

TALLER PRÁCTICO

Programación asistida por IA: construcción de una aplicación web completa en PHP y MySQL

Nivel: Bootcamp inicial/intermedio **Modalidad:** práctica guiada **Duración sugerida:** 12–16 horas

Propósito: que el estudiante aprenda a utilizar una secuencia de prompts especializados para pasar de unos requerimientos de cliente a una aplicación web funcional, organizada, responsiva y conectada a MySQL mediante PHP.

Producto final: una aplicación llamada **Gestión de Citas Médicas**, con pacientes, médicos, especialidades y citas; interfaz HTML/CSS/JS, base de datos MySQL y CRUD en PHP.

1. Objetivos de aprendizaje

- Analizar requerimientos y convertirlos en instrucciones precisas para una IA.
- Generar interfaces frontend modernas, responsivas y accesibles mediante prompts.
- Diseñar un modelo de datos y convertirlo en SQL para MySQL.
- Generar y revisar CRUD en PHP usando una arquitectura organizada.
- Integrar formularios, PHP y MySQL sin mezclar innecesariamente presentación y lógica.
- Utilizar IA para detectar, explicar y corregir errores de frontend, backend y base de datos.
- Verificar el resultado mediante pruebas funcionales y documentar los cambios.

2. Reglas de trabajo con IA

- **No aceptar código a ciegas:** el estudiante debe leer, ejecutar y comprender lo generado.
- **Trabajar por etapas:** no pedir toda la aplicación en un único prompt gigantesco.
- **Conservar contexto:** mantener nombres de tablas, campos, archivos y funcionalidades consistentes.
- **Probar después de cada etapa:** una aplicación funcional se construye mediante ciclos de generar → ejecutar → revisar → corregir.
- **Solicitar explicaciones:** cuando aparezca código desconocido, pedir a la IA que lo explique de forma sencilla.
- **Seguridad:** pedir validación de entradas, consultas preparadas, control de errores y protección de credenciales.

3. Caso práctico: Clínica VitalCare

Situación: una pequeña clínica necesita una aplicación web para administrar pacientes, médicos, especialidades y citas. Actualmente registra la información en hojas de cálculo y desea centralizarla.

Módulo	Datos principales	Operaciones
Pacientes	id, documento, nombre, apellidos, teléfono, dirección, fecha_nacimiento	Crear, listar, editar, eliminar
Médicos	id, nombre, apellidos, teléfono, especialidad_id	Crear, listar, editar, eliminar
Especialidades	id, nombre, descripción	Crear, listar, editar, eliminar
Citas	id, paciente_id, medico_id, fecha, hora, motivo, estado	Crear, listar, editar, cancelar/eliminar

Restricciones: PHP 8+, MySQL/MariaDB, HTML5, CSS3, JavaScript vanilla y Bootstrap opcional. El proyecto debe poder ejecutarse en un entorno local como XAMPP, Laragon o similar.

4. Prompt 1 — Analizar requerimientos y planificar la aplicación

Prompt para practicar:

Actúa como analista de sistemas y arquitecto de aplicaciones web.
A partir de los siguientes requerimientos de una clínica: [pegar caso VitalCare], realiza:

1. resumen funcional;
2. actores y funcionalidades;
3. módulos;
4. entidades y relaciones;
5. reglas de negocio;
6. pantallas necesarias;
7. estructura de carpetas propuesta;
8. tecnologías a utilizar.

No inventes funcionalidades que no estén justificadas. Presenta el resultado de forma clara y lista para usar como especificación del proyecto.

Producto esperado: Entregable: documento breve con módulos, funcionalidades, reglas y estructura de carpetas.

Reto adicional: Reto: pedir a la IA que identifique ambigüedades y genere 5 preguntas que el cliente debería responder antes de programar.

5. Prompt 2 — Diseñar la interfaz frontend

Prompt para practicar:

Actúa como diseñador UI/UX y desarrollador frontend.
Usando exclusivamente los requerimientos del proyecto VitalCare, diseña una interfaz web moderna, profesional, atractiva y responsiva.
Genera:

- index.html con navegación y dashboard;
- formularios independientes para pacientes, médicos, especialidades y citas;
- tablas/listados;
- styles.css independiente;
- validaciones HTML5 básicas;
- diseño adaptable a móvil, tablet y escritorio;
- accesibilidad básica.

Usa nombres de archivos claros y no mezcles CSS dentro de los HTML.
Explica brevemente la función de cada archivo.

Producto esperado: Entregable: carpeta frontend con index.html, formularios HTML y styles.css.

Reto adicional: Reto: pedir tres variantes visuales y seleccionar una justificando criterios de legibilidad, consistencia y usabilidad.

6. Prompt 3 — Crear el modelo de datos y SQL MySQL

Prompt para practicar:

Actúa como diseñador de bases de datos.
A partir de los requerimientos de VitalCare, diseña:

1. modelo entidad-relación descrito textualmente;
2. modelo relacional;
3. tablas, claves primarias y foráneas;
4. tipos de datos;
5. restricciones;
6. script SQL completo para crear la base de datos;
7. INSERT de datos de prueba.

Usa nombres consistentes con los formularios y prepara el diseño para consultas y CRUD en PHP.
No agregues entidades sin justificación.

Producto esperado: Entregable: database.sql y una explicación del modelo.

Reto adicional: Reto: solicitar a la IA que revise normalización, integridad referencial e índices y proponga ajustes explicados.

7. Prompt 4 — Generar conexión segura PHP/MySQL

Prompt para practicar:

Actúa como desarrollador backend PHP.
Crea una conexión independiente para MySQL para el proyecto VitalCare.
Requisitos:

- PHP 8+;
- PDO;
- manejo de excepciones;
- credenciales separadas de la lógica de negocio;
- charset UTF-8;
- archivo db.php independiente;
- ejemplo de prueba de conexión;
- no mostrar credenciales ni errores sensibles al usuario final.

Explica cada bloque del código para un estudiante principiante.

Producto esperado: Entregable: db.php y prueba de conexión funcional.

Reto adicional: Reto: pedir una segunda versión usando variables de entorno y comparar ventajas y limitaciones.

8. Prompt 5 — Generar CRUD PHP por módulos

Prompt para practicar:

Actúa como desarrollador PHP especializado en CRUD.
Genera los scripts PHP independientes para el CRUD de pacientes, médicos, especialidades y citas de VitalCare.
Usa PDO y consultas preparadas.
Cada módulo debe permitir listar, crear, consultar, actualizar y eliminar cuando corresponda.
Incluye:

- validación de datos;
- mensajes de éxito/error;
- manejo de excepciones;
- consultas preparadas;
- separación clara entre formulario, lógica y acceso a datos;
- compatibilidad con la estructura de tablas definida en database.sql.

No cambies nombres de campos sin justificarlo.

Producto esperado: Entregable: carpeta php/ con los scripts CRUD organizados por módulo.

Reto adicional: Reto: pedir a la IA que revise cada DELETE y explique cómo evitar eliminaciones accidentales.

9. Prompt 6 — Integrar frontend, PHP y MySQL

Prompt para practicar:

Actúa como integrador full-stack.
Integra el frontend de VitalCare con los scripts PHP y MySQL ya generados.
Mantén independientes HTML, CSS, JS y PHP.
Conecta los formularios con los endpoints PHP usando fetch cuando sea apropiado.
Implementa:

- alta de registros;
- consulta/listado;
- edición;
- eliminación;
- mensajes al usuario;
- actualización de tablas sin recargar cuando sea razonable.

Respetar exactamente los nombres de campos de la base de datos y explica cualquier cambio necesario.

Producto esperado: Entregable: aplicación navegable con al menos un CRUD completo funcionando de extremo a extremo.

Reto adicional: Reto: solicitar que la IA dibuje textualmente el flujo: formulario → JS/fetch → PHP → MySQL → respuesta → interfaz.

10. Prompt 7 — Validar y mejorar la interfaz

Prompt para practicar:

```
Revisa la aplicación VitalCare como especialista en UI/UX y accesibilidad.
Analiza HTML, CSS y JavaScript proporcionados y detecta:
- problemas de responsividad;
- inconsistencias visuales;
- campos sin etiquetas;
- botones ambiguos;
- mensajes poco claros;
- problemas de navegación;
- código CSS duplicado;
- errores comunes de JavaScript.
Propón correcciones concretas sin alterar la funcionalidad del backend.
Después genera los archivos corregidos completos solo cuando sea necesario.
```

Producto esperado: Entregable: versión mejorada del frontend y lista de problemas encontrados.

Reto adicional: Reto: comprobar la aplicación en móvil, tablet y escritorio y registrar diferencias.

11. Prompt 8 — Depurar errores con IA

Prompt para practicar:

```
Actúa como ingeniero de debugging full-stack.
Voy a proporcionarte un error de VitalCare.
Analiza primero el mensaje de error, después el archivo implicado y finalmente el contexto.
1. Identifica la causa probable.
2. Explica por qué ocurre en lenguaje sencillo.
3. Propón la corrección mínima.
4. Entrega el código corregido.
5. Indica cómo comprobar que el error quedó solucionado.
No inventes archivos o código que no haya sido proporcionado.
```

Producto esperado: Entregable: ficha de depuración con error, causa, solución y prueba.

Reto adicional: Reto: introducir deliberadamente un error de conexión, un nombre de campo incorrecto y un error de JavaScript para practicar el diagnóstico.

12. Prompt 9 — Auditoría final y documentación

Prompt para practicar:

```
Actúa como revisor técnico de una aplicación web PHP/MySQL.
Audita VitalCare en:
- funcionalidad;
- estructura de archivos;
- consistencia entre frontend, PHP y BD;
- validación;
- seguridad básica;
- manejo de errores;
- usabilidad;
- responsividad;
- calidad del código;
- documentación.
```

No otorgues una calificación global. Presenta hallazgos clasificados como: CRÍTICO, IMPORTANTE o MEJORA.
Para cada hallazgo indica evidencia, riesgo y corrección propuesta.
Finalmente genera un checklist de pruebas de aceptación.

Producto esperado: Entregable: informe de auditoría y checklist final.

Reto adicional: Reto: convertir los hallazgos en tareas concretas y volver a ejecutar la auditoría después de corregirlos.

13. Proyecto integrador final

El estudiante debe entregar una versión funcional de VitalCare que cumpla, como mínimo, con:

- Menú principal y dashboard.
- CRUD completo de pacientes.
- CRUD completo de médicos y especialidades.
- Gestión de citas con relaciones entre paciente y médico.
- Base de datos MySQL creada mediante script.
- Conexión PHP mediante PDO y consultas preparadas.
- Validaciones de formularios y mensajes de operación.
- Diseño responsivo.
- Código organizado por carpetas y archivos.
- Datos de prueba suficientes para demostrar las operaciones.
- Documento README con instalación, configuración y uso.
- Registro de errores encontrados y soluciones aplicadas con ayuda de IA.

14. Checklist de pruebas

Prueba	Resultado esperado	OK
Crear paciente	El registro aparece en MySQL y en el listado	■
Editar paciente	Los cambios permanecen después de recargar	■
Eliminar paciente	El registro desaparece y se informa el resultado	■
Crear médico	Se guarda correctamente la especialidad	■
Crear cita	Paciente, médico, fecha y hora quedan relacionados	■
Validar campos obligatorios	No se envían datos incompletos	■
Probar diseño móvil	La navegación y formularios son utilizables	■
Probar error de BD	La aplicación muestra un mensaje controlado	■
Recargar y verificar persistencia	Los datos siguen almacenados	■

15. Evaluación sugerida

Criterio	Peso
Análisis de requerimientos y uso de prompts	15%
Diseño frontend y responsividad	15%
Modelo de datos y SQL	15%
CRUD PHP y conexión MySQL	25%
Integración y funcionamiento	15%
Depuración, pruebas y documentación	15%

16. Reflexión final del estudiante

- ¿Qué parte del proyecto fue más fácil de generar con IA y por qué?
- ¿En qué momento la IA produjo código que necesitó corrección?
- ¿Cómo comprobaste que el código generado realmente funcionaba?
- ¿Qué información del requerimiento fue indispensable para obtener mejores resultados?
- ¿Qué aprendiste sobre la diferencia entre pedir código y saber validar código?
- ¿Qué mejorarías del proyecto si tuvieras una segunda iteración?

Mensaje para el bootcamp: la IA es un asistente de programación, no un sustituto del razonamiento. El objetivo del taller es aprender a formular instrucciones, revisar resultados, ejecutar pruebas, detectar errores y tomar decisiones técnicas fundamentadas.