puedes hacer la lectura de la base de datos MySQL con algún paquete como mysql2 para obtener los correos de la base de datos.

5. Uso de la Base de Datos

Para cada instancia hay una base de datos (en el caso de formularios-suscripciones hay 2). Vamos a ver cómo configurarlas y crearlas, también a como usarlas y como se guardan los datos.

A. Configuración Base de Datos formulario-web

Para entrar en MySQL ejecutaremos el siguiente comando:

```
sudo mysql -u root -p
```

Una vez dentro ejecutamos esta lista de comandos:

```
CREATE DATABASE suscriptores_db;
```

```
USE suscriptores_db;
```

```
CREATE TABLE suscriptores (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(100) NOT NULL,
    correo VARCHAR(150) NOT NULL,
    telefono VARCHAR(20) NOT NULL,
    ciudad VARCHAR(100) NOT NULL,
    provincia VARCHAR(100) NOT NULL,
    direccion VARCHAR(255)
);
```

Creamos la base de datos suscriptores_db y la usamos para poder crear las tablas dentro. En mi caso creo la tabla de suscriptores con los datos a rellenar de mi formulario.

Teniendo la base de datos creada vamos a ver cómo guardamos los datos rellenados en el formulario en la base de datos.

```
ubuntu@ip-172-31-23-135:~$ cd /var/www/html/
ubuntu@ip-172-31-23-135:/var/www/html$ ls
gracias.html  guardar.php  index.html
ubuntu@ip-172-31-23-135:/var/www/html$
```

Dentro de este directorio encontramos el index.html el formulario a rellenar, guardar.php que se encarga de leer los datos del formulario y guardarlos en la base de datos y gracias.html que es donde te redirige una vez enviado el formulario.

```php
$servername = "localhost";
$username = "root";
$password = "0000";
$dbname = "suscriptores_db";

$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Error de conexión: " . $conn->connect_error);
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Sanear datos recibidos
    $nombre = isset($_POST["nombre"]) ? trim($_POST["nombre"]) : '';
    $apellido1 = isset($_POST["apellido1"]) ? trim($_POST["apellido1"]) : '';
    $apellido2 = isset($_POST["apellido2"]) ? trim($_POST["apellido2"]) : '';
    $telefono = isset($_POST["telefono"]) ? trim($_POST["telefono"]) : '';
    $email = isset($_POST["email"]) ? trim($_POST["email"]) : '';
    $ciudad = isset($_POST["ciudad"]) ? trim($_POST["ciudad"]) : '';
    $poblacion = isset($_POST["poblacion"]) ? trim($_POST["poblacion"]) : '';
    $direccion = isset($_POST["direccion"]) ? trim($_POST["direccion"]) : '';


    $sql = "INSERT INTO suscriptores (nombre, apellido1, apellido2, telefono>
            VALUES (?, ?, ?, ?, ?, ?, ?, ?)";
    $stmt = $conn->prepare($sql);
```

Aquí podemos ver cómo establecemos una conexión con la base de datos, indicando el usuario y contraseña y la base de datos que queremos usar, comprobamos los datos y los insertamos en la tabla dentro de la base de datos.

```
Database changed
MariaDB [suscriptores_db]> SELECT * FROM suscriptores;
+----+--------+-----------+-----------+-----------+-------------------------
+----------+-----------+------------+
| id | nombre | apellido1 | apellido2 | telefono  | email
| ciudad   | poblacion | direccion  |
+----+--------+-----------+-----------+-----------+-------------------------
+----------+-----------+------------+
|  1 | Carlos | Giménez   | Márquez   | 625147874 | carlosgimmar11@gmail.com
| Valencia | Torrent   | Calle 2pac |
+----+--------+-----------+-----------+-----------+-------------------------
+----------+-----------+------------+
1 row in set (0.000 sec)
```

Así se vería la tabla de la base de datos rellenada con algunos datos como ejemplo.

B. Configuración Base de Datos formulario-suscripción

Seguiremos los mismos pasos para acceder a MySQL en esta instancia. En esta instancia tenemos varias bases de datos, para almacenar los datos de las suscripciones, para ver a quien se le ha enviado el correo y otra para los CSV.

```
USE boletines;

CREATE TABLE historial_envios (
  id INT AUTO_INCREMENT PRIMARY KEY,
  usuario VARCHAR(255),
  nombre VARCHAR(255),
  estado VARCHAR(20),
  fecha_envio DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE DATABASE boletines_db;

USE boletines_db;

CREATE TABLE suscriptores (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(50),
    apellidos VARCHAR(100),
    correo VARCHAR(100) UNIQUE,
    gustos ENUM('Todas', 'Motos', 'Comics') NOT NULL,
    fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE DATABASE clientes_csv;

USE clientes_csv;

CREATE TABLE suscriptores_csv (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(50),
  apellidos VARCHAR(100),
  correo VARCHAR(100),
  telefono VARCHAR(20),
  gustos VARCHAR(20)
);
```

Estas son las Bases de datos que he creado en mi caso para almacenar los datos necesarios para llevar un control de la gente que se suscribe, el historial de envíos, y los clientes que añadimos mediante un CSV.

Para la base de datos de suscriptores seguimos la misma estructura que en la anterior:

```
ubuntu@ip-172-31-92-107:~$ cd /var/www/html/
ubuntu@ip-172-31-92-107:/var/www/html$ ls
gracias.html  guardar.php  index.html
ubuntu@ip-172-31-92-107:/var/www/html$
```

Para el envío de correos mediante Node.js y CSV hay que crear sus carpetas correspondientes para poder trabajar con estos servicios.

```
ubuntu@ip-172-31-92-107:~$ ls
boletines  clientes_csv
ubuntu@ip-172-31-92-107:~$
```

Dentro de la carpeta de boletines encontramos lo siguiente:

```
ubuntu@ip-172-31-92-107:~$ cd boletines/
ubuntu@ip-172-31-92-107:~/boletines$ ls
2ruedas.mjml      node_modules       package.json
SalonComic.mjml   package-lock.json  sendEmails.js
ubuntu@ip-172-31-92-107:~/boletines$
```

Los MJML que se enviaran correspondiendo a los gustos del suscriptor, paquetes que nos facilitan el envío y aseguran que todo salga como debe, y el archivo sendEmails.js que realiza el envío de los correos.

```
const fs = require('fs');
const mjml = require('mjml');
const mysql = require('mysql2/promise');
const nodemailer = require('nodemailer');

// Leer archivo MJML de comics
const htmlComics = mjml(fs.readFileSync('SalonComic.mjml', 'utf-8')).html;

// Leer archivo MJML de motos
const htmlMotos = mjml(fs.readFileSync('2ruedas.mjml', 'utf-8')).html;

// Configuración de conexión MySQL
const connectionConfig = {
  host: 'localhost',
  user: 'mailer',
  password: 'mailerpass',
  database: 'boletines'
};

// Configurar transporte de correo
const transporter = nodemailer.createTransport({
  service: 'gmail',
  auth: {
    user: 'feria.valencia.mailer@gmail.com',
    pass:
  }
});
```

Dentro del .js, realizamos la conversión de los archivos MJML a HTML ya que para que se visualice el MJML en el correo debe estar en formato HTML. Leemos los datos de la Base de datos Boletines estableciendo una conexión con esta, y a su vez configuramos el transporte de correo con el correo remitente y su contraseña de aplicación.

```
try {
  const connection = await mysql.createConnection(connectionConfig);

  const [suscriptores] = await connection.execute(
    "SELECT nombre, apellidos, correo, gustos FROM suscripciones"
  );

  if (suscriptores.length === 0) {
    console.log('❗No hay suscriptores en la base de datos.');
    return;
  }

  for (const usuario of suscriptores) {
    const gusto = normalizarTexto(usuario.gustos);

    let asunto = '';
    let html = '';

    if (gusto === 'comics') {
      asunto = '📚Boletín del Salón del Comic de Valencia';
      html = htmlComics;
    } else if (gusto === 'motos') {
      asunto = '🏍️Boletín de MotoGP y 2 Ruedas';
      html = htmlMotos;
    } else {
      console.log(`⚠️Gusto no reconocido para ${usuario.correo}: ${usuario>
      continue;
```

Este es el programa principal que envia los correos basandose en los gustos del suscriptor.

```
ubuntu@ip-172-31-92-107:~/clientes_csv$ ls
SalonComic.mjml     node_modules       package.json
importCsvEnviar.js  package-lock.json  suscriptores.csv
ubuntu@ip-172-31-92-107:~/clientes_csv$
```

Para hacerlo mediante CSV es lo mismo solo que tendremos que añadir el archivo .csv con los datos de los suscriptores.

```
nombre,apellidos,correo,telefono,gustos
Ana,Pérez,ana.perez@example.com,600123456,Cómics
Luis,García,luis.garcia@example.com,600654321,Motos
María,López,maria.lopez@example.com,600987654,Cómics
Carlos,Ruiz,carlosgimmar11@gmail.com,600456789,Cómics
Elena,Sánchez,elena.sanchez@example.com,600112233,Cómics
Javier,Morales,javier.morales@example.com,600223344,Motos
Laura,Fernández,laura.fernandez@example.com,600334455,Cómics
Pedro,Gómez,pedro.gomez@example.com,600445566,Motos
Sofía,Martínez,sofia.martinez@example.com,600556677,Cómics
Miguel,Ramírez,miguel.ramirez@example.com,600667788,Motos
```

Aqui vemos los datos que contiene el .csv, el archivo importCsvEnviar.js se encarga de importar los datos del csv a la base de datos y realizar el envío de los correos teniendo en cuenta los datos que le pasamos.

Para que el Node.js se ejecute debemos realizar el siguiente comando:

```
node importCsvAndSend.js
```

Este sería el manual para poder crear un proyecto similar paso a paso teniendo en cuenta lo realizado para que el proyecto funcione y redactando los pasos para las instalaciones de los servicios.