

MEMORIA PROYECTO

HAPPINESS&Co

Realizado por:

Ivan Jonas Fernandez Correa

Índice

Introducción	3
Programación (Java)	4
Creación Hashmap y ArrayList	4
Menú	6
Caso 1: Añadir Usuarios	7
Caso 2: Eliminar Usuario	8
Caso 3: Añadir evento	8
Caso 4: Eliminar Evento	9
Caso 5: Añadir galería.	10
Caso 6: Eliminar galería.	11
Caso 7: Añadir favorito	12
Caso 8: Eliminar favorito.	13
Base de Datos	14
Creación Base de Datos	15
Añadir Datos	16
Vistas:	17
● Vista que Devuelvan las galerías anteriores al 28-02-2025:	17
● Vista que Devuelvan los eventos favoritos del usuario 1.	17
● Vista que devuelvan las imágenes de la galería del evento del 12-01-2025(usar su id para crear la vista, no la fecha).	17
● Vista que Devuelvan los eventos favoritos del usuario 2 posteriores al 28-02-2025.	17
Lenguaje de Marcas	19
Bocetos.	19
● Boceto Página Index:	19
● Boceto Páginas de los Eventos	20
● Boceto Página Sobre Nosotros	21
● Boceto Página Formulario	21
● Boceto Página Fuentes	22
RSS	24
Conclusión	25
Bibliografía	27

Introducción

HAPPINESS Co una empresa joven dedicada al sector del ocio, busca desarrollar un portal que reúna todos los eventos de la ciudad en una agenda cultural, facilitando su difusión y conocimiento. Esta empresa encomendó la realización de un proyecto que consiste en el desarrollo de un portal web que recopile y organice eventos culturales (como música, teatro y exposiciones), junto con las bases de datos, la programación en Java y un canal RSS necesarios para su funcionamiento. Para su realización, se estableció una división en cuatro partes principales: Lenguaje de Marcas, RSS, Base de Datos y Programación. Esta memoria detalla los pasos seguidos en cada sección, desde la investigación inicial hasta la implementación, incluyendo capturas de los procesos, así como los retos encontrados y las soluciones aplicadas para cumplir con las especificaciones establecidas.

Programación (Java)

El desarrollo del proyecto HAPPINESS Co se inició con la parte de Programación, conforme a las premisas establecidas. Esta etapa fue el punto de partida por ser el área más consolidada en conocimientos, lo que facilitó su ejecución y permitió comprender mejor los conceptos necesarios para la posterior creación de bases de datos y entender mejor las relaciones entre sí de las diferentes partes del proyecto.

La parte de programación consistió en la creación de las clases necesarias (Usuarios, Eventos, Galerías y Favoritos), en cada una de las clases fueron creados sus atributos, constructores, los métodos getter y setter y el método `toString` correspondiente a cada uno. También se pidió la creación una clase principal que integra un menú interactivo mediante un bucle Do-While para trabajar con los diferentes métodos que fueron solicitados.

La creación de la clase principal, que sirve como núcleo del programa, se inició con el diseño de colecciones específicas para la gestión de datos. Se empleó un `HashMap` para almacenar los Usuarios, utilizando el email como clave única para garantizar su identificación; otro `HashMap` para los Eventos, con el id como elemento diferenciador; y un `ArrayList` para los Favoritos, estructurado para registrar de manera dinámica las preferencias de los usuarios.

Creación Hashmap y ArrayList

```
//Creamos el HashMap para usuarios
final HashMap<String, Usuarios> cuentas = new HashMap<>();
//Creamos el HashMap para eventos
final HashMap<Integer, Eventos> eventos = new HashMap<>();
//Creamos el ArrayList para favoritos
final ArrayList<Favoritos> favoritos = new ArrayList<>();
```

Una vez concluida la gestión de datos, se procedió a desarrollar la interacción con el usuario, organizada mediante un menú interactivo a través de un bucle Do-While. Este bucle realiza todas las tareas requeridas por el cliente: añadir y eliminar usuarios, eventos, galerías y favoritos, además de una opción para finalizar el programa .El código resultante es el siguiente:

Menú

```
//Creamos el menu
do {
    System.out.println("1. Añadir usuario");
    System.out.println("2. Eliminar usuario");
    System.out.println("3. Añadir evento");
    System.out.println("4. Eliminar evento");
    System.out.println("5. Añadir galería");
    System.out.println("6. Eliminar galería");
    System.out.println("7. Añadir favorito");
    System.out.println("8. Eliminar favorito");
    System.out.println("9. Salir");
    System.out.print("Seleccione una opción: ");
    opcion = scanner.nextInt();
    scanner.nextLine(); // Consumir el salto de linea

    switch (opcion) {
        case 1:
            añadirUsuario(cuentas);
            break;
        case 2:
            eliminarUsuario(cuentas);
            break;
        case 3:
            añadirEvento(eventos);
            break;
        case 4:
            eliminarEvento(eventos);
            break;
        case 5:
            añadirGaleria(eventos);
            break;
        case 6:
            eliminarGaleria(eventos);
            break;
        case 7:
            añadirFavorito(eventos, cuentas, favoritos);
            break;
        case 8:
            eliminarFavorito(favoritos);
            break;
        case 9:
            System.out.println("Saliendo...");
            break;
        default:
            System.out.println("Introduce un numero Correcto.");
    }
} while (opcion != 9);
```

Finalizada la integración del menú en el programa, se procedió al desarrollo de los métodos correspondientes a dicho menú:

Caso 1: Añadir Usuarios

```
private static void añadirUsuario(HashMap<String, Usuarios> mapa) {
    Scanner escaner = new Scanner(System.in);
    System.out.print("Indica tu Nombre: ");
    String nombre = escaner.nextLine();
    System.out.print("Indica tu Email: ");
    String correo = escaner.nextLine();
    System.out.print("Indica tu contraseña: ");
    String password = escaner.nextLine();

    if (mapa.containsKey(correo)) {
        System.out.println("El usuario ya existe.");
    } else {
        mapa.put(correo, new Usuarios(nombre, correo, password));
        System.out.println("Usuario creado correctamente.");
    }
}
```

En este método, se gestiona la creación de un nuevo usuario solicitando nombre, email y contraseña por consola. Antes de añadir el objeto al HashMap, se verifica si el email ya existe como clave, mostrando 'El usuario ya existe' en caso afirmativo o 'Usuario creado correctamente' tras su inserción.

Caso 2: Eliminar Usuario

```
private static void eliminarUsuario(HashMap<String, Usuarios> mapa) {  
  
    Scanner escaner = new Scanner(System.in);  
    System.out.print("Indica el correo del usuario a eliminar: ");  
    String correo = escaner.nextLine();  
  
    if (mapa.containsKey(correo)) {  
        mapa.remove(correo);  
        System.out.println("El usuario se ha eliminado correctamente");  
    } else {  
        System.out.println("El usuario no existe");  
    }  
}
```

Este método pide al usuario un correo por consola y lo busca en el HashMap de usuarios. Si el correo coincide con una clave existente, se elimina el usuario y se muestra 'Usuario eliminado correctamente'; de lo contrario muestra, 'El usuario no existe'

Caso 3: Añadir evento

```
static int contadorEventos = 0;  
  
// Metodos Eventos  
private static void añadirEvento(HashMap<Integer, Eventos> mapa) {  
  
    //Creamos el contador de eventos  
    Scanner escaner = new Scanner(System.in);  
  
    System.out.print("Indica la fecha del evento: ");  
    String fecha = escaner.nextLine();  
    System.out.print("Indica el titulo del Evento: ");  
    String titulo = escaner.nextLine();  
    System.out.print("Indica la ubicacion del evento: ");  
    String ubicacion = escaner.nextLine();  
    System.out.print("Escribe una breve descripcion del evento: ");  
    String descripcion = escaner.nextLine();  
    mapa.put(contadorEventos, new Eventos(contadorEventos, fecha, titulo, ubicacion, descripcion));  
    System.out.println("Evento creado correctamente.");  
    //Aumentamos el contador en cada creacion del siguiente evento  
    contadorEventos++;  
}
```

Este método , recoge por consola los datos del evento (fecha, título, ubicación y descripción), generando un id único mediante un contador estático que se incrementa por cada nuevo evento. El objeto se añade al HashMap de eventos con su colección de galerías inicializada vacía, notificando 'Evento creado correctamente' al finalizar. El contador garantiza identificadores únicos.

Caso 4: Eliminar Evento

```
private static void eliminarEvento(HashMap<Integer, Eventos> mapa) {  
    //Inicializamos el escaner  
    Scanner escaner = new Scanner(System.in);  
  
    //Mostramos toda la lista de eventos con un bucle  
    System.out.println("Listado de eventos:");  
    for (Eventos evento : mapa.values()) {  
        System.out.println(evento);  
    }  
    //Preguntamos el evento que quiere eleiminar  
    System.out.print("Ponga el ID del evento a eliminar: ");  
    int id = escaner.nextInt();  
    //Limpiamos el Buffer despues de pedir un INT  
    escaner.nextLine();  
  
    //Comprobamos si el mapa contiene el id y lo elimina  
    if (mapa.containsKey(id)) {  
        mapa.remove(id);  
        System.out.println("Evento eliminado correctamente.");  
    } else {  
        System.out.println("El evento no existe.");  
    }  
}
```

En este método, se gestiona la eliminación de un evento mostrando la lista del HashMap en un bucle. El usuario introduce el id; si existe, se elimina y se notifica 'Evento eliminado correctamente'; si no, se indica 'El evento no existe'. El bucle facilita la visualización previa de las opciones.

Caso 5: Añadir galería.

```
//Metodos Galeria
static int contadorGalerias = 0;
private static void añadirGaleria(HashMap<Integer, Eventos> mapa) {
    //Inicializamos el escaner
    Scanner escaner = new Scanner(System.in);

    //Mostramos toda la lista de eventos con un bucle
    System.out.println("Listado de eventos:");
    for (Integer evento : mapa.keySet()) {
        System.out.println("ID : " + evento + " Titulo Evento: " + mapa.get(evento).getTitulo());
    }
    //Pedimos el ID del evento que crearemos una galeria
    System.out.println("Introduce el ID del evento para crear una galeria:");
    int id = escaner.nextInt();
    //Comprobamos que el id introducido esta en el hashmap
    if (!mapa.containsKey(id)) {
        System.out.println("Introduce un ID de evento correcto");
    } else {

        escaner.nextLine(); //Para eliminar el buffer del escaner despues del INT
        if (mapa.containsKey(id)) {

            //Preguntamos el Nuevo titulo de la Galeria
            System.out.println("Introduce el Titulo para la Galeria");
            String titulo = escaner.nextLine();

            //Añadimos la Galeria con los datos introducidos y el contador creado anteriormente
            mapa.get(id).getGalerias().put(contadorGalerias, new Galerias(contadorGalerias, titulo, id));
            System.out.println("La galeria se ha creado correctamente");
            //Aumentamos el contador de galerias para la siguiente creacion
            contadorGalerias++;
        } else {
            System.out.println("El evento no existe");
        }
    }
}
```

En este método, se gestiona la creación de una galería mostrando los eventos del HashMap en un bucle. Si el id ingresado es válido, se solicitan datos, se genera un id único con un contador y se añade al ArrayList del evento, mostrando 'Galería creada correctamente'.

Caso 6: Eliminar galería.

```
private static void eliminarGaleria(HashMap<Integer, Eventos> mapa) {
    //Inicializamos el escaner
    Scanner escaner = new Scanner(System.in);

    //Mostramos toda la lista de eventos con un bucle
    System.out.println("Listado de eventos:");
    for (Integer evento : mapa.keySet()) {
        System.out.println("El evento " + mapa.get(evento).getTitulo() + "-- su ID es : " + mapa.get(evento).getId());
    }
    //Pedimos el ID del evento que queremos borrar las galerias
    System.out.println("Introduce el ID del evento para ver sus galerias:");
    int id = escaner.nextInt();

    //Hacemos un bucle para mostrar las galerias que tiene dicho evento
    if (mapa.get(id).getGalerias().isEmpty()) {
        System.out.println("El evento no tiene galerias creadas");
    } else {
        //Creamos un bucle para mostrar las galerias que tiene dicho evento
        System.out.println("El evento tiene estas galerias ");
        for (Galerias galerias : mapa.get(id).getGalerias().values()) {
            System.out.println("El titulo de esta galeria es : " + galerias.getTitulo() + "-- su ID es: " + galerias.getId());
        }
        for (Integer num:mapa.keySet()){
            System.out.println("El titulo de la galeria es: "+ mapa.get(num).getTitulo());
        }
        //Preguntamos que galeria quiere eliminar dentro de un Evento
        System.out.println("Ingrese el Id de la galeria a borrar");
        int idGaleria = escaner.nextInt();
        //Comprobamos que el evento tenga la galeria y si es asi la borramos
        if (mapa.get(id).getGalerias().containsKey(idGaleria)) {
            //Eliminamos la Galeria si esta dentro del HashMap
            mapa.get(id).getGalerias().remove(idGaleria);
            System.out.println("Galeria eliminada correctamente");
        } else {
            System.out.println("No existe galeria para ser borrada");
        }
    }
}
```

En este método, se gestiona la eliminación de una galería presentando en un bucle todos los eventos con su título e id. Se solicita al usuario el id del evento a revisar; si no tiene galerías, se muestra un mensaje de advertencia. Si las tiene, un segundo bucle lista sus galerías con id y título, pidiendo el id a eliminar. Si existe en el ArrayList, se suprime y se notifica 'Galería eliminada correctamente'; si no, se indica 'La galería no existe'.

Caso 7:Añadir favorito

```
//Metodos Favorito
private static void añadirFavorito(HashMap<Integer, Eventos> mapal, HashMap<String, Usuarios> mapa2, ArrayList<Favoritos> array) {
    //Inicializamos el escaner
    Scanner escaner = new Scanner(System.in);

    //Mostramos toda la lista de eventos con un bucle
    System.out.println("Listado de eventos:");
    for (Integer evento : mapal.keySet()) {
        System.out.println(" Titulo el Evento: " + mapal.get(evento).getTitulo() + " --- ID: " + evento);
    }
    //Mostramos todos los usuarios
    for (String usu : mapa2.keySet()) {
        System.out.println("Nombre del usuario:" + mapa2.get(usu).getNombre() + " --- Correo:" + mapa2.get(usu).getEmail());
    }

    //Pedimos al usuario el id del evento y el correo del usuario
    System.out.println("De que Evento quieres crear el favorito?(id)");
    int id = escaner.nextInt();

    escaner.nextLine();

    System.out.println("De que usuario quieres crear el favorito?(Correo)");
    String usu = escaner.nextLine();

    //Comprobamos si estan en el HashMap el correo y el Id introducido
    if (!mapa1.containsKey(id) || !mapa2.containsKey(usu)) {
        System.out.println("No se ha introducido un evento o usuario correcto");
    } else {
        array.add(new Favoritos(usu, id));
        System.out.println("El favorito se ha creado correctamente");
    }
}
```

En este método, se gestiona la adición de un favorito mostrando en bucles todos los eventos y usuarios disponibles. Se solicita al usuario el id del evento y el correo del usuario; si ambos son válidos, se crea el objeto Favorito y se añade al ArrayList, notificando 'Favorito creado correctamente'. Si algún dato es incorrecto, se muestra un mensaje de error.

Caso 8: Eliminar favorito.

```
private static void eliminarFavorito(ArrayList<Favoritos> array) {
    //Inicializamos el escaner
    Scanner escaner = new Scanner(System.in);
    //Comprobamos si el array esta vacio
    if (array.isEmpty()) {
        System.out.println("No hay favoritos registrados.");
        return;
    }
    //Mostramos toda la lista de eventos con un bucle
    System.out.println("Favoritos actuales:");
    for (Favoritos fav : array) {
        System.out.println("Usuario: " + fav.getCorreoUsuario()
            + " | Evento ID: " + fav.getIdEvento());
    }
    //Pedimos al usuario el Email a comprobar para eliminar
    System.out.print("Ingrese el email del usuario: ");
    String correo = escaner.nextLine();
    //Pedimos al usuario el ID a comprobar para eliminar
    System.out.print("Ingrese el ID del evento: ");
    int id = escaner.nextInt();
    //Limpiamos el Buffer
    escaner.nextLine();
    // Creamos un Bucle para recorrer el array y un boolean para comprobar el ID Y el Correo
    boolean encontrado = false;
    for (int i = 0; i < array.size(); i++) {

        if (array.get(i).getIdEvento() == id && array.get(i).getCorreoUsuario().equals(correo)) {
            array.remove(i);
            encontrado = true;
            break;
        }
    }
    if (encontrado) {
        System.out.println("Elemento eliminado correctamente");
    } else {
        System.out.println("No existe un elemento con esos datos");
    }
}
```

En este método, se gestiona la eliminación de un favorito verificando inicialmente si el ArrayList no está vacío mostrando el mensaje “No hay favoritos registrados ”. Si contiene elementos, se muestra la lista de favoritos con su correo de usuario e id de evento mediante un bucle. Se solicita al usuario un correo y un id; con un boolean como indicador, se recorre la colección y, si coinciden, se elimina el favorito, notificando 'Favorito eliminado correctamente'. Si los datos son erróneos, se indica 'El favorito no existe'.

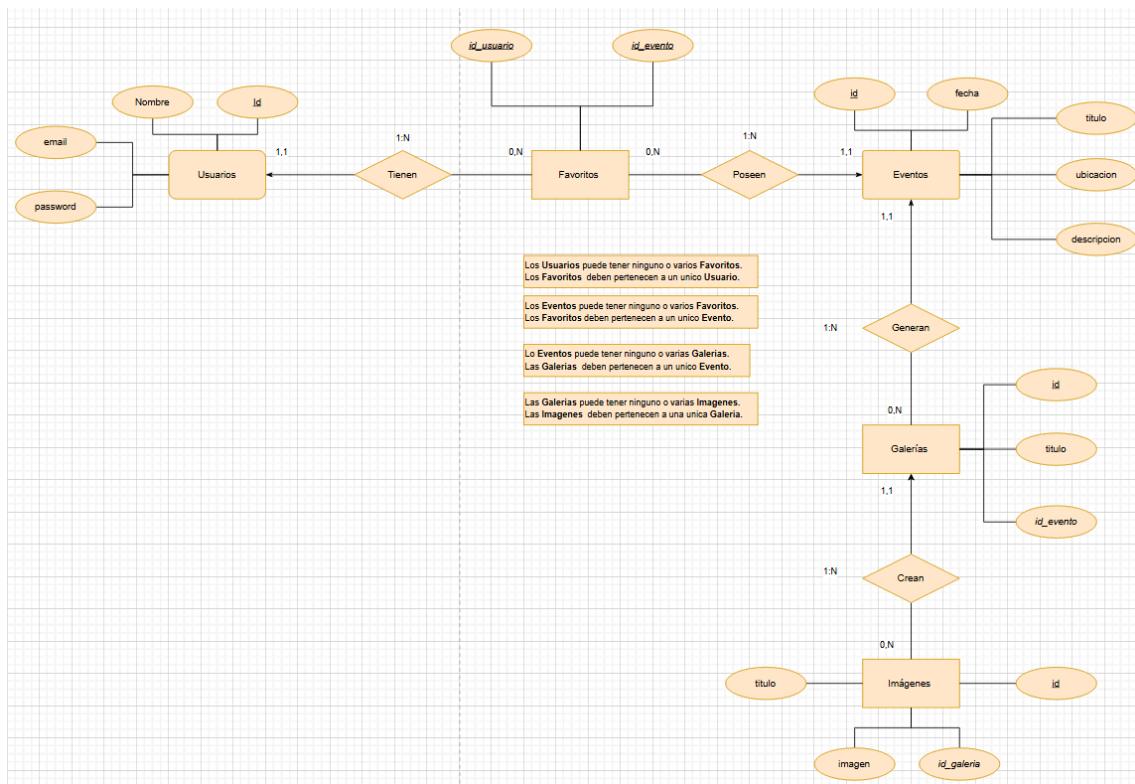
Dificultades encontradas:

Uno de los retos más importantes fue la gestión correcta del HashMap de usuarios, especialmente la verificación de correos únicos antes de insertar un nuevo usuario , para comprobar que no tenga duplicados. Otro desafío surgió en la eliminación de favoritos, requirió un esfuerzo debido a la necesidad de recorrer el ArrayList de forma eficiente. Para resolverlo, se recurrió a ejemplos de la página web StackOverflow, donde se encontraron modelos de bucles y el uso de un boolean como indicador. La integración de estas soluciones fortaleció la robustez del programa.

Base de Datos

Una vez terminada la parte de Programación, se continuó con el desarrollo de la parte de Bases de Datos, una etapa que resultó más fácil de lo esperado ya que tras consolidar la lógica de gestión de datos en Java entender la bases de datos resultó ligeramente más fácil.

Inicialmente, se elaboró un modelo entidad-relación en Draw.io. Este diseño sirvió como guía para crear una base de datos coherente y funcional en SQL Server. La experiencia previa en programación resultó clave, ya que los métodos implementados previamente sugerían las asociaciones necesarias (como crear favoritos usando el id de usuario y el id del evento) lo que agiliza el proceso y aseguró que las tablas cumplieran con las especificaciones del proyecto HAPPINESS Co.



Una vez finalizado el modelo entidad-relación y con una visión más clara de la estructura de la base de datos gracias a su modelo entidad-relación, se dio inicio a la creación de la base de datos y sus tablas, tal como se muestra a continuación.

Creación Base de Datos

```
CREATE DATABASE HappinessCo;
GO
USE HappinessCo;
GO
-- Tabla de Usuarios
CREATE TABLE Usuarios (
    id INT IDENTITY(1,1) PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL
);
GO
-- Tabla de Eventos
CREATE TABLE Eventos (
    id INT IDENTITY(1,1) PRIMARY KEY,
    fecha DATE NOT NULL,
    titulo VARCHAR(200) NOT NULL,
    ubicacion VARCHAR(200) NOT NULL,
    descripcion TEXT
);
GO
-- Tabla de Galerías
CREATE TABLE Galerias (
    id INT IDENTITY(1,1) PRIMARY KEY,
    titulo VARCHAR(200) NOT NULL,
    id_evento INT NOT NULL,
    FOREIGN KEY (id_evento) REFERENCES Eventos(id)
);
GO
-- Tabla de Imágenes de las galerías
CREATE TABLE Imagenes (
    id INT IDENTITY(1,1) PRIMARY KEY,
    titulo VARCHAR(200) NOT NULL,
    imagen VARCHAR(255) NOT NULL,
    id_galeria INT NOT NULL,
    FOREIGN KEY (id_galeria) REFERENCES Galerias(id)
);
GO
-- Tabla de Favoritos (relación muchos a muchos entre usuarios y eventos)
CREATE TABLE Favoritos (
    id_usuario INT NOT NULL,
    id_evento INT NOT NULL,
    PRIMARY KEY (id_usuario),
    FOREIGN KEY (id_usuario) REFERENCES Usuarios(id),
    FOREIGN KEY (id_evento) REFERENCES Eventos(id)
);
GO
```

Tras haber creado y verificado que todas las tablas se generaron correctamente, se procedió a incorporar los datos a cada una de ellas, asegurando el cumplimiento de las premisas establecidas. Para ello, se seleccionaron eventos que posteriormente serán utilizados en la página web, garantizando coherencia entre las distintas partes del

proyecto. Asimismo, se generaron usuarios ficticios para completar los datos requeridos y la creación de los favoritos y las galerías.

Añadir Datos

```
USE HappinessCo;
GO
--Insertamos los Usuarios
INSERT INTO Usuarios (nombre, email, password) VALUES
('Alma', 'almita@gmail.com', 'catarritos'),
('Ivan', 'ivancito@gmail.com', 'plunitos'),
('Vagabond', 'ronin@hotmail.com', 'lacaida123');

--Insertamos los Eventos
INSERT INTO Eventos (fecha, titulo, ubicacion, descripcion) VALUES
('2025-01-01', 'Feria de Muestras', 'Recinto ferial', 'Primera gran feria del año'),
('2025-01-12', 'Tsunami Xixon', 'Parking Feria de Muestras', 'Llegan grandes grupos de Gijon'),
('2025-01-24', 'Concierto de Rock en Vivo', 'Plaza ayuntamiento', 'Grupos local realizan concierto de rock'),
('2025-03-05', 'Fiesta de la Sidra', 'Playa de San Lorenzo', 'Intentar batir el record de escanciar sidra'),
('2025-03-15', 'Tarde en el Teatro', 'Teatro Jovellanos', 'Obras de grupos de teatro tanto local como nacional'),
('2025-03-25', 'Concierto NWNF', 'La laboral', 'Toca grupo NWNF en la laboral');

--Insertamos las Galerias
INSERT INTO Galerias (titulo, id_evento) VALUES
('Galería Feria', 1),
('Galería Tsunami', 2),
('Galería Concierto', 3);

--Insertamos las Imagenes de las Galerias
-- Para la galería de la Feria Muestras
INSERT INTO Imagenes (titulo, imagen, id_galeria) VALUES
('Entrada', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\feria.jpg"', 1),
('Multitud gente', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\ferial.jpg"', 1),
('Salida', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\feria2.jpg"', 1);

-- Para la galería del evento Tsunami en Gijon
INSERT INTO Imagenes (titulo, imagen, id_galeria) VALUES
('Plaza noche', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\tsu.jpg"', 2),
('Tsunami Mujer', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\tsu1.jpg"', 2),
('Llenazo', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\tsu2.jpg"', 2);

-- Para la galería del Concierto Rock
INSERT INTO Imagenes (titulo, imagen, id_galeria) VALUES
('Melendi cerca', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\concierto1.jpg"', 3),
('Escenario', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\concierto2.jpg"', 3),
('Melendi', '"C:\Users\ALUMNOS_FP\OneDrive - TuniverS Formación\Practica 1\FernandezCorreaIvanJonas\Base de Datos\Imagenes\concierto3.jpg"', 3);

--Insertamos los Favoritos (3 mínimo en cada uno , 2 del historial)
--Favoritos de Alma (Concierto , Exposición y festival cine)
INSERT INTO Favoritos (id_usuario, id_evento) VALUES
(1, 1),
(1, 2),
(1, 4);

--Favoritos de Ivan (Exposición ,Teatro y Festival de cine)
INSERT INTO Favoritos (id_usuario, id_evento) VALUES
(2, 2),
(2, 3),
(2, 4);

--Favoritos Vagabond (Concierto,Teatro y Conferencia)
INSERT INTO Favoritos (id_usuario, id_evento) VALUES
(3, 1),
(3, 3),
(3, 5);
```

Para finalizar la última etapa de la creación de la base de datos, se procedió a desarrollar las vistas requeridas por el proyecto.

Vistas:

- Vista que Devuelvan las galerías anteriores al 28-02-2025:

```
CREATE VIEW GaleriasAnteriores AS
SELECT galerias.id, galerias.titulo
FROM Galerias
JOIN Eventos ON id_evento = Eventos.id
WHERE eventos.fecha < '2025-02-28';
```

- Vista que Devuelvan los eventos favoritos del usuario 1.

```
CREATE VIEW FavoritosUsu1 AS
SELECT eventos.titulo
FROM Eventos
JOIN Favoritos ON eventos.id = Favoritos.id_evento
WHERE Favoritos.id_usuario = 1;
```

- Vista que devuelvan las imágenes de la galería del evento del 12-01-2025(usar su id para crear la vista, no la fecha).

```
CREATE VIEW Imagenes1201 AS
SELECT Imagenes.imagen
FROM Imagenes
JOIN Galerias ON Imagenes.id_galeria = Galerias.id
WHERE Galerias.id_evento = 2;
```

- Vista que Devuelvan los eventos favoritos del usuario 2 posteriores al 28-02-2025.

```
CREATE VIEW FavoritosUsu2 AS
SELECT Eventos.titulo
FROM Eventos
JOIN Favoritos ON eventos.id = Favoritos.id_evento
WHERE Favoritos.id_usuario=2 and Eventos.fecha >'28-02-2025';
```

Dificultades encontradas:

Uno de los retos al desarrollar la base de datos fue crear el modelo entidad-relación, que se necesitaba para que las vistas cumplieran con lo que se pedía. Como solo se

dieron las pautas de las tablas, hacer el modelo se complicó un poco, ya que había que deducir cómo se relacionaban las entidades pero comenzar con la parte de Programación utilizo bastante esta parte del proyecto. Una vez que estuvo listo, el modelo ayudó bastante a las siguientes partes del proyecto , como crear las tablas y las vistas. Tener las relaciones bien definidas fue clave, porque si algo estaba mal, las vistas no iban a mostrar los datos como se esperaba.

Lenguaje de Marcas

Por último, se llevó a cabo la parte de Lenguaje de Marcas, la cual resultó ser la más extensa y complicada del proyecto , ya que se partía con menos experiencia en la parte de desarrollo web, especialmente en lo relacionado con estilos y maquetación de páginas. Para abordar esta etapa, se elaboraron unos bocetos iniciales que reflejaban una idea general de cómo debía verse la página, para ello se usó la página Balsamiq. Además, se buscaron ideas en otras páginas web que ayudaron a mejorar el resultado final. A continuación, se presentan bocetos creados:

Bocetos.

- Boceto Página Index:



HAPPINESS&Co

- Boceto Páginas de los Eventos

A Web Page
https://

HAPPINESS&Co

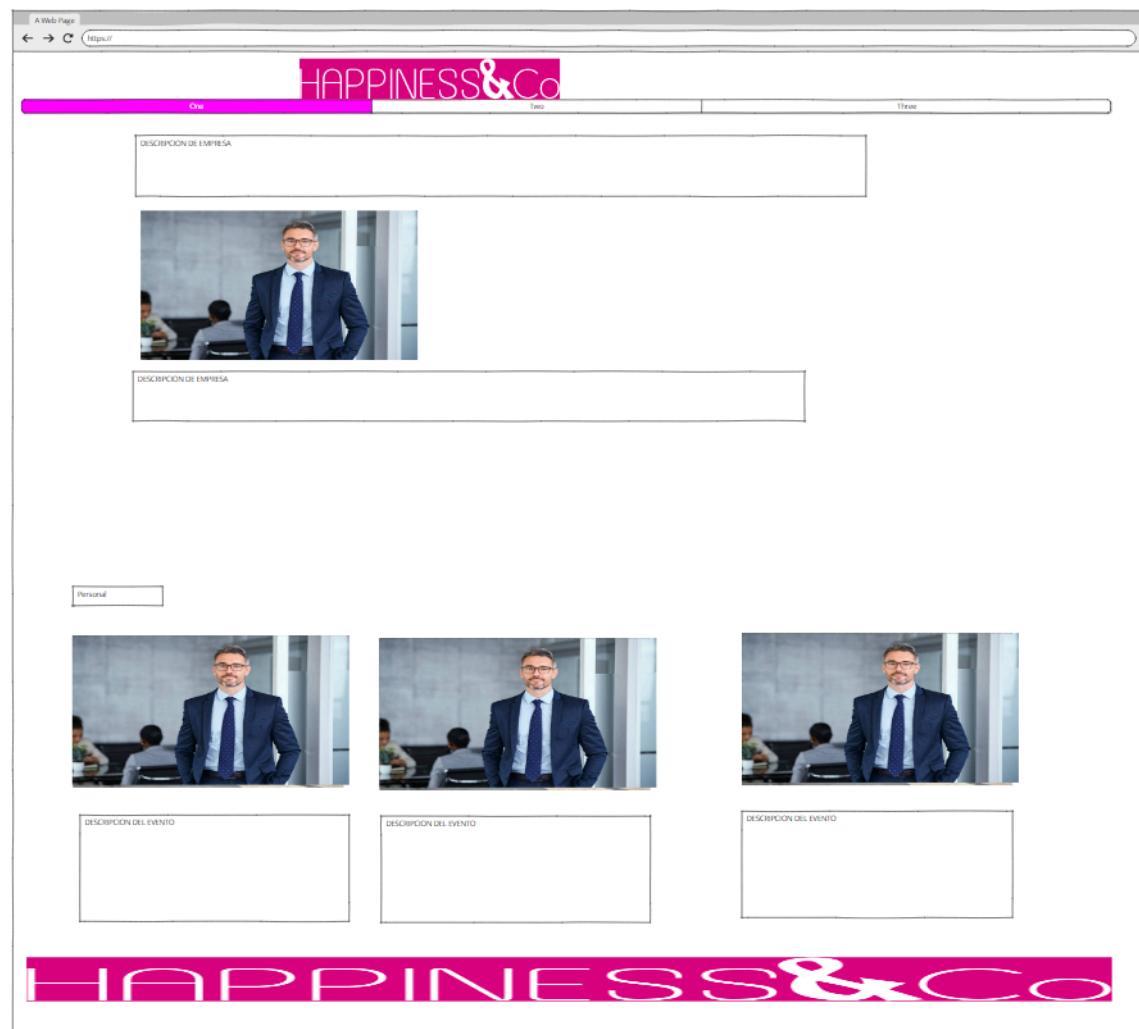
One Two Three

DESCRIPCION DEL EVENTO

GALERIA

HAPPINESS&Co

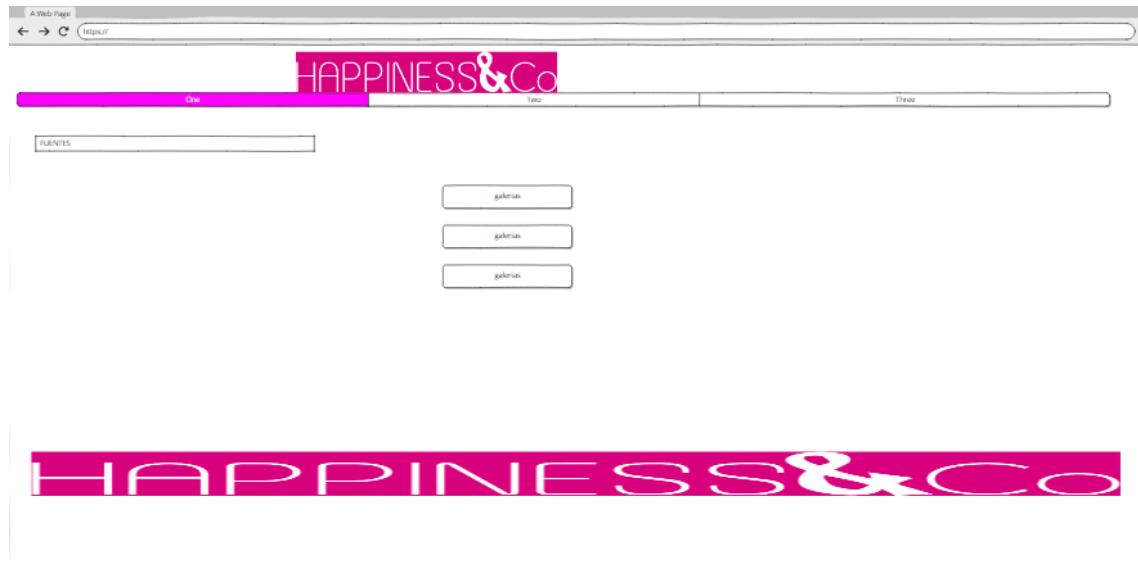
- Boceto Página Sobre Nosotros



- Boceto Página Formulario



● Boceto Página Fuentes



Una vez definida la idea principal de cómo debía verse la página, considerando que los bocetos solo ofrecían una visión general y gracias a las recomendaciones del tutor de prácticas, Johnny, sobre cómo organizar las carpetas y estructurar el proyecto, se procedió a la creación y organización en carpetas de todo el contenido del proyecto y de la página, así como de las relaciones entre sus elementos. A continuación, se muestra la distribución del proyecto: la carpeta 'assets' incluye las imágenes, el código CSS y los scripts utilizados por la página, mientras que la carpeta 'eventos' aloja las páginas de cada evento, facilitando la posible incorporación de nuevos eventos en el futuro.

📁 assets	✓	08/04/2025 12:11	Carpeta de archivos
📁 eventos	✓	08/04/2025 12:11	Carpeta de archivos
📝 contacto.html	✓	08/04/2025 9:13	Brave HTML Docu... 3 KB
📝 eventos.html	✓	08/04/2025 12:12	Brave HTML Docu... 5 KB
📝 fuentes.html	✓	08/04/2025 9:22	Brave HTML Docu... 2 KB
📝 historial.html	✓	02/04/2025 12:48	Brave HTML Docu... 6 KB
📝 index.html	✓	08/04/2025 12:06	Brave HTML Docu... 5 KB
📝 nosotros.html	✓	08/04/2025 12:31	Brave HTML Docu... 5 KB

Una vez finalizada la creación de las páginas web y su rellenado de datos y relaciones entre las páginas, se continuó con el diseño de la página, explorando ejemplos e ideas de CSS en diversas páginas para obtener inspiración e implementarlas en el proyecto lo que mejoró enormemente el diseño del mismo. Este proceso resultó ser el más extenso, ya que implicó una investigación exhaustiva del diseño, que definió la apariencia final de la página web. Al analizar ejemplos de CSS, el boceto inicial y la idea final de la página variaron en algunos aspectos, debido a la integración de código CSS proveniente de sitios como 'Codepen' y plantillas de 'Free-CSS', de donde se tomaron algunos elementos para incorporarlos y dar forma al diseño final del proyecto. Una vez aplicadas las ideas investigadas y las plantillas, la página principal quedó configurada como se muestra a continuación (<https://jonasitohappinesco.neocities.org/>).



The screenshot shows the homepage of the Happiness&Co website. At the top, there is a navigation bar with links for INICIO, EVENTOS, HISTORIAL DE EVENTOS, SOBRE NOSOTROS, FUENTES, and CONTRATO. Below the navigation bar is a large banner image of a live rock concert with a dense crowd. The banner has a caption that reads "Concierto de Rock en Vivo" and "Fecha: 24 de enero de 2025". Underneath the banner, there is a section titled "Ro te_ Gijon-]" which contains three event cards. The first card is for "Fiesta de la Sidra" on March 5th, featuring two small images of people at a festival. The second card is for "Torde en el Teatro" on March 15th, showing two images of theater performances. The third card is for "Concierto NWTF" on March 25th, showing two images of a band performing on stage. Below these cards is a section titled "Informate+" which features two more event cards. The first is for "Feria de Muestras" on January 1st, showing images of a busy fairground. The second is for "Tsunemi Xivon" on January 12th, showing images of a night concert. At the bottom of the page, there is a footer with the text "© 2025 Happiness&Co" and social media icons for Facebook, Twitter, and YouTube.

RSS

Una vez terminado todo lo relacionado con la página web, incluyendo las relaciones y el contenido de todas las páginas, la organización de los archivos del proyecto, y tras verificar que las implementaciones de CSS y JavaScript funcionaban correctamente, se dio por concluida la sección de Lenguaje de Marcas con la creación de las RSS de la página. Mostrada a continuación:

```
<?xml version="1.0" encoding="UTF-8"?>
|<rss version="2.0">
|<channel>
|<title>HappinessCo - Próximos Eventos en Gijón</title>
|<link>http://jonasitohappinesco.ct.ws/</link>
|<description>Descubre los próximos eventos culturales y de ocio en Gijón con HappinessCo.</description>
|<language>es-es</language>
|
|<item>
|<title>Feria de Muestras</title>
|<link>http://jonasitohappinesco.ct.ws/eventos/eventoFeria.html</link>
|<description>El mayor escaparate comercial y cultural de Asturias, donde empresas, innovación y tradición se unen.</description>
|<pubDate>Wed, 01 Jan 2025 15:12:54 GMT</pubDate>
|</item>
|</channel>
|</rss>
```

En la imagen se muestra la estructura utilizada para la creación de las RSS, incluyendo un ejemplo de un evento extraído de la página, con su título, fecha, descripción y enlace correspondiente. Esta estructura se sugirió después de buscar información de la correcta estructura de las RSS.

Dificultades encontradas:

Durante la creación de la página, se investigaron ideas y plantillas de JavaScript para incorporarlas al proyecto. Este proceso resultó ser uno de los mayores desafíos al diseñar la página, ya que la correcta integración de estos elementos de JavaScript tomó más tiempo del esperado. Entre las integraciones de JavaScript implementadas en el proyecto, se incluyó un script que muestra texto 'dinámico' cambiante sobre eventos próximos y pasados actualizándose cada pocos segundos para captar la atención y hacer la página más interactiva. Además, se añadió un botón que, al hacer clic, muestra eventos recomendados en las páginas de eventos pasados con galerías para hacerlas más visuales, como se muestra a continuación.

The screenshot shows the main landing page of the Happiness&Co website. At the top, there's a large pink header with the brand name 'HAPPINESS&Co' in white. Below it is a navigation bar with links: INICIO, EVENTOS, HISTORIAL DE EVENTOS, SOBRE NOSOTROS, FUENTES, and CONTACTO. A sub-navigation menu for 'Eventos Recomendados' is visible under the 'EVENTOS' link. The main content area features a large image of a crowded outdoor event with red umbrellas and a banner for 'volare THE WHITE FESTIVAL'. A pink button labeled 'Eventos Recomendados' is overlaid on the image. The overall design is clean with a pink color scheme.

This screenshot shows a section titled 'Eventos Recomendados' on the Happiness&Co website. It displays two event cards. The first card is for 'Concierto Rock' on January 24, 2025, featuring a photo of a large crowd at a concert. The second card is for 'Tsunami Xixon' on January 12, 2025, featuring a photo of a night concert. Both cards include a brief description of the event and its location. The footer of this section includes the text '© 2025 Happiness&Co' and social media icons for Facebook, Twitter, and Instagram.

Conclusión

Una vez terminado el proyecto para HAPPINESS CO. y haciendo un análisis del mismo, puedo darme cuenta de cómo funcionan los proyectos de creación de páginas web en todo el proceso: desde la parte de programación lógica para el funcionamiento de la web, además de la correcta creación de las bases de datos para guardar y manejar los futuros datos obtenidos de la página, y, por último, de la parte más visual y directa con los usuarios, de la creación de la página web, ya no solo a nivel de contenido, sino también de la aplicación de estilos y scripts para adornar y hacer más llamativa la página.

Tras estas semanas en la realización del proyecto, me he dado cuenta de lo entretenido y frustrante que puede llegar a ser realizar un proyecto en su totalidad cuando tienes cierta libertad de decisiones, pero tienes que mantener una correcta relación entre las partes. Porque, a medida que iba avanzando el proyecto y realizando las diferentes partes, se me iban ocurriendo diferentes ideas o no todos los días trabajaba de la misma manera, y tener que ir cuadrando todo para realizar un proyecto final, completo y agradable visualmente, que cumpla todo lo que un cliente te puede pedir, es más complicado de lo que en un principio puede parecer.

Pero, una vez colocadas todas las piezas y viendo el proyecto en su totalidad, estoy más que contento con el trabajo realizado, ya que poder aplicar los conocimientos que estamos adquiriendo durante el ciclo para, en cierta manera, realizar un proyecto completo de creación de una página web es una sensación satisfactoria y no me deja más que ganas de ver qué siguientes proyectos acabaré desarrollando.

Un saludo,Iván.

Bibliografía

1. [Codepen](#)
2. [StackOverflow](#)
3. [Balsamiq](#)
4. <https://blog.hubspot.es/website/maquetacion-web>
5. https://developer.mozilla.org/es/docs/Learn_web_development/Getting_started/Your_first_website/Adding_interactivity