

EXAMEN Desarrollo Web Entorno Cliente — 2º DAW

Aplicación Web: Gestor de Parking — ParkingPereMaria

Duración: 2 horas

Archivos obligatorios:

- **index.html**
- **styles.css**
- **parkingPanel.js**
- **imagen.jpg**

Contexto del proyecto

La empresa ParkingPereMaria necesita una aplicación web funcional para gestionar su parking interno.

Han contratado a vuestro equipo de desarrollo para implementar la primera versión del sistema.

El sistema debe permitir:

- Registrar vehículos que podrán acceder al parking.
- Controlar entradas y salidas.
- Consultar plazas libres.
- Administrar el operario que está gestionando el sistema.

Se os proporciona la estructura base del proyecto con los tres archivos obligatorios:

index.html (estructura de la web)

styles.css (estilos base obligatorios, con imagen de fondo)

parkingPanel.js (código JavaScript que debéis completar)

imagen.jpg

Objetivo.

A partir de los cuatro archivos que se os han entregado, debéis completar la funcionalidad del Gestor de Parking, siguiendo las especificaciones oficiales de la empresa.

Toda la lógica debe quedar implementada en parkingPanel.js, utilizando:

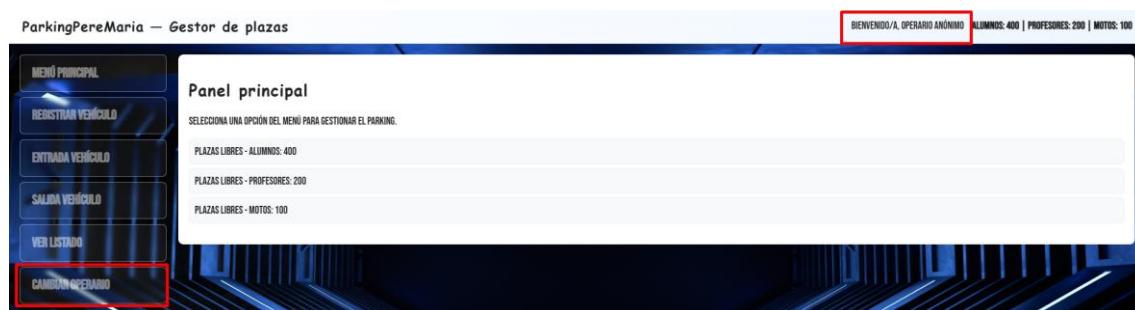
- DOM (manipulación y eventos)
- localStorage
- Objetos JavaScript

- Fechas y tiempos
- Arrays y operaciones con ellos

REQUISITOS FUNCIONALES

1) Gestión de operario

- El sistema debe solicitar el nombre del operario al iniciarse.
- El nombre debe guardarse en localStorage.
- Debe mostrarse en la parte superior:
“Bienvenido/a, NOMBRE_DEL_OPERARIO”
- Desde el menú debe poder cambiarse el operario.



2) Registro de vehículos. (1.5 puntos)

El formulario de Registrar vehículo debe almacenar en localStorage:

- Propietario: Alumno / Profesor
- Tipo: Coche / Moto
- Matrícula
- Curso (siempre 2025)
- Operario que registró el vehículo
- Tiempo total acumulado en minutos
- Historial de entradas y salidas
- Estado (si está dentro del parking o no)

Reglas:

- No puede haber matrículas duplicadas.
- Al registrar un vehículo, debe volver a “Menú principal”.

ParkingPereMaria — Gestor de plazas

Bienvenido/a, OPERARIO ANÓNIMO ALUMNOS: 400 | PROFESORES: 200 | MOTOS: 100

REGISTRAR VEHÍCULO

Registrar vehículo

PROPIETARIO:

ALUMNO

TIPO:

COCHE

MATRÍCULA:

CURSO:

2025

REGISTRAR VEHÍCULO **VOLVER**

3) Entrada de vehículo. (1.5 puntos)

El sistema debe permitir registrar la entrada de un vehículo solo indicando la matrícula.

Al entrar se debe:

- Verificar que el vehículo existe.
- Verificar que hay plazas disponibles según:
 - 400 plazas alumnos (coches)
 - 200 plazas profesores (coches)
 - 100 plazas motos
- Registrar fecha/hora de entrada.
- Marcar vehículo como "en parking".

4) Salida de vehículo. (1.5 puntos)

Al registrar la salida:

- Verificar que el vehículo está dentro del parking.
- Calcular los minutos desde su última entrada.
- Sumar ese tiempo al total del vehículo.
- Registrar fecha/hora de salida.
- Liberar plaza correspondiente.

5) Listado de vehículos. (1 punto)

Debe mostrarse una tabla con:

- Matrícula
- Propietario
- Tipo
- Si está dentro del parking

- Tiempo total acumulado
- Última entrada
- Última salida
- Operario que lo registró

Además:

Mostrar total de vehículos registrados

Mostrar plazas libres actuales

Colorear correctamente según propietario o tipo

6) Contadores de plazas libres. (0.5 puntos)

En todo momento deben mostrarse actualizadas:

- Plazas libres alumnos
- Plazas libres profesores
- Plazas libres motos

Deben recalcularse tras:

- Registrar vehículo
- Registrar entrada
- Registrar salida

REQUISITOS DE ESTILO

El fichero styles.css ya incorpora:

- La imagen imagen.jpg como fondo obligatorio.
- Transparencias y diseño claro.
- Estética similar al proyecto final real.

Los alumnos pueden añadir más reglas, pero no eliminar las ya existentes.

ENTREGA

El alumno debe entregar un ZIP con:

- index.html
- styles.css
- parkingPanel.js
- imagen.jpg

Todo el código debe funcionar sin errores en un navegador estándar.

Criterios de evaluación (100 puntos)

- **Funcionalidad completa JS — 60 pts**
 - **Registro vehículo: 15**
 - **Entrada: 15**
 - **Salida: 15**
 - **Listado dinámico: 10**
 - **Contadores actualizados: 5**
- **Uso correcto de DOM y eventos — 10 pts**
- **Uso de localStorage — 10 pts**
- **Validaciones y mensajes — 10 pts**
- **Diseño y usabilidad (CSS) — 10 pts**