

Ingeniería en Sistemas de Información								
Cátedra: programac	Paradigmas ión III	y	lenguajes	de	Profesor: Mgter. Ing. Agustín Encina			
Alumno:					Fecha:			

Duración máxima: 2.30 horas

Instrucciones Generales:

- Este examen es interactivo y se compone de varias decisiones que tomarás a lo largo del camino.
- Siga las instrucciones cuidadosamente en cada punto de decisión.
- La puntuación total se basará en las decisiones tomadas y en la implementación de las tareas relacionadas con cada opción.
- No se permiten consultas en línea ni colaboración con otros **estudiantes ni con un transformador generativo preentrenado**.

📜 NARRATIVA DE LA AVENTURA

Has sido contratado por una startup tecnológica que necesita urgentemente un proyecto web funcional. Tu misión es demostrar tus habilidades como desarrollador full-stack navegando por diferentes desafíos. Cada decisión que tomes definirá tu camino y las tecnologías que dominarás.

¡Tu reputación como desarrollador está en juego! 🎯

X PARTE 1: DESAFÍOS TEÓRICOS (20 puntos)

ELECCIÓN DE MISIÓN INICIAL

Antes de comenzar tu proyecto, el equipo técnico necesita evaluar tus conocimientos fundamentales. Elige tu ruta de especialización:

Elige tu Proyecto (tildar la opción que vas a desarrollar):

- ☐ Ruta A: desarrolla el grupo A de preguntas.
- ☑ Ruta B: desarrolla el grupo B de preguntas.





RUTA A: "El Arquitecto Web" (Fundamentos y Estructura)

Desafío 1 - Arquitectura de la Web (5 puntos)

El CTO te pregunta durante la reunión inicial...

Dibuja y explica detalladamente la arquitectura Cliente-Servidor. Incluye:

- Componentes principales
- Flujo de comunicación
- Protocolos involucrados
- Ejemplo práctico con un caso real

Desafío 2 - Maestría en CSS (5 puntos)

El diseñador UX necesita claridad en la nomenclatura...

Explica la diferencia entre selectores de clase y selectores de ID en CSS:

- ¿Cuándo usar cada uno?
- Nivel de especificidad
- Proporciona 2 ejemplos prácticos de cada uno aplicados a una interfaz real

Desafío 3 - Fundamentos de JavaScript (5 puntos)

El líder técnico evalúa tu comprensión de JS...

Explica el concepto de variables en JavaScript:

- Propósito y utilidad
- Diferencias entre var, let y const
- Proporciona 3 ejemplos mostrando scope y hoisting

Desafío 4 - Introducción a PHP (5 puntos)

El backend developer senior te hace una pregunta clave...

¿Qué es PHP y cuál es su rol en el desarrollo web moderno?

- Características principales
- Diferencias con lenguajes frontend
- Ejemplo de código PHP integrado en HTML (procesamiento de formulario)





Arquitectura RUTA B: "El Innovador Técnico" (Evolución y Arquitectura)

Desafío 1 - Evolución del HTML (5 puntos)

El product manager pregunta sobre tecnologías modernas...

Explica las diferencias clave entre HTML y HTML5:

- Nuevas etiquetas semánticas
- Mejoras en accesibilidad y SEO
- ¿Cómo HTML5 revolucionó el desarrollo web?
- 1. Nuevas etiquetas semánticas: HTML5 introdujo etiquetas como <header>, <nav>, <article>, <section>, <aside> y <footer> que proporcionan un significado estructural a las diferentes partes de una página web, facilitando la comprensión tanto para los desarrolladores como para los navegadores y motores de búsqueda.
- 2. Mejoras en accesibilidad y SEO: Gracias a estas nuevas etiquetas semánticas, los lectores de pantalla pueden interpretar mejor el contenido, mejorando la accesibilidad. Además, al proporcionar una estructura más clara y significativa, los motores de búsqueda pueden indexar y clasificar el contenido de manera más efectiva, mejorando el SEO.
- 3. ¿Cómo HTML5 revolucionó el desarrollo web? HTML5 revolucionó el desarrollo web al integrar funcionalidades multimedia (audio, video), gráficos (Canvas, SVG) y capacidades offline (localStorage/SessionStorage) directamente en el navegador sin necesidad de plugins externos. También mejoró la interactividad con APIs, y estandarizó la semántica de los documentos, lo que llevó a sitios web más dinámicos y accesibles.

Desafío 2 - Arquitectura CSS Avanzada (5 puntos)

El tech lead quiere saber si conoces buenas prácticas...

Explica la diferencia entre arquitectura y metodología en CSS:

- Menciona al menos UNA arquitectura (ej: ITCSS, SMACSS, Atomic)
- Menciona al menos UNA metodología (ej: BEM, OOCSS, SUIT)
- ¿Por qué son importantes en proyectos grandes?

La arquitectura y la metodología en CSS son formas de organizar el código para que sea más făcil de mantener y escalar. La arquitectura es como el plano general de tu proyecto, la estructura grande de cómo vas a organizar tus estilos.

La metodología es un conjunto de reglas o pautas que sigues para escribir el CSS dentro de esa arquitectura.

Una arquitectura de CSS:



• ITCSS (Inverted Triangle CSS): Para entenderla primero debemos imaginarnos un triángulo invertido. En la parte de arriba (la más ancha) van los estilos más genéricos y que afectan a todo, como las bases (reset, tipografía). A medida que bajamos la pirámide, los estilos se vuelven más específicos y con mayor impacto visual (componentes, utilidades). Esto ayuda a que no haya conflictos entre estilos y a que el código sea más predecible.

Una metodología de CSS:

- **BEM (Block, Element, Modifier):** Es una forma de nombrar tus clases CSS que hace que el código sea súper claro y fácil de entender. Se basa en tres cosas:
 - o **Bloque:** Componente independiente (ej: header, button).
 - Elemento: Parte de un bloque que no tiene sentido fuera de él (header_logo, button icon).
 - Modificador: Cambia la apariencia o el comportamiento de un bloque o elemento (button--primary, button--disabled).
 La idea es que con solo leer el nombre de la clase, sepas qué es, a qué pertenece y si tiene alguna variación.

¿Por qué son importantes en proyectos grandes?

En proyectos grandes, donde hay muchos desarrolladores trabajando y el código crece un montón, si no hay una buena arquitectura y metodología, el CSS se convierte en un desastre. Es como intentar construir un edificio enorme sin planos. Termina siendo un código difícil de leer, con estilos que se pisan entre sí, y donde cambiar algo en un lugar puede romper otra cosa en otro lado.

Con una buena organización desde el inicio, evitamos todo eso, hacemos el código más robusto, y cada uno sabe dónde y cómo escribir su CSS sin generar conflictos.

Desafío 3 - JavaScript vs PHP (5 puntos)

El arquitecto de software evalúa tu visión técnica...

Compara y contrasta JavaScript y PHP:

- Diferencias fundamentales (ejecución, tipado, uso)
- 3 escenarios donde JavaScript es más apropiado
- 3 escenarios donde PHP es más apropiado
- Ejemplo de código de cada uno



Comparación de JavaScript y PHP

Característica	JavaScript	PHP		
Ejecución	Principalmente en el lado del cliente (navegador), también en el servidor con Node.js.	Exclusivamente en el lado del servidor.		
Tipado	Dinámico y débilmente tipado.	Dinámico y débilmente tipado.		
Uso principal	Interactividad frontend, desarrollo backend, aplicaciones móviles.	Desarrollo backend, gestión de bases de datos, sistemas de gestión de contenido.		

Escenarios donde JavaScript es más apropiado:

- 1. Interfaces de usuario interactivas: Para añadir dinamismo e interactividad a páginas web
- 2. Aplicaciones en tiempo real: Como chats o notificaciones push
- 3. Desarrollo de aplicaciones móviles: Con frameworks como React Native

Escenarios donde PHP es más apropiado:

- 1. Desarrollo de sitios web dinámicos con bases de datos: Ideal para la creación de sistemas de contenido, e-commerce y blogs
- 2. Aplicaciones empresariales y de gestión: Donde se requiere una lógica de negocio compleja y persistencia de datos en el servidor.

JavaScript (lado del cliente):

```
// Validar un campo de entrada
function mgl_validarEmail() {
 const emailEntrada = document.getElementById("email");
 const emailValue = emailEntrada.value;
  if (emailValue.includes("@") && emailValue.includes(".")) {
    console.log("Email válido");
  } else {
   console.log("Email inválido");
```





PHP (lado del servidor):

```
<?php
   if ($_SERVER["REQUEST_METHOD"] == "POST") {
       $nombre = htmlspecialchars($_POST['nombre']);
       echo "<h1>;Hola, " . $nombre . "! Bienvenido.</h1>";
10 <form method="post" action="index.php">
       <label for="nombre">Tu nombre:</label>
       <input type="text" id="nombre" name="nombre">
       <button type="submit">Enviar</putton>
   </form>
```

Desafío 4 - Conexión a Bases de Datos (5 puntos)

El DBA necesita confirmar tus conocimientos de persistencia...

Describe los conceptos fundamentales para conectar PHP con una Base de Datos:

- Métodos de conexión (MySQLi vs PDO)
- Pasos para establecer conexión
- Manejo de errores
- Ejemplo de código con consulta preparada

Métodos de Conexión (MySQLi vs. PDO)

Existen dos extensiones principales en PHP para interactuar con bases de datos MySQL:

- MySQLi (MySQL Improved): Es una extensión específica para MySQL. Ofrece una interfaz orientada a objetos y una interfaz procedural, siendo ideal cuando trabajas exclusivamente con bases de datos MySQL.
- PDO (PHP Data Objects): Es una capa de abstracción de bases de datos, significa que puedes usar la misma interfaz de código para conectarte a diferentes tipos de bases de datos (MySQL, PostgreSQL, SQLite, etc.), lo que lo hace más flexible y portable. PDO es generalmente la opción preferida por su flexibilidad, seguridad y soporte para múltiples bases de datos.

Pasos para Establecer Conexión

Independientemente del método, los pasos generales para establecer una conexión son:

- 1. **Definir Credenciales:** Necesitas la siguiente información:
 - Host: La dirección del servidor de la base de datos (comúnmente localhost).
 - **Usuario:** El nombre de usuario para acceder a la base de datos.



- Contraseña: La contraseña del usuario.
- Nombre de la Base de Datos: El nombre de la base de datos a la que quieres conectarte.
- 2. **Crear la Conexión:** Utiliza la extensión elegida (MySQLi o PDO) para crear una instancia de conexión, pasando las credenciales.
- Manejar Errores de Conexión: Es crucial verificar si la conexión fue exitosa. Si no lo fue, el script debe detenerse y mostrar un mensaje de error apropiado (sin exponer información sensible).

Manejo de Errores

Un manejo de errores robusto es vital para la seguridad y estabilidad de tu aplicación.

- MySQLi: Puedes usar mysqli_connect_errno() y mysqli_connect_error()
 en el enfoque procedural, o el método connect_error en el enfoque orientado a objetos para detectar y obtener información sobre errores de conexión.
- PDO: Utiliza bloques try-catch para capturar excepciones (PDOException) que puedan surgir durante la conexión o al ejecutar consultas. Esto permite un manejo de errores más estructurado y elegante.

Ejemplo de conexión a una base de datos

Esta función establece una conexión a una base de datos MySQL utilizando MySQLi.

Funcionamiento:

1. Conexión: Usa mysqli_connect() con los parámetros:

Servidor: localhost
Usuario: root
Contraseña: 2103
Base de datos: actividad22

2. **Validación:** Verifica si la conexión falló con **mysqli_connect_errno():** Si hay error: Muestra el mensaje de error y termina la ejecución



- 3. Si es exitosa: Configura el charset a UTF-8 para soportar caracteres especiales
- Retorno: Devuelve el objeto de conexión \$conexionDB para poder ejecutar consultas SQL

Función complementaria:

- desconectarDB(): Cierra la conexión a la base de datos usando mysqli_close() para liberar recursos
 - **PARTE 2: PROYECTO PRÁCTICO (80 puntos -** distribuidos en 4 niveles)
 - **M** Nivel 1 : ELECCIÓN DE PROYECTO BASE (20 puntos)

⚠ **REGLA CRÍTICA DE NOMENCLATURA:** Todos los archivos, carpetas, clases, funciones, tablas, etc., deben usar como prefijo tus iniciales.

Ejemplo: Si eres María González López (MGL):

- Carpeta: mgl_assets/
- CSS: mgl estilos.css
- Base de datos: mgl parcial plp3
- Tabla: mgl usuarios
- Función: function mgl_validar()
- Imagen: mgl_logo.png Clase CSS: .mgl-header

© M	IISIÓN PRI	NCIPAL - Elig	ge tu Proyect	o (tildar la ope	ción que vas d	a desarrollar):
------------	------------	---------------	---------------	------------------	----------------	-----------------

- Opción A: "MusicStream" Plataforma de Música Online
- ☐ Opción B: "FoodExpress" Sistema de Pedidos Online.
- ☑ Opción C: "QuizMaster" Plataforma de Trivia.



□ PROYECTO A: "MusicStream" - Plataforma de Música Online

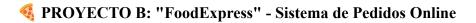
Una discográfica indie quiere su propia plataforma de streaming

Requisitos Funcionales:

- Catálogo de álbumes/canciones con reproductor básico (mínimo 8 items)
- Sistema de búsqueda por artista, género o álbum
- Formulario de suscripción con validación
- Listas de reproducción o favoritos (almacenadas en BD)
- Panel para agregar/editar canciones (CRUD)

- Mínimo 3 secciones distintas (header, galería, formulario)
- Diseño responsive (3 breakpoints)
- Navegación intuitiva y accesible
- Código comentado y estructura modular





Un restaurante local necesita digitalizar sus pedidos

Requisitos Funcionales:

- Menú de productos con categorías (mínimo 10 productos)
- Carrito de compras dinámico con subtotales
- Formulario de pedido que guarda en BD
- Sistema de filtrado por categoría
- Panel administrativo para gestionar productos

- Mínimo 3 secciones (menú, carrito, checkout)
- Responsive design con mobile-first
- Feedback visual en todas las interacciones
- Tiempo de carga optimizado





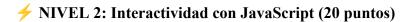
Una institución educativa quiere gamificar el aprendizaje

Requisitos Funcionales:

- Sistema de preguntas con múltiple opción (mínimo 15 preguntas)
- Validación de respuestas en tiempo real
- Sistema de puntuación y temporizador
- Tabla de mejores puntajes (stored en BD)
- Categorías temáticas con dificultad variable

- Mínimo 3 secciones (inicio, juego, resultados)
- Animaciones fluidas y feedback inmediato
- Diseño responsive
- Interfaz intuitiva sin instrucciones complejas





Documenta: Comenta la funcionalidad al inicio del archivo JS (3-5 líneas).

Para PROYECTO A (MusicStream):

Implementar: Reproductor Interactivo con Playlist

- Play/Pause/Skip con controles visuales
- Barra de progreso funcional
- Lista de reproducción dinámica
- Almacenar última canción reproducida

Para PROYECTO B (FoodExpress):

Implementar: Carrito de Compras Dinámico

- Agregar/eliminar productos sin recargar
- Cálculo automático de subtotales
- Validación de cantidades
- Mostrar contador de items en el carrito

Para PROYECTO C (QuizMaster):

Implementar: Lógica de Juego Completa

- Algoritmo de validación de respuestas
- Sistema de puntuación progresiva
- Temporizador con penalización
- Feedback visual inmediato (correcto/incorrecto)



NIVEL 3: Backend con PHP (20 puntos)

⚠ OBLIGATORIO: Conexión e interacción con Base de Datos MySQL

Documenta: Comenta la funcionalidad PHP implementada.

Implementación Requerida:

Para MusicStream:

- CRUD de canciones/álbumes
- Sistema de favoritos persistente
- Búsqueda con queries SQL

Para FoodExpress:

- CRUD de productos
- Registro de pedidos en BD
- Cálculo de totales en servidor

Para QuizMaster:

- Banco de preguntas desde BD
- Sistema de ranking persistente
- Registro de partidas jugadas

Base de Datos:

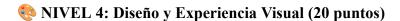
Crear con mínimo 2 tablas relacionadas:

- Claves primarias y foráneas
- Datos de prueba (mínimo 10 registros)

Exportar:

- [iniciales] estructura.sql
- [iniciales]_datos.sql





Documenta: Explica tus decisiones de diseño (paleta, tipografía, layout).

Requisitos Funcionales:

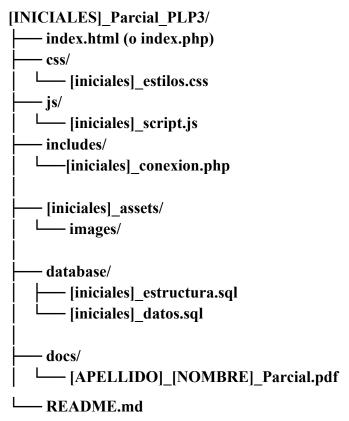
- Paleta de colores coherente (4-5 colores)
- Tipografía consistente (jerarquía clara)
- Responsive: 3 breakpoints mínimo
- Menú adaptativo (hamburguesa en mobile)

- Transiciones suaves (hover, focus)
- Loading states visibles
- Contraste adecuado (accesibilidad)
- Espaciado uniforme (grid/flexbox)





Estructura del Proyecto:



📤 Método de Entrega:

- 1. Archivo ZIP: [APELLIDO] [NOMBRE] PLP3.zip
- 2. Repositorio GIT con commits descriptivos
- 3. Subir a aula virtual dentro del tiempo del examen

Y EVALUACIÓN

Puntos Bonus (+10 máximo):

- Creatividad excepcional (+3)
- Seguridad (prepared statements) (+2)
- 6 Accesibilidad (ARIA, semántica) (+2)
- Features avanzadas (+3)



Penalizaciones:

- X Sin nomenclatura de prefijos: -5 pts
- Código sin comentarios: -3 pts

 No funciona: -10 pts
- X Plagio: 0 en el parcial

© CHECKLIST FINAL

- ✓ Nomenelatura con prefijo
- ☑ BD exportada
- ☑ JavaScript comentado
- ☑ PHP con BD funcionando
- ☑ README.md claro
- ✓ GIT con commits
- ☑ ZIP correctamente nombrado

🚀 ¡ÉXITO, DESARROLLADOR!

🖖 ¡Que la fuerza del código esté contigo! 🎃