Programación Avanzada en Web

Gerald Blanco Navarro, Bryan Cantillo Aguilar, David Fernández, Anthony Ruiz González

Departamento de Ingeniería,
Universidad Fidélitas, San José,
Costa Rica

gblanco30269@ufide.ac.cr
bcantillo60745@ufide.ac.cr
dfernandez10362@ufide.ac.cr
aruiz00724@ufide.ac.cr

***Abstract*— Este documento IEEE describe el diseño y desarrollo de un sistema educativo en línea, que proporciona una amplia gama de cursos en diferentes áreas de conocimiento. El sistema se basa en una arquitectura web que integra una aplicación web (Web Application) y una API web (Web API), implementando el patrón de diseño Modelo-Vista-Controlador (MVC) para una organización eficiente del código y una experiencia de usuario fluida.**

I. Introducción

El presente documento describe el diseño y desarrollo de un sistema educativo en línea, basado en una arquitectura web sólida y eficiente, que proporciona una experiencia educativa enriquecedora y accesible para usuarios de diferentes perfiles y niveles de conocimiento. El sistema se concibe como una herramienta versátil y dinámica, inspirada en plataformas exitosas como Udemy, que ofrece una amplia gama de cursos. Para garantizar un funcionamiento óptimo y una experiencia de usuario fluida, se aplican técnicas de desarrollo modernas, principios de diseño responsivo y estándares de calidad de software.

II. OBJETIVOS

A. Objetivo general

Diseñar, desarrollar e implementar un sistema educativo en línea que proporcione una plataforma accesible y eficiente para el aprendizaje en diferentes áreas de conocimiento.

B. Objetivos específicos

Diseño de una arquitectura web robusta: Definir una arquitectura web sólida y escalable que permita el desarrollo de una aplicación web y una API web.

Desarrollo de funcionalidades clave: Implementar funcionalidades esenciales, como el registro de usuarios, el inicio de sesión, la gestión de cursos y la administración de perfiles de usuario.

Garantizar la seguridad y confiabilidad del sistema: Aplicar prácticas de seguridad informática y buenas prácticas de desarrollo de software para proteger los datos sensibles de los usuarios.

Optimización de la experiencia de usuario: Diseñar una interfaz de usuario intuitiva y atractiva utilizando tecnologías como Bootstrap y Razor.

Cumplimiento de estándares y regulaciones: Asegurar el cumplimiento de estándares y regulaciones relevantes en materia de privacidad de datos, accesibilidad web y usabilidad.

III. IMPORTANCIA

El desarrollo de un sistema educativo en línea es de suma importancia en el contexto actual, ya que brinda acceso universal a la educación de alta calidad, promoviendo la flexibilidad y conveniencia en el aprendizaje. Al ofrecer una amplia variedad de cursos en diversas áreas de conocimiento, el sistema facilita la actualización de habilidades y competencias para profesionales de todas las industrias, fomentando la innovación en los métodos de enseñanza y contribuyendo a la inclusión y diversidad en la educación. Además, al reducir los costos asociados con la educación presencial, el sistema ayuda a democratizar el acceso a la educación y a impulsar el crecimiento personal y profesional de los usuarios en el entorno digital actual.

IV. NECESIDADES QUE SOLVENTA

El proyecto propuesto es una simulación conceptual de un sistema educativo en línea. Aunque el proyecto es ficticio, se basa en los principios y estándares de desarrollo de sistemas educativos reales y responde a las necesidades crecientes de aprendizaje en un mundo digitalizado. Este sistema busca solventar una serie de necesidades importantes en el ámbito educativo actual, tales como el acceso universal a la educación de calidad, la diversidad de opciones de cursos y la actualización constante de habilidades y competencias. Además, busca abordar la creciente demanda de educación a distancia, promoviendo la inclusión y la igualdad de oportunidades educativas para personas de diversas ubicaciones geográficas, orígenes socioeconómicos y niveles de experiencia. En resumen, este proyecto tiene como objetivo principal ofrecer una plataforma educativa versátil y accesible que satisfaga las necesidades cambiantes de aprendizaje en la era digital.

A. *Requerimientos funcionales*

Registro de usuarios: Los usuarios deben poder registrarse en el sistema proporcionando información básica como nombre, correo electrónico y contraseña.

Inicio de sesión: Los usuarios registrados deben poder iniciar sesión en el sistema utilizando sus credenciales.

Gestión de perfiles de usuario: Los usuarios deben poder gestionar su perfil de usuario, incluyendo la actualización de la información personal y la configuración de preferencias.

Búsqueda y navegación de cursos: Los usuarios deben poder buscar cursos al navegar por una lista de cursos disponibles.

Visualización de detalles del curso: Los usuarios deben poder ver detalles específicos de un curso, incluyendo su descripción, instructor, duración, precio, entre otros.

Inscripción en cursos: Los usuarios deben poder inscribirse en cursos seleccionados y realizar el pago correspondiente, si es necesario.

B. *Hiper-Vínculos y Accesos Directos*

Cualquier hiper-vínculo o referencia a Internet debe escribirse por completo. Es decir, escribir el URL complete de la ubicación del recurso en lugar de dejar accesos directos.

Las referencias se escriben usando fuente regular igual que el resto del artículo.

VI. MARCO TEÓRICO

Para entender un poco mejor el proyecto como tal, es importante comprender también qué son los términos de los que se habla en este documento. Para ello, se han recopilado los siguientes términos y definiciones que podrían ser útiles para el lector.

API: El término API se abrevia de Application Programming Interfaces, que en español significa interfaz de programación de aplicaciones. Se trata de un conjunto de reglas y definiciones que se utilizan para desarrollar e integrar software de aplicaciones, permitiendo la comunicación entre dos aplicaciones de software. Por lo tanto, una API es una especificación oficial que especifica cómo un módulo de software se comunica o interactúa con otro para realizar una o más funciones. Todo depende de las aplicaciones que usen y de los permisos que el propietario de la API les otorgue a los desarrolladores de terceros. [1]

Aplicación Web: Este tipo de aplicación es la que se ejecuta y es creada específicamente para entornos web. La aplicación web se basa en HTML, JavaScript, CSS, entre otros, para poder desplegar el sistema que necesite el programador.

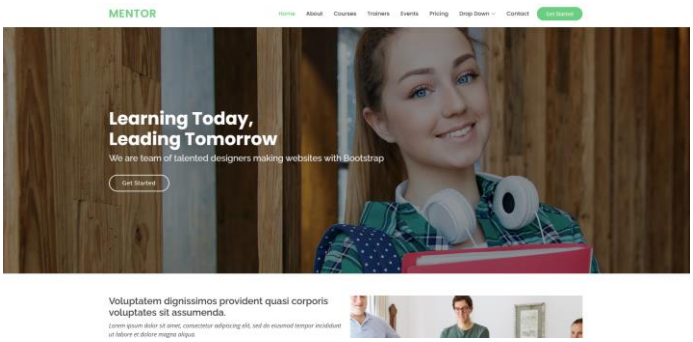
MVC: El patrón de diseño conocido como Modelo-Vista-Controlador, es una propuesta de arquitectura del software que separa el código por sus diferentes tareas, manteniendo diferentes capas que se encargan de realizar una tarea muy particular, lo que ofrece múltiples ventajas. El fundamento de esto es la división del código en tres capas distintas, cada una con sus responsabilidades definidas, conocidas como Modelos, Vistas y Controladores. [2]

Bootstrap: Bootstrap es una biblioteca de herramientas de código abierto que está optimizada para la creación de sitios web y aplicaciones. Esta plataforma utiliza el lenguaje HTML y CSS y cuenta con una variedad de elementos de diseño, como formularios, botones y menús que se adaptan a varios formatos de navegación. [3]

eLearning: Esta es una manera nueva de enseñar y aprender online en cualquier dispositivo digital. Ver vídeos educativos, leer artículos, hacer cuestionarios o incluso cursos de e-learning virtual, en esto consiste realmente el e-learning. [4]

VII. MARCO METODOLÓGICO

La manera en la que se empezó a trabajar con el proyecto fue definiendo una plantilla en la que nos pudiéramos basar para empezar a trabajar en el proyecto. Fue sumamente importante que esta plantilla fuera sencilla de implementar en nuestro diseño MVC, pero también que fuera visualmente atractiva y usara Bootstrap para que de esta manera se pudiera trabajar sin tener que modificar tanto CSS de nuestra parte.



Anexo 1. Plantilla inicial encontrada en línea

Una vez encontrada la plantilla, lo que se requirió fue realizar la base de datos conforme a cómo queríamos que fuera implementado el sistema dentro de nuestra plantilla recién escogida.

Para lograr este objetivo, tuvimos que realizar un diagrama Entidad-Relación y así organizar mejor nuestras ideas antes de plasmarlas en un archivo SQL.

modeloER.drawio.html

Anexo 2. Modelo Entidad-Relación

Después de montar el modelo, era momento de construir un

archivo de SQL Server en el cual se introdujo manualmente toda la información relativa a las tablas y campos que se deseaban hasta que el resultado fue el que se buscaba.

```
USE [master]
GO

CREATE DATABASE [sabado_db]
GO

USE [sabado_db]
GO

CREATE TABLE [dbo].[tCategoria](
    [IdCategoria] [smallint] IDENTITY(1,1) NOT NULL,
    [Nombre] [varchar](50) NOT NULL,
 CONSTRAINT [PK_tCategoria] PRIMARY KEY CLUSTERED
(
    [IdCategoria] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[tRol](
    [IdRol] [smallint] IDENTITY(1,1) NOT NULL,
    [Nombre] [varchar](50) NOT NULL,
 CONSTRAINT [PK_tRol] PRIMARY KEY CLUSTERED
(
    [IdRol] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[tServicio](
    [IdServicio] [bigint] IDENTITY(1,1) NOT NULL,
    [Nombre] [varchar](200) NOT NULL,
    [Precio] [decimal](18, 2) NOT NULL,
    [Imagen] [varchar](500) NOT NULL,
    [Video] [varchar](500) NULL,
    [Estado] [bit] NOT NULL,
 CONSTRAINT [PK_tServicio] PRIMARY KEY CLUSTERED
(
    [IdServicio] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[tUsuario](
    [IdUsuario] [bigint] IDENTITY(1,1) NOT NULL,
    [Correo] [varchar](200) NOT NULL,
    [Contrasenna] [varchar](200) NOT NULL,
    [Nombre] [varchar](200) NOT NULL,
    [IdRol] [smallint] NOT NULL,
    [Estado] [bit] NOT NULL,
    [EsTemporal] [bit] NOT NULL,
    [IdCategoria] [smallint] NOT NULL,
```

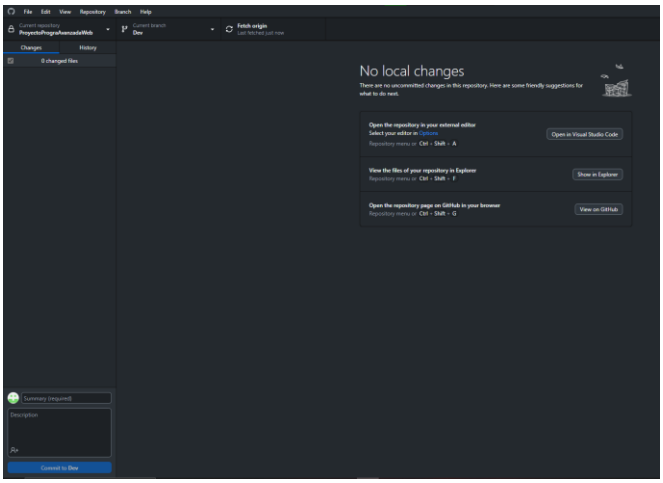
Anexo 3. Parte de la primera versión del SQL

A continuación, se inició por programar la parte del API con los procedimientos almacenados que debían ir en el SQL para luego posteriormente ser utilizados, así como varias modificaciones para optimizar y expandir la base de datos.

```
CREATE PROCEDURE [dbo].[ActualizarServicio]
    @IdServicio BIGINT,
    @Precio decimal(18,2),
    @Video varchar(500)
AS
BEGIN

    UPDATE tServicio
       SET Precio = @Precio,
           Video = @Video
     WHERE IdServicio = @IdServicio

END
GO

CREATE PROCEDURE [dbo].[CambiarContrasenna]
    @Correo                 VARCHAR(200),
    @Contrasenna            VARCHAR(200),
    @ContrasennaTemporal    VARCHAR(200),
    @EsTemporal             BIT
AS
BEGIN

    DECLARE @Consecutivo BIGINT

    SELECT  @Consecutivo = IdUsuario
    FROM    tUsuario
    WHERE   Correo = @Correo
        AND Contrasenna = @ContrasennaTemporal
        AND Estado = 1

    IF @Consecutivo IS NOT NULL
    BEGIN
        UPDATE tUsuario
        SET Contrasenna = @Contrasenna,
            EsTemporal = @EsTemporal
        WHERE Correo = @Correo
    END

    SELECT  IdUsuario,Correo,U.Nombre 'NombreUsuario',U.IdRol,R.Nombre 'NombreRol',Estado,EsTemporal
      FROM  tUsuario U
      INNER JOIN tRol R ON U.IdRol = R.IdRol
      WHERE Correo = @Correo
        AND Estado = 1

END
GO
```

Anexo 4. Ejemplo de Procedimiento Almacenado

Luego, construimos el proyecto API en Visual Studio y conectamos la base de datos local para que pudiera traer los datos de esta. Una vez hecho esto, se clonó el repositorio para todos los integrantes del grupo.



Anexo 5. Repositorio del Proyecto

Ahora que estaba creado el proyecto, era momento de programar el API. Esta programación se dividió de acuerdo con lo visto en el curso en: Entidades, Controladores, Interfaces y Modelos.

```
using SistemaEducacion_API.Entities;

namespace SistemaEducacion_API.Entity
{
    13 referencias
    public class User
    {
        2 referencias
        public int UserID { get; set; }
        2 referencias
        public string? FirstNameUser { get; set; }
        1 referencia
        public string? LastNameUser { get; set; }
        5 referencias
        public string? EmailUser { get; set; }
        3 referencias
        public string? PasswordUser { get; set; }
        0 referencias
        public int? RoleID { get; set; }
        0 referencias
        public string? RoleName { get; set; }
        0 referencias
        public bool? Estado { get; set; }
        1 referencia
        public string? Token { get; set; }
        0 referencias
        public bool Temporary { get; set; }
        1 referencia
        public string? TemporalPassword { get; set; }
        1 referencia
        public int IsTeacher { get; set; } //1 Usuario normal //2 En proceso de aprobacion //3 Teacher
        1 referencia
        public string? PictureUrl { get; set; }
    }

    15 referencias
    public class UserAnswer
    {

        7 referencias
        public UserAnswer()
        {
            Code = "00";
            Message = string.Empty;
        }

        10 referencias
        public string? Code { get; set; }
        10 referencias
        public string? Message { get; set; }
        5 referencias
        public User? Datum { get; set; }
        1 referencia
        public List<User>? Data { get; set; }
```

Anexo 6. Ejemplo de una Entidad del Proyecto

```
[HttpGet]
[Route("ViewProfessorApplicants")]
0 referencias
public IActionResult ViewProfessorApplicants()
{
    using (var db = new SqlConnection(_configuration.GetConnectionString("DefaultConnection")))
    {
        UserAnswer answer = new UserAnswer();
        var result = db.Query<User>("ViewProfessorApplicants",
        commandType: CommandType.StoredProcedure).ToList();

        if (result.Count <= 0)
        {
            answer.Code = "-1";
            answer.Message = "No existe ningún profesor asociado";
        }
        else
        {
            answer.Data = result;
        }
        return Ok(answer);
    }
}

[HttpGet]
[Route("SeeProfesorCourse/{CourseID}")]
0 referencias
public IActionResult SeeProfesorCourse(int CourseID)
{
    using (var db = new SqlConnection(_configuration.GetConnectionString("DefaultConnection")))
    {
        UserAnswer answer = new UserAnswer();

        var result = db.Query<User>("SeeProfesorCourse", new { CourseID }
            , commandType: CommandType.StoredProcedure).FirstOrDefault();

        if (result == null)
        {
            answer.Code = "-1";
            answer.Message = "No hay ningún profesor...";
        }
        else
        {
            answer.Datum = result;
        }

        return Ok(answer);
    }
}
```

Anexo 7. Ejemplo de un Controlador con 2 Acciones

```
namespace SistemaEducacion_API.Services
{
    3 referencias
    public interface IUtilitariosModel
    {
        2 referencias
        string GenerarToken(int UserID);
        3 referencias
        string Encrypt(string text);
        2 referencias
        string GenerateNewPassword();
        2 referencias
        string Decrypt(string texto);
        2 referencias
        void SendEmail(string Recipient, string Subject, string Message);
    }
}
```

Anexo 8. Interfaz IUtilitariosModel Genérica

```
namespace SistemaEducacion_API.Models
{
    1 referencia
    public class UtilitariosModel(IConfiguration _configuration) : IUtilitariosModel
    {
        string SecretKey = _configuration.GetSection("settings:SecretKey").Value ?? string.Empty;


        2 referencias
        public string GenerarToken(int UserID)
        {
            List<Claim> claims = new List<Claim>();
            claims.Add(new Claim(ClaimTypes.Name, Encrypt(UserID.ToString())));

            var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(SecretKey));
            var cred = new SigningCredentials(key, SecurityAlgorithms.HmacSha256Signature);

            var token = new JwtSecurityToken(
                claims: claims,
                expires: DateTime.UtcNow.AddMinutes(10),
                signingCredentials: cred);

            return new JwtSecurityTokenHandler().WriteToken(token);
        }

        3 referencias
        public string Encrypt(string text)
        {
            byte[] iv = new byte[16];
            byte[] array;

            using (Aes aes = Aes.Create())
            {
                aes.Key = Encoding.UTF8.GetBytes(SecretKey);
                aes.IV = iv;

                ICryptoTransform encryptor = aes.CreateEncryptor(aes.Key, aes.IV);

                using (MemoryStream memoryStream = new MemoryStream())
                {
                    using (CryptoStream cryptoStream = new CryptoStream(memoryStream, encryptor, CryptoStreamMode.Write))
                    {
                        using (StreamWriter streamWriter = new StreamWriter(cryptoStream))
                        {
                            streamWriter.Write(text);
                        }

                        array = memoryStream.ToArray();
                    }
                }
            }
```

Anexo 9. Parte del UtilitariosModel

Como pudimos ver anteriormente, el API fue programada para que cada entidad correspondiera con sus debidos atributos de la base de datos y para que sus métodos en el controlador se conectaran de la base de datos, recolectaran la información de acuerdo con el procedimiento almacenado deseado y presentara a la capa web según la solicitud.

Finalmente, un modelo de utilitarios genérico fue programado para que toda la parte de seguridad funcione correctamente y no ocasione operaciones no autorizadas en el sistema en la posterioridad.

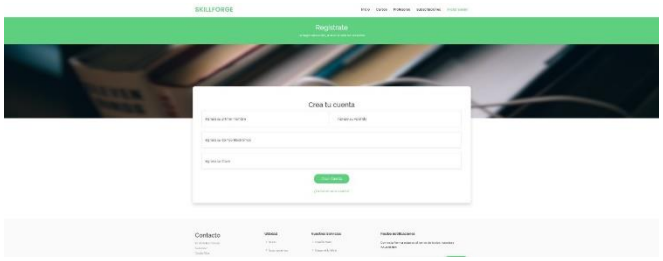
Una vez toda la parte del API estaba configurada, era momento de comenzar con el desarrollo de las vistas y toda la lógica en web.

Anexo 10. Parte del Login

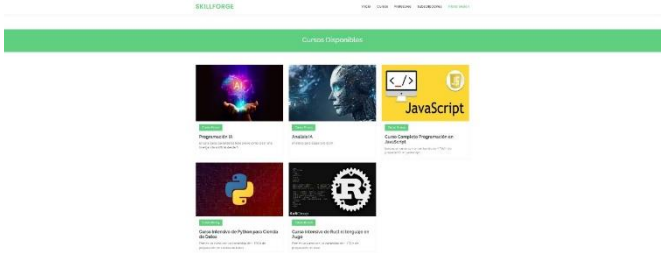
El proceso de inicio de sesión, o login, es una funcionalidad fundamental en cualquier sistema que requiera autenticación de usuarios. En el contexto del programa, el login permite a los usuarios registrados acceder de forma segura a sus cuentas personales. Cuando un usuario desea iniciar sesión, proporciona sus credenciales de acceso, generalmente un nombre de usuario o dirección de correo electrónico junto con una contraseña. Estas credenciales son enviadas al sistema, donde se verifica la identidad del usuario comparando la información proporcionada con los registros almacenados en la base de datos.

Anexo 11. Recuperar contraseña

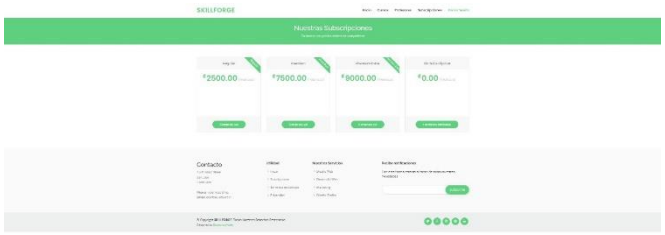
La página de recuperación de contraseña es una funcionalidad esencial en cualquier sistema que requiera autenticación de usuarios. Cuando un usuario olvida su contraseña, esta página le proporciona un medio seguro para restablecerla y recuperar el acceso a su cuenta. En la página de recuperación de contraseña, el usuario generalmente debe proporcionar alguna información de identificación, como su dirección de correo electrónico o nombre de usuario registrado en el sistema. Una vez que se proporciona esta información, el sistema verifica la identidad del usuario y envía un enlace de restablecimiento de contraseña a la dirección de correo electrónico asociada a la cuenta.



Anexo 12. Registrarse

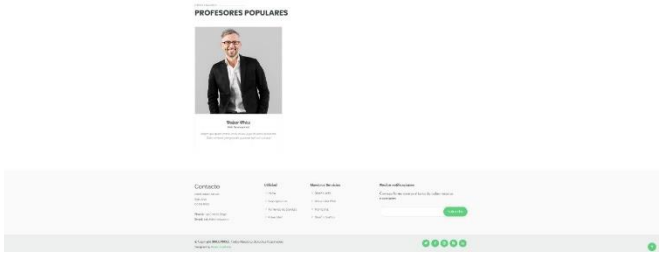


Anexo 13. Cursos Disponibles



Anexo 14. Suscripciones

Anexo 15. Perfil de usuario



Anexo 16. Profesores

VIII. CONCLUSIONES

La conclusión de este proyecto ficticio destaca la importancia de desarrollar sistemas educativos en línea para satisfacer las necesidades cambiantes de aprendizaje en la era digital. A través de la implementación de este sistema, se ha demostrado la viabilidad y el potencial de ofrecer acceso universal a la educación de calidad, promoviendo la flexibilidad, la inclusión y la innovación en la enseñanza y el aprendizaje.

Además, se ha enfatizado la importancia de cumplir con estándares de seguridad y calidad, así como de adoptar prácticas de desarrollo de software sólidas y éticas. La colaboración entre diferentes equipos, incluyendo desarrolladores, diseñadores, administradores y usuarios finales, ha sido fundamental para el éxito del proyecto.

En última instancia, este proyecto ficticio representa un paso hacia adelante en el camino hacia la democratización de la educación y el empoderamiento de las personas a través del conocimiento. A medida que avanzamos hacia el futuro, es crucial seguir innovando y mejorando las plataformas educativas en línea para garantizar un acceso equitativo y una experiencia de aprendizaje enriquecedora para todos.

IX. CONCLUSIONES

[1] Y. Fernández, «API: qué es y para qué sirve,» Xataka, 23 Agosto 2019. [En línea]. Available: https://www.xataka.com/basics/api-que-sirve. [Último acceso: 21 Abril 2024].

[2] M. Á. Álvarez, «Qué es MVC,» DesarrolloWeb.com, 20 Septiembre 2023. [En línea]. Available: https://desarrolloweb.com/articulos/que-es-mvc.html. [Último acceso: 21 Abril 2024].

[3] P. Londoño, «Qué es Bootstrap, para qué sirve y cómo funciona,» HubSpot, 30 Mayo 2023. [En línea]. Available: https://blog.hubspot.es/website/que-es-bootstrap#que-es. [Último acceso: 21 Abril 2024].

[4] H. Colman, «E-learning: Qué es y cómo funciona, beneficios del e-learning,» iSpring, 24 Enero 2023. [En línea]. Available: https://www.ispring.es/blog/what-is-elearning. [Último acceso: 21 Abril 2024].