

Ingeniería en Sistemas de Información	
Cátedra: Paradigmas y lenguajes de programación III	Profesor: Mgter. Ing. Agustín Encina
Alumno: Escallier Alejandro	Fecha: 29-10-25

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! 

#### ※ 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

**Cliente:**

Es el dispositivo del usuario (navegador web, app móvil, etc.) que solicita información o servicios.

Ejemplo: Google Chrome, Safari.

**Servidor:**

Equipo o software que recibe las peticiones del cliente, las procesa y devuelve una respuesta.

Ejemplo: Apache, Node.js.

**Base de datos:**

Almacena la información que el servidor necesita consultar o modificar.

Ejemplo: MySQL, MongoDB.

- Flujo de comunicación

El **cliente** envía una **petición HTTP** al servidor (por ejemplo, al escribir una URL).

El **servidor** recibe la solicitud, la procesa (puede consultar la base de datos).

El **servidor responde** con una página HTML o datos (JSON, XML).

El **cliente renderiza** la respuesta para el usuario.

- Protocolos involucrados

**HTTP / HTTPS:** protocolo base de comunicación entre cliente y servidor.

**TCP/IP:** garantiza el envío de paquetes entre ambos puntos.

**DNS:** traduce el nombre de dominio ([www.porejemplo.com](http://www.porejemplo.com)) a una dirección IP.

- Ejemplo práctico con un caso real

Cuando escribís [www.instagram.com](http://www.instagram.com):

1. Tu navegador (cliente) hace una solicitud HTTP a los servidores de Instagram.
2. El servidor procesa la petición, obtiene tus datos e imágenes de su base de datos.
3. Te devuelve un HTML con tu feed, imágenes y scripts.
4. Tu navegador interpreta el contenido y muestra la interfaz.

## 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?

Tipo	Se declara con	Se usa para	Especificidad
Clase	. (punto)	Agrupar y aplicar estilos a varios elementos	Baja
ID	# (numeral)	Identificar un único elemento específico	Alta

- Nivel de especificidad

**Clase:** cuando varios elementos comparten el mismo estilo.

Ejemplo: botones, títulos, tarjetas, etc.

**ID:** cuando solo un elemento de la página debe tener ese estilo.

Ejemplo: un encabezado principal, un contenedor único.

- Proporciona 2 ejemplos prácticos de cada uno aplicados a una interfaz real

### Clases

```
<button class="btn btn-primario">Enviar</button>
```

```
<button class="btn btn-secundario">Cancelar</button>
```

```
.btn {
```

```
padding: 10px 20px;  
border-radius: 5px;  
}
```

```
.btn-primario {  
background-color: blue;  
color: white;  
}  
  
.btn-secundario {  
background-color: gray;  
color: white;  
}
```

## IDs

```
<h1 id="titulo-principal">Bienvenido a Mi Sitio</h1>
```

```
<section id="hero">Contenido destacado</section>
```

```
#titulo-principal {  
font-size: 2.5em;  
text-align: center;  
}  
  
#hero {
```

```
background-image: url('banner.jpg');  
height: 400px;  
}
```

### 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

Una **variable** es un espacio en memoria donde se guarda un valor que puede cambiar o no durante la ejecución del programa.

**Propósito:** almacenar datos (números, texto, booleanos, objetos, etc.) para usarlos y manipularlos dinámicamente.

- Diferencias entre *var*, *let* y *const*

Tipo	Alcance (Scope)	Reasignable	Hoisting
<b>var</b>	Función o global	Sí	Sí (valor undefined)
<b>let</b>	Bloque	Sí	Sí, pero no usable antes de definirla
<b>const</b>	Bloque	No	Sí, pero no usable antes de definirla

- Proporciona 3 ejemplos mostrando scope y hoisting

```
// var (scope de función y hoisting)

console.log(a); // undefined

var a = 5;

function test() {

    var a = 10;

    console.log(a); // 10

}

test();

console.log(a); // 5
```

```
// let (scope de bloque)

{
    let x = 20;

    console.log(x); // 20

}

// console.log(x); // ✗ Error, no accesible fuera del bloque
```

```
// const (constante, no reasignable)

const PI = 3.1416;

// PI = 3; ✗ Error

console.log(PI);
```

#### 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?

**PHP (Hypertext Preprocessor)** es un lenguaje de programación del lado del **servidor** usado para crear páginas web dinámicas.

- Procesa datos antes de enviar la respuesta al cliente.
  - Interactúa con bases de datos (MySQL, PostgreSQL).
  - Genera HTML dinámico.
  - Maneja sesiones, autenticación y formularios.
- 
- Características principales
    - Lenguaje interpretado (no necesita compilación).
    - Código abierto y multiplataforma.

- Integración sencilla con HTML.
- Amplio soporte y frameworks (Laravel, Symfony, etc.).
- Diferencias con lenguajes frontend

PHP	HTML/CSS/JS
Se ejecuta en el <b>servidor</b>	Se ejecutan en el <b>navegador</b>
Genera contenido dinámico	Muestran contenido al usuario
Puede acceder a bases de datos	No pueden hacerlo directamente

- Ejemplo de código PHP integrado en HTML (procesamiento de formulario)

```
<!-- archivo: formulario.html -->

<form action="procesar.php" method="POST">

<input type="text" name="nombre" placeholder="Tu nombre">

<input type="email" name="correo" placeholder="Tu correo">

<button type="submit">Enviar</button>

</form>

<!-- archivo: procesar.php -->

<?php

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $nombre = $_POST['nombre'];

    $correo = $_POST['correo'];

    echo "Gracias $nombre, te contactaremos en $correo.";

}

?>
```

## 🔗 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?

### 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?

### 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

### 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

⌚ PARTE 2: PROYECTO PRÁCTICO (80 puntos - *distribuidos en 4 niveles*)

☛ 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

⌚ MISIÓN PRINCIPAL - Elige tu Proyecto (*tildar la opción que vas a desarrollar*):

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

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web



### 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) **Requisitos No**

##### Funcionales:

- Mínimo 3 secciones distintas (header, galería, formulario)
- Diseño responsive (3 breakpoints)

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web



### PROYECTO

- Navegación intuitiva y accesible
- Código comentado y estructura modular



### B: "FoodExpress" - Sistema de Pedidos Online

*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 **Requisitos No**

#### Funcionales:

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web



### PROYECTO

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



### C: "QuizMaster" - Plataforma de Trivia

*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)

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web



### PROYECTO

- Categorías temáticas con dificultad variable **Requisitos No Funcionales:**

#### Funcionales:

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

## **Examen Interactivo de PLP III: La Aventura del Desarrollador Web**

### **NIVEL 2: Interactividad con JavaScript (20 puntos)**

 **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:

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web

### 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:**
- Registro de partidas jugadas 

### Crear con mínimo 2 tablas relacionadas:

- Claves primarias y foráneas
  - Datos de prueba (mínimo 10 registros)
- Exportar:**
- [\[iniciales\]\\_estructura.sql](#)
  - [\[iniciales\]\\_datos.sql](#)

## NIVEL 4: Diseño y Experiencia Visual (20 puntos)

 **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)

### Requisitos No Funcionales:

- Transiciones suaves (hover, focus)
- Loading states visibles
- Contraste adecuado (accesibilidad)

**Examen Interactivo de PLP III: La Aventura del Desarrollador Web** 

- Espaciado uniforme (grid/flexbox)

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web

### ENTREGA FINAL

#### Estructura del Proyecto:

```
[INICIALES]_Parcial_PLP3/
├── index.html (o index.php)
├── css/
│   └── [iniciales]_estilos.css
├── js/
│   └── [iniciales]_script.js
├── includes/
│   └── [iniciales]_conexion.php
├── [iniciales]_assets/
│   └── images/
├── database/
│   ├── [iniciales]_estructura.sql
│   └── [iniciales]_datos.sql
└── docs/
    └── [APELLIDO]_[NOMBRE]_Parcial.pdf
└── README.md
```

#### 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

### EVALUACIÓN

#### Puntos Bonus (+10 máximo):

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

## Examen Interactivo de PLP III: La Aventura del Desarrollador Web 🎮

- 📱 Features avanzadas (+3)

### Penalizaciones:

- ✗ Sin nomenclatura de prefijos: -5 pts
- ✗ Código sin comentarios: -3 pts
- ✗ No funciona: -10 pts
- ✗ Plagio: 0 en el parcial

### 🔗 CHECKLIST FINAL

- Teoría completa
- Nomenclatura con prefijo
- Proyecto funcional
- BD exportada
- CSS responsive
- JavaScript comentado
- PHP con BD funcionando
- README.md claro
- GIT con commits
- ZIP correctamente nombrado

🚀 ¡ÉXITO, DESARROLLADOR!

👋 ¡Que la fuerza del código esté contigo! 😊