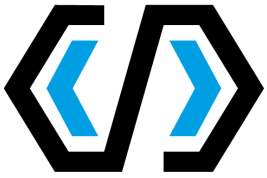
Centro Escolar Catolico San Luis
Manual Técnico

Instituto Tecnico Ricaldone
Tercer año Desarrollo de Software



Desarrolladores:

Fernando Alonso Moreno Flores 20220664

Michael Josué Morales Mejía 20220619

César Andrés López Valdez 20190156

Fernando Alonso Martínez Rosales 2022013

Grupo:	**1A**
Docente:	**Antonio Samuel Benavides Rivas**
Fecha de Entrega:	**4/10/24**
Lugar:	**Instituto Técnico Ricaldone**

ÍNDICE

INTRODUCCIÓN

Este manual se hizo con el fin de dar a entender mejor el funcionamiento y estructuras del sistema, explicando las tecnologías que se utilizaron:

- Estructura de Base de Datos: donde se guarda toda la información, su arquitectura.
- Estructura del Proyecto: donde se explicarán los directorios y archivos del sistema, diseños, requerimientos, instalación y configuraciones.

El objetivo del manual es para que los ingenieros o personas con conocimiento de programación comprendan mucho mejor la lógica y desempeño del sistema.

El documento se orienta a un público conocedor de los aspectos técnicos, por este motivo brindamos toda la información del sistema, desarrollado con sus características y creado para llevar en orden la información de los estudiantes del Colegio Católico San Luis, desarrollando un Sistema de

Gestión Estudiantil con el cual los docentes podrán asignar códigos, control de llegadas tarde, observaciones, control de asistencias, agregar nuevos estudiantes, crear materias y asignar notas. También permite generar reportes, gráficos automatizados, parametrizados y predictivos

Tecnologías Empleadas

Para el Proyecto se utilizó diferentes herramientas, editores y lenguajes para desarrollar el sistema web lo cuales son los siguientes:

- **Servidor Web**: Se ocupó el servidor **XAMPP** paquete de servidores para PHP, incluye un servidor Apache, una base de datos MariaDB, PHP. Con XAMPP, se pudo ejecutar los servicios y la aplicación web localmente.

- **Lenguajes de Programación:** Los lenguajes de programación que se utilizaron para el lado del cliente y del servidor son:

    1- PHP: Se utilizó para la programación del sistema, los cruds, validaciones, las librerías de reportes y gráficos, consultas a la base de datos por medio API de modelos Handler y Data.

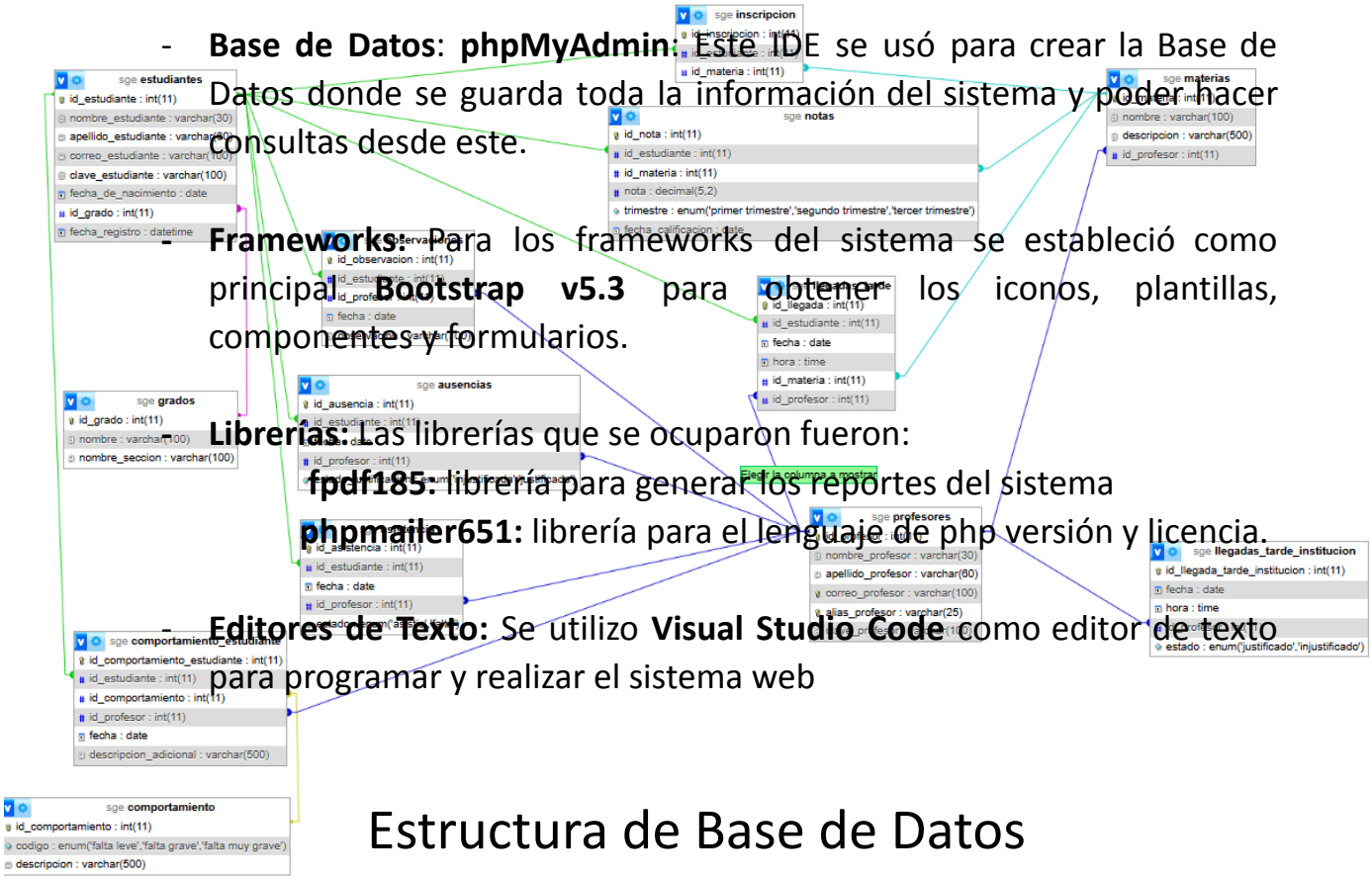
2- **JavaScript**: Se empleó para programar, validar las clases y servicios para realizar consultas a la Base de Datos, también para implementar componentes para la parte del cliente.

- **Base de Datos**: **phpMyAdmin:** Este IDE se usó para crear la Base de Datos donde se guarda toda la información del sistema y poder hacer consultas desde este.

- **Frameworks:** Para los frameworks del sistema se estableció como principal **Bootstrap v5.3** para obtener los iconos, plantillas, componentes y formularios.

- **Librerías:** Las librerías que se ocuparon fueron:
  fpdf185: librería para generar los reportes del sistema
  phpmailer651: librería para el lenguaje de php versión y licencia.

- **Editores de Texto:** Se utilizo **Visual Studio Code** como editor de texto para programar y realizar el sistema web



Estructura de Base de Datos

Diccionario de Datos

Tabla Estudiante:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_estudiante	int	11	No	Llave primaria	autonumerico
nombre_estudiante	Varchar	30	No		ninguno
apellido_estudiante	varchar	60	No		ninguno
correo_estudiante	varchar	100	No		ninguno

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_estudiante	int	11	No	Llave primaria	autonumerico
nombre_estudiante	Varchar	30	No		ninguno
apellido_estudiante	varchar	60	No		ninguno
clave_estudiante	varchar	100	No		ninguno
Fecha_de_nacimiento	date		Si		Null
id_grado	int	11	No	Llave foreana	ninguno
Fecha_registro	datetime		No		Current_timestamp

Tabla Profesor:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado

id_profesor	int	11	No	Llave primaria	autonumerico
nombre_profesor	varchar	30	No		ninguno
apellido_profesor	varchar	60	No		ninguno
correo_profesor	varchar	100	No	Llave foreana	ninguno
clave_profesor	varchar	100	No	Llave foreana	ninguno
alias_profesor	varchar	25	No		ninguno

Tabla Grados:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_grado	int	11	No	Llave primaria	autonumerico
nombre	varchar	100	No		ninguno
Nombre_seccion	varchar	100	No		ninguno

Tabla Materias:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_materia	int	11	No	Llave primaria	autonumerico
nombre	varchar	100	No		ninguno
descripcion	varchar	500	Si		Null
id_profesor	int	11	Si	Llave foreana	Null

Tabla Inscripción:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_inscripcion	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
id_materia	int	11	No	Llave foreana	Llave foreana

Tabla Observación:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_observacion	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
fecha	date		No		ninguno
id_profesor	int	11	No	Llave foreana	ninguno
observacion	varchar	700	No		ninguno

Tabla Ausencia:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_ausencia	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
fecha	date		No		ninguno
id_profesor	int	11	No	Llave foreana	ninguno
Estado_justificacion	enum('injustificada', 'justificada')		No		injustificada

Tabla Asistencia:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_asistencia	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
fecha	date		No		ninguno
id_profesor	int	11	No	Llave foreana	ninguno
estado	enum('asistio', 'falta')		No		ninguno

Tabla Comportamiento:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_comportamiento	int	11	No	Llave primaria	autonumerico
codigo	enum('falta leve', 'falta grave', 'falta muy grave.		No		ninguno
descripcion	varchar	500	No		ninguno

Tabla Comportamiento Estudiante:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_comportamiento_estudiante	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
Id_comportamiento	int	11	No	Llave foreana	ninguno
Id_profesor	int	11	No	Llave foreana	ninguno
fecha	date		No		ninguno
descripción_adicional	varchar	500	Si		Null

Tabla Llegada Tarde:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_llegada	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
Id_materia	int	11	No	Llave foreana	ninguno
Id_profesor	int	11	No	Llave foreana	ninguno
fecha	date		No		ninguno
hora	time		No		ninguno

Tabla Llegada Tarde Institución:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_llegada_tarde_institucion	int	11	No	Llave primaria	autonumerico
Id_profesor	int	11	No	Llave foreana	ninguno
fecha	date		No		ninguno
hora	time		No		ninguno
estado	enum('asistio', 'falta')		No		ninguno

Tabla Notas:

Columna	Tipo	Longitud	Nulo	Índice	Predeterminado
id_nota	int	11	No	Llave primaria	autonumerico
Id_estudiante	int	11	No	Llave foreana	ninguno
nota	decimal(5,2)		No		ninguno
id_materia	int	11	No	Llave foreana	ninguno
trimestre	enum('primer trimestre', 'segundo trimestre', 'ter...		No		ninguno
fecha_calificacion	date		No		ninguno

Arquitectura del Software

Este sistema de gestión estudiantil es lo que usamos en el colegio San Luis para llevar control de todo lo relacionado con los alumnos: desde las notas, el comportamiento, las asistencias, hasta las materias que cursan. Está compuesto por varias partes que se conectan entre sí para que todo funcione de manera eficiente y organizada.

1. Capa de Datos:

En esta parte es donde se guarda toda la información importante, como los datos de los alumnos, profesores, materias, y todo lo demás. Cada uno de estos elementos tiene su propia "tabla" en la base de datos. Esas tablas se conectan entre sí para que podamos saber, por ejemplo, qué materias lleva un estudiante o qué profesor da cada clase.

Tablas Principales:

Estudiantes (sge estudiantes): Aquí se guarda todo sobre los estudiantes: nombre, apellido, correo, fecha de nacimiento y el grado que están cursando.

Grados (sge grados): Esta tabla tiene los nombres de los diferentes grados (por ejemplo, primero, segundo, tercero, etc.) y sirve para saber a qué grado pertenece cada estudiante.

Materias (sge materias): Aquí están listadas todas las materias que se imparten en el colegio y qué profesor es responsable de cada una.

Profesores (sge profesores): Almacena la información de los profesores que dan clases en el colegio.

Tablas Relacionales:

Estas tablas son las que hacen las conexiones entre las principales para poder manejar otros datos importantes, como las notas, la asistencia, etc.

Inscripción (sge inscripción): Aquí se asocia a los estudiantes con las materias en las que están inscritos.

Notas (sge notas): Esta tabla guarda las calificaciones de los estudiantes en las materias que llevan, separadas por trimestres.

Observaciones (sge observaciones): Aquí los profesores pueden registrar comentarios o notas adicionales sobre los estudiantes, como observaciones de comportamiento o desempeño.

Comportamiento Estudiantil (sge comportamiento estudiante): Guarda registros sobre la conducta de los estudiantes, indicando si han tenido algún comportamiento destacado o problemas de disciplina.

Asistencia y Llegadas Tarde:

También hay una parte del sistema que se encarga de controlar si los estudiantes asisten a clases o llegan tarde, lo que permite llevar un registro detallado.

Asistencia (sge asistencia): Esta tabla almacena si los estudiantes asistieron o no a clases.

Ausencias (sge ausencias): Aquí se registran las faltas, indicando si son justificadas o no.

Llegadas Tarde (sge llegadas tarde): Guarda las llegadas tarde de los estudiantes, con la posibilidad de justificar si la tardanza fue por una razón válida o no.

2. Capa de Lógica de Negocios:

Esta es la parte del sistema donde se manejan todas las reglas y procesos. Es lo que permite, por ejemplo, que los profesores puedan registrar las notas de sus alumnos, o marcar si alguien faltó a clase.

Gestión de Notas: Los profesores pueden ingresar las notas de los estudiantes por materia y trimestre. Estas notas se vinculan automáticamente a los estudiantes y sus inscripciones.

Control de Asistencia: Se puede registrar si un estudiante asistió o no, y también si llegó tarde. Todo queda guardado y se puede consultar más adelante.

Registro de Comportamiento: Los profesores también pueden añadir observaciones sobre el comportamiento de los alumnos, clasificando si la conducta fue buena, regular, o si hubo algún problema.

3. Capa de Presentación:

Esta es la parte visual del sistema, lo que ve el usuario (administrador, profesor, estudiante). Desde aquí es donde se interactúa con todos los datos que están almacenados en el sistema.

Panel del Administrador: Permite a los administradores del colegio tener una vista general de todo lo que está ocurriendo: ver estudiantes, materias, grados, y generar reportes de asistencia o comportamiento.

Panel del Profesor: Los profesores tienen su propio espacio donde pueden ver a los estudiantes que tienen a cargo, registrar las notas, la asistencia, y hacer observaciones sobre el comportamiento.

4. Cómo Interactúan Todos los Componentes:

Todas estas tablas y capas interactúan entre sí gracias a las relaciones entre las tablas. Por ejemplo, un estudiante está asociado a un grado, que a su vez está relacionado con las materias que cursa y con los profesores que enseñan esas materias.

Estas conexiones permiten que, por ejemplo:

Se puede ver un reporte completo con las calificaciones, las faltas a clases y cualquier comentario del profesor sobre el comportamiento de un estudiante.

Los administradores puedan tener todo bajo control, sin perder detalles importantes como las llegadas tarde o las ausencias.

Este sistema está diseñado para ayudar al colegio San Luis a gestionar todo lo relacionado con la vida académica de los estudiantes, haciendo que tanto profesores como administradores tengan un control claro y centralizado de la información, y los estudiantes puedan consultar sus datos de forma fácil.

Estructura del Proyecto

api/: Este directorio suele contener los puntos de acceso de la API (Application Programming Interface) que la aplicación expone. Estos endpoints pueden estar escritos en lenguajes como PHP, Node.js, etc. Los archivos dentro de este directorio se encargan de manejar las solicitudes HTTP y devolver respuestas en formato JSON, XML u otros.

controllers/:

Los controladores son responsables de procesar las solicitudes entrantes. Este directorio contiene los controladores de la aplicación, que actúan como intermediarios entre los modelos y las vistas. Cada controlador maneja la lógica de la aplicación para diferentes rutas o funcionalidades.

resources/:

Es común que este directorio almacena archivos como vistas, plantillas, traducciones, archivos de configuración, entre otros recursos de la aplicación. Podría también contener scripts o estilos que sean utilizados en el frontend.

vendor/:

En aplicaciones PHP que utilizan Composer, este directorio contiene todas las dependencias de terceros que se hayan instalado. Estas librerías son usadas por el sistema para añadir funcionalidades que no están incluidas en el lenguaje base.

view/:

Aquí se ubican las vistas, que contienen el HTML, CSS y scripts que se renderizan en el lado del cliente. En un patrón MVC (Modelo-Vista-Controlador), este directorio se encarga de la representación visual de los datos procesados por los controladores.

htaccess:

Archivo de configuración de Apache que permite definir reglas de redireccionamiento, control de acceso y configuración del servidor web. Puede ser clave para la seguridad y el SEO de la aplicación.

buenas_practicas/:

Este directorio contiene guías o documentos que describen las mejores prácticas a seguir en el desarrollo del proyecto, asegurando que todos los desarrolladores trabajen bajo los mismos estándares.

composer y composer.lock:

composer.json define las dependencias del proyecto (librerías de terceros) y la configuración básica de Composer. El archivo composer.lock asegura que todos los desarrolladores utilicen exactamente las mismas versiones de las dependencias.

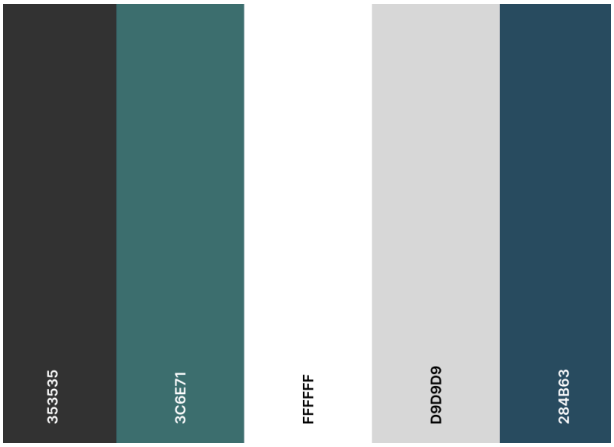
estandares_javascript/ y estandares_php/:

Estos directorios contienen guías o reglas específicas de codificación para los lenguajes JavaScript y PHP, respectivamente. Los equipos suelen definir estándares de codificación para mantener el código limpio, mantenible y coherente.

Diseño de la Aplicación

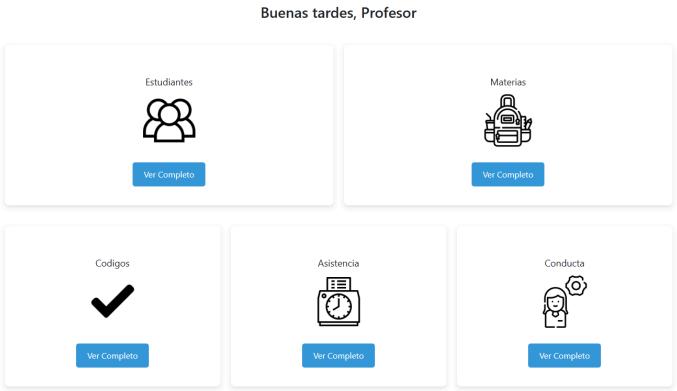
Paleta de Colores: Se escogió esta paleta ya que los colores del logos del instituto concordaba con los colores selectos para el sistema web

- Elementos: Se ocuparon los colores blanco y gris para el fondo de pantallas del sistema, para los botones fue el color azul tambien para las diferentes vistas y para los textos y cuerpo del pie de pagina el color negro.



Tipos y tamaños de texto: Para el tipo de fuente se implementó **Arial** y el tamaño de fuente se aplicó de **12** para contraste con las pantallas.

Dimensión de imágenes utilizadas: El tamaño de las imágenes que se utilizaron para las vistas de cada apartado fue de 24px y para los encabezados fue de 10px.



Buenas prácticas de Desarrollo

Estandares de Programacion:

1- **JavaScript:**

- Nombre de Variables: Los nombres de variables fueron escritos con notación camelCase.
- Las llaves de las funciones: Empezar en la misma línea que se declara la función y debe haber un espacio entre el paréntesis de cierre de argumentos y la llave de inicio de cuerpo de función
- Tipos de funciones**:** Las funciones o funciones como variables. Ya que las funciones como variables se aplicó usando punto y coma (;) después del cuerpo de la función para finalizar las sentencia.
- Nombres de funciones: Los nombres de funciones deben ser escritos con notación camelCase.
- Argumentos de una función: Para la legibilidad del código javascript los argumentos de funciones se ordenaron por números con notación camelCase. Los argumentos se separan por comas (,) y un espacio.

2- **PHP**:

-Los archivos se utilizaron únicamente etiquetas `<?php` y `<?=`.

- El código se utilizó 4 espacios para sangría, no tabulaciones

- Los métodos que se aplicó fue una cadena como mensaje o un objeto con un toString()método.

-Los nombres de las clases se colocaron en CamelCase empezando con mayúscula.

- Los nombres de los métodos deben estar en camelCase empezando con minúscula

3- **Base de Datos:**

- Se utilizó la notación Underscore Separated, ocupando el guión bajo (_) para separar palabras y en minúsculas.
- Se usó las letras “id” en las columnas de clave primaria y foránea.
- No se incluyo letras con tilde ni Ñ, ñ, letras con diéresis.
- Se implementó solo minúsculas para nombrar los elementos de la base de datos, esquemas, vistas, funciones, tablas, campos.
- No se usó caracteres especiales como por ejemplo ($, #, *, – +, etc., ´, ¨,).

4- **Comentarios:**

- Se incluye indistintamente comentarios de una sola línea con "//".
- Se utilizó comentarios de más de una sola línea con "/*".

Requerimientos de Hardware y Software

Hardware y Software:

Pc o Laptop Procesador Intel(R) Core(TM) i5 2.50GHz , Ram mínimo de 4GB, recomendable 8GB, Disco duro SSD mínimo de 200GB, recomendable 500GB.

Instalación y Configuración

Este sistema web puede ser configurado en dos tipos de entornos: local y en línea bajo el host de hamacoteca.online. A continuación, se describirá a detalle los pasos para cada uno de ellos.

1. Instalación en entorno local.

Requisitos previos.

Antes de comenzar la instalación, asegúrate de tener los siguientes componentes instalados:

- Servidor web: Apache v2.4.54.
- Lenguaje de programación: PHP v8.2.0.
- Gestor de base de datos: MariaDB v10.4.27.
- Administrador de dependencias de PHP: Composer v2.7.6.
- Para clonar el repositorio del proyecto: Git.

Pasos para la instalación

- **Clonar el proyecto o descargar el código fuente.**

Clona el repositorio del sistema web o descarga el archivo zip desde el repositorio:

https://github.com/alonso134/Expo2024.git

- **Configurar la base de datos.**

Crea una base de datos en tu servidor local de MariaDB e importa el archivo .sql que se encuentra en el siguiente enlace proporcionado:

https://drive.google.com/file/d/1UY2FkOX7huJ9GmKBlD0IAGimj-jWtu0-/view?usp=sharing

Esto creará las tablas necesarias para que el sistema funcione.

- **Configurar el archivo de conexión a la base de datos.**

Modifica el archivo de configuración (config.php) con las credenciales correctas de la base de datos, este se encuentra en la carpeta /api/helpers/.

- **Instalar dependencias.**

Como el proyecto utiliza librerías externas de php, utiliza Composer para gestionar dependencias, ejecuta el siguiente comando en la raíz del proyecto:

composer install

- **Prueba en el navegador.**

Accede a https://hamacoteca.online/Expo2024/view/admin/ o la URL configurada en tu servidor local para verificar que el sistema esté funcionando correctamente