Práctica Integrada DAW: Plataforma de Entrenamientos

1. Contexto

Esta práctica integra los módulos de **Desarrollo Web en Entorno Servidor (DWES)** y **Desarrollo Web en Entorno Cliente (DWEC)** del ciclo formativo **DAW**. El alumnado deberá desarrollar una **aplicación web completa (frontend + backend)** orientada a la gestión de entrenamientos deportivos, especializado en el ámbito del ciclismo.

La aplicación seguirá una arquitectura **cliente–servidor**, donde el backend expone una **API REST** y el frontend consume dicha API mediante **AJAX (fetch/XHR)**.

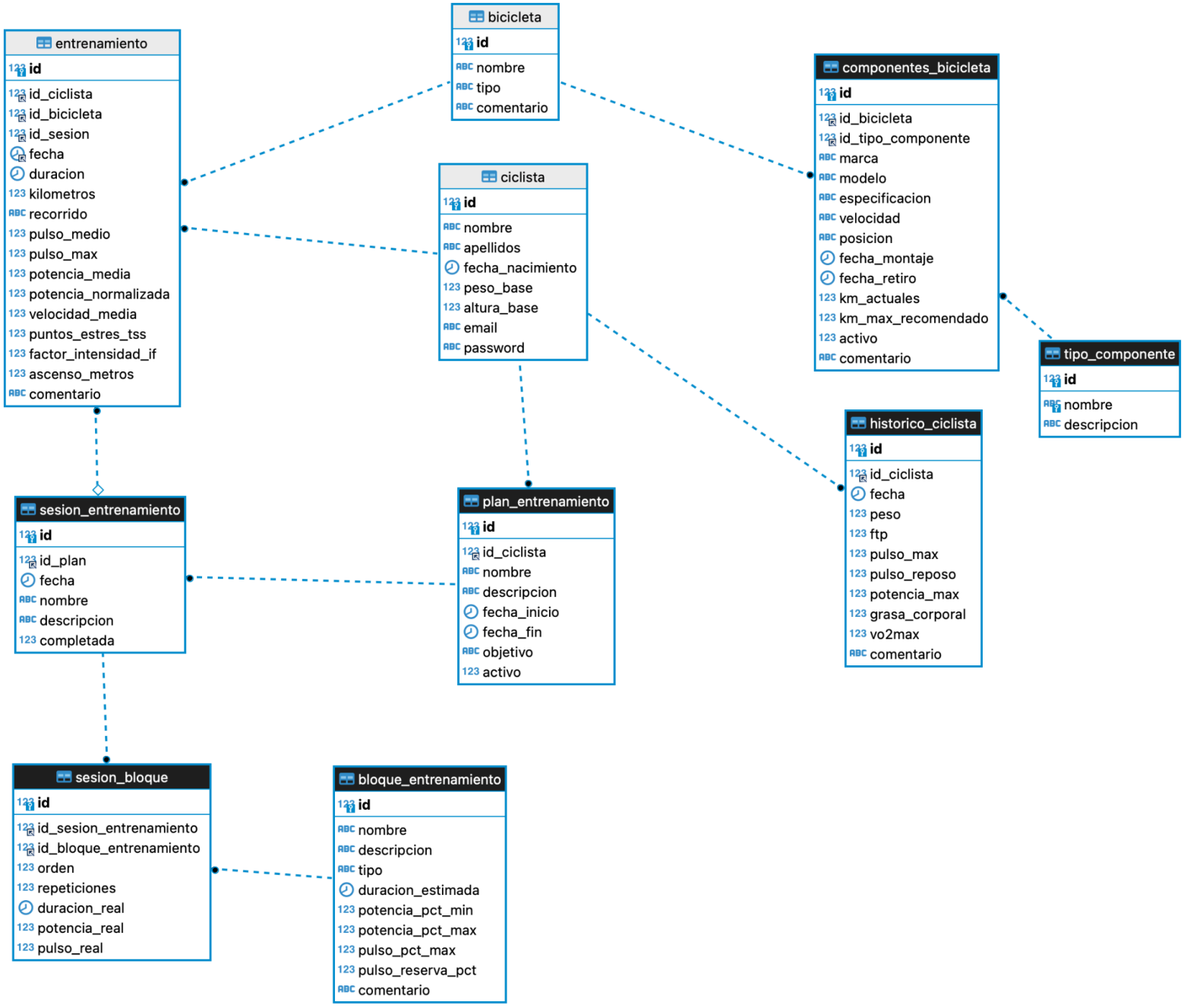
2. Objetivo general

Diseñar e implementar una aplicación web que permita:

- Gestionar entrenamientos (bloques, planes y sesiones).
- Registrar y consultar resultados de sesiones.
- Autenticar usuarios.
- Consumir datos del servidor de forma asíncrona desde el cliente.

3. Modelo de datos propuesto

El backend deberá implementar el modelo entidad–relación proporcionado en el siguiente diagrama


entrenamiento
id
id_ciclista
id_bicicleta
id_sesion
fecha
duracion
kilometros
recorrido
pulso_medio
pulso_max
potencia_media
potencia_normalizada
velocidad_media
puntos_estres_tss
factor_intensidad_if
ascenso_metros
comentario
bicicleta
id
nombre
tipo
comentario
componentes_bicicleta
id
id_bicicleta
id_tipo_componente
marca
modelo
especificacion
velocidad
posicion
fecha_montaje
fecha_retiro
km_actuales
km_max_recomendado
activo
comentario
tipo_componente
id
nombre
descripcion
ciclista
id
nombre
apellidos
fecha_nacimiento
peso_base
altura_base
email
password
historico_ciclista
id
id_ciclista
fecha
peso
ftp
pulso_max
pulso_reposo
potencia_max
grasa_corporal
vo2max
comentario
sesion_entrenamiento
id
id_plan
fecha
nombre
descripcion
completada
plan_entrenamiento
id
id_ciclista
nombre
descripcion
fecha_inicio
fecha_fin
objetivo
activo
sesion_bloque
id
id_sesion_entrenamiento
id_bloque_entrenamiento
orden
repeticiones
duracion_real
potencia_real
pulso_real
bloque_entrenamiento
id
nombre
descripcion
tipo
duracion_estimada
potencia_pct_min
potencia_pct_max
pulso_pct_max
pulso_reserva_pct
comentario

3.1. Explicación de las tablas que componen el modelo

A continuación se describen las distintas tablas que forman el modelo:

Tabla ciclista. Es el elemento principal de la aplicación. Todos los usuarios son ciclistas. Se puede identificar a un usuario con el campo email y password de esta tabla. En el registro, el atleta proporcionará sus datos personales: nombre, apellidos, fecha_nacimiento, peso y altura.

Tabla bloque_entrenamiento. Se define un bloque como una actividad dentro de una sesión de entrenamiento. Por ejemplo, un bloque podría definirse como el calentamiento, configurado con una duración de 20 minutos y datos de potencia y pulso bajos para representar baja intensidad.

Tabla sesión_entrenamiento. Una sesión de entrenamiento es la puesta en marcha de un conjunto de bloques en un día concreto. Es similar a la acción de agendar un entrenamiento para realizar en un día concreto del calendario. Puede usarse un calendario para representar todas las sesiones. Las relación entre sesion_entrenamiento y bloque_entrenamiento es N:M para lo que se utiliza la tabla sesion_bloque para unirlas. En la tabla **sesion_bloque** además se especifican campos específicos de esta unión, como por ejemplo el orden o las repeticiones del bloque en esa sesión concreta.

A continuación se muestran la descripción de dos sesiones que utilizarían los mismos 3 bloques, pero cada uno con un número de repeticiones distinto.

Sesión 1: 19-1-2026 Bloques: ● Bloque 1 - Calenamiento. 1 repetición, orden 1 ● Bloque 2 - Series. 5 repeticiones, orden 2 ● Bloque 3 - Vuelta a la calma. 1 repetición, orden 3	**Sesión 2**: 21-1-2026 Bloques: ● Bloque 1 - Calenamiento. 2 repetición, orden 1 ● Bloque 2 - Series. 6 repeticiones, orden 2 ● Bloque 3 - Vuelta a la calma. 1 repetición, orden 3

Tabla plan_entrenamiento. Es la definición de un plan completo asignado a un ciclista. Es decir, el conjunto de sesiones que se deben realizar para poder llegar a un objetivo. Por ejemplo, para ser capaz de hacer una tirada de 80km, es necesario realizar 10 sesiones con sus correspondientes bloques distribuidos en 2 semanas.

Tabla entrenamiento. Es el resultado obtenido en una sesión de entrenamiento. En esta sesión, el ciclista especificará con qué bicicleta se ha hecho el entrenamiento, el tiempo de la actividad, los kilometros totales y las estadísticas de rendimiento como pulso, potencia, velocidad, etc)

El resto de tablas se proporcionarán con los datos fijos para ser usados. No es necesario desarrollar funcionalidades para añadir datos.

Es importante incluir la o las migraciones necesarias según el modelo indicado. Es importante que si te equivocas, intentes crear nuevas migraciones, no modificar las existentes que ya estén ejecutadas.

Se deberán respetar las claves primarias y foráneas indicadas. Las tablas que no están incluidas en este punto no es necesario desarrollar el modelo en la aplicación.

4. Requisitos funcionales

La aplicación deberá cumplir, como mínimo, los siguientes requisitos:

1. **Usuarios**
    - Registro de usuario.
    - Inicio de sesión (login).
    - Cierre de sesión (logout).
2. **Bloques de entrenamiento**
    - Crear bloque de entrenamiento.
    - Ver detalle de un bloque.
    - Eliminar bloque.
    - Listar bloques.
3. **Sesión de bloques**
    - Crear sesión de bloque.
    - Editar sesión de bloque.
    - Eliminar sesión de bloque.

    - Listar sesiones de bloques.
4. **Sesiones de entrenamiento**
    - Crear sesión de entrenamiento.
    - Eliminar sesión de entrenamiento.
    - Listar sesiones de entrenamiento.
5. **Plan de entrenamiento**
    - Crear plan de entrenamiento.
    - Eliminar plan de entrenamiento.
    - Editar plan de entrenamiento
    - Listar planes de entrenamiento.
6. **Resultados de entrenamiento**
    - Registrar resultados de una sesión de entrenamiento.
    - Consultar resultados de una sesión.

5. Backend (DWES)

5.1 Tecnología

Es obligatorio el uso de las siguientes tecnologías:

- Lenguaje: PHP
- Framework: Laravel
- Base de datos: MySQL
- API REST con respuestas en JSON

5.2 Rutas mínimas de la API

Autenticación	**Bloques de entrenamiento**	**Planes de entrenamiento**
POST /register POST /login POST /logout	GET /bloque → listar bloques POST /bloque/crear GET /bloque/{id} DELETE /bloque/{id}/eliminar	GET /plan POST /plan/crear PUT /plan/{id} DELETE /plan/{id}

Sesiones de entrenamiento	**Resultados de entrenamiento**	**Sesión-plan entrenamiento**
GET /sesion → listar sesiones POST /sesion/crear DELETE /sesion/{id} GET /sesion/{id}	POST /resultado/crear GET /resultado/{id}	GET /sesionbloque POST /sesionbloque/crear DELETE /sesionbloque/{id}

Todas las rutas privadas deberán requerir autenticación mediante token de sesión. Debes controlar todos los errores posibles y devolver el código de error correspondiente.

6. Frontend (DWEC)

6.1 Tecnología

- HTML5, CSS3
- JavaScript (ES6)
- Uso obligatorio de **fetch / AJAX**
- No se permite recarga completa de página para mostrar datos dinámicos

6.2 Requisitos del cliente

- Interfaz clara y usable.
- Formularios para altas, edición y eliminación.
- Manejo de errores del servidor.
- Renderizado dinámico de datos recibidos en JSON.

6.3 Uso obligatorio de AJAX (requisito clave)

Se deberá implementar al menos un caso avanzado de carga asíncrona:

Listado de sesiones con scroll infinito

- El endpoint /sesion devolverá las sesiones paginadas (por ejemplo, de 10 en 10).
- El frontend cargará automáticamente más sesiones cuando el usuario haga scroll hasta el final.
- No se permite paginación clásica con recarga.

Ejemplo:

GET /sesion?offset=0&limit=10
GET /sesion?offset=10&limit=10

8. Entregables

1. Código fuente del proyecto.
2. Enlace a GitHub del proyecto (con acceso)

3. Video de demostración del funcionamiento de la web
4. Memoria del proyecto explicando al menos los siguientes puntos:
    - Cómo ejecutar el proyecto.
    - Decisiones técnicas tomadas.
    - Proceso de desarrollo
        i. Comandos laravel ejecutados para la creación de todos los elementos de backend
        ii. Migraciones creadas y explicación de la necesidad de cada una de ellas
        iii. Documentación del API REST. Documentar todas las rutas, con el formato de la petición, los distintos parámetros que pueden recibir las peticiones y el formato de la respuesta.
        iv. Pruebas del backend. Puedes hacerlo con Postman.
        v. Problemas surgidos durante el desarrollo y soluciones.

La memoria tendrá un peso importante en la nota. Debe mostrar el proceso de desarrollo y reflejar la comprensión del proyecto desarrollado, coherencia entre lo explicado y el código entregado, y evidencias claras de autoría propia mediante explicaciones personalizadas, razonadas y consistentes del trabajo realizado. Memorias genéricas, poco detalladas o que no demuestren dominio del proyecto podrán afectar negativamente a la calificación.