Aplicaciones Web - Curso 2023/2024

Convocatoria ordinaria (16/01/2023) - Duración: **3 horas**
Puntuación máxima del examen: **10 puntos**
Puntuación mínima en el examen para poder aprobar la asignatura: **4 puntos**

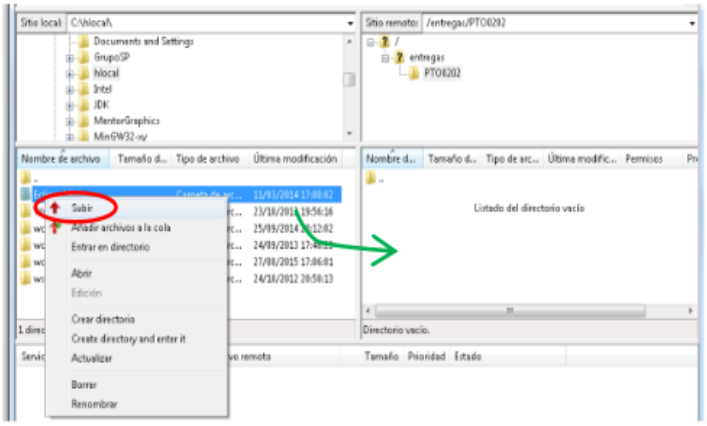
INSTRUCCIONES

1. Descarga el fichero "recursos.zip" y cópialo en un directorio donde tengas privilegios de escritura. En el fichero encontrarás la carpeta "**servidor**" que contiene todas las carpetas necesarias para la comenzar a realizar el examen, la carpeta **node-modules** y los ficheros **package.json** y **package-lock.json**. También verás una carpeta llamada "**cliente**" con los archivos que necesitarás para probar tus ejercicios desde el cliente.
2. Abre **Visual Studio Code** y comprueba que la carpeta donde realizarás el examen está correctamente formada.
3. **Los ejercicios que se vaya a entregar debe funcionar, cumpliendo los objetivos requeridos en menor o mayor medida, pero no se admitirán errores de recursos no encontrados (páginas, imágenes, etc..). Si un ejercicio entregado lanza cualquier error de este tipo la calificación será de 0 en el ejercicio correspondiente.**
4. El diseño es importante, intenta que los resultados se muestren de forma clara y ordenada.
5. En la corrección se valorará el funcionamiento, la claridad del código y el diseño de las soluciones.

INSTRUCCIONES DE ENTREGA

Para entregar la solución del examen, crea un fichero **NombreApellidos.zip** (**usar sólo ZIP**). En él debes incluir todos los ejercicios bien en carpetas separadas o todas juntas siempre exceptuando la carpeta **node-modules** y los ficheros **package.json** y **package-lock.json**. Haz doble clic en el icono del escritorio denominado: **"EXAMENES en LABs entregas"**, y dentro de la ventana que aparece, haz doble clic en "**ALUMNOS entrega de prácticas y exámenes"**. Se abre otra ventana en la que debes seleccionar el archivo **zip** en el panel inferior izquierdo y arrastrarlo al panel inferior derecho (o utiliza el botón derecho del ratón y la opción **Subir**), como se indica en la figura.

Antes de abandonar el laboratorio debes pasar por el puesto del profesor para asegurarte de que lo que se ve en el puesto del profesor es lo que has entregado (**comprobando el tamaño del archivo**) y firmar en la hoja de entregas (usar tu bolígrafo para evitar el contacto).



Pregunta 1 [3 puntos]

Desarrolla un configurador de estilos web que permita a los usuarios personalizar diferentes aspectos de una página. El configurador debe ser implementado utilizando tecnologías HTML, JavaScript, CSS y el Framework Bootstrap.

La página web contará con un botón para **abrir el configurador** que consistirá en una ventana emergente donde se visualizarán las diferentes opciones.

El configurador debe incluir un botón para **cerrar la ventana** así como los controles que se estimen oportunos para personalizar al menos los siguientes aspectos:

- Tamaño de fuente para títulos y textos de los div de la página.
- Color de títulos y textos de los div de la página.
- Color de fondo de secciones.
- Color de fondo de sección 1 y sección 2.
- Color de fondo para la cabecera y el footer.
- Color para los enlaces, sin visitar y visitados.
- Subrayado de enlaces (con una casilla de verificación).
- Color de fondo de la barra de navegación.
- Color de los enlaces de la barra de navegación
- Subrayado de los enlaces de la barra de navegación.

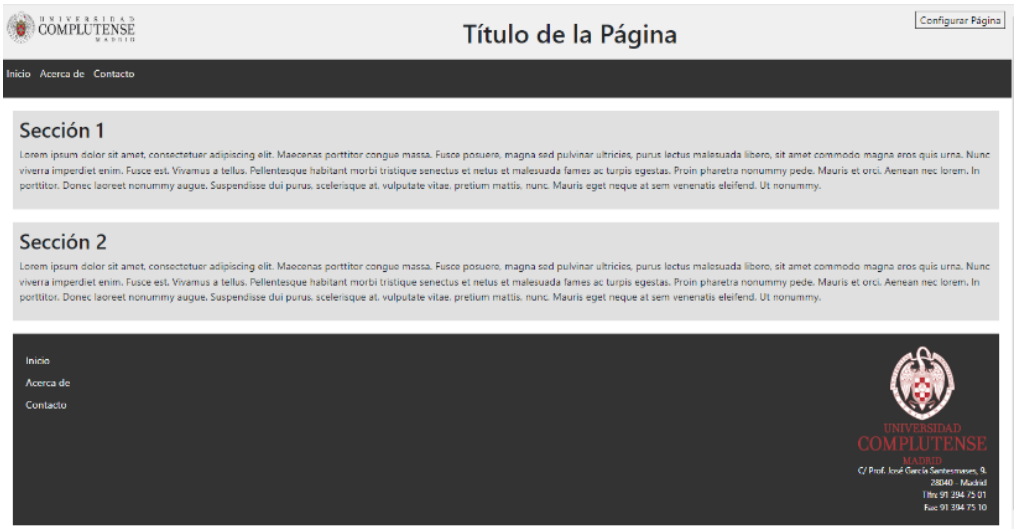
El configurador de estilos tendrá un diseño responsivo que se adapte a diferentes tamaños de pantalla. Utiliza para ello las “herramientas para desarrolladores” del navegador.

Utiliza el Framework Bootstrap para mejorar la apariencia y disposición de los elementos en el configurador. Puedes utilizar Word para generar el texto ‘Lorem ipsum’ mediante la instrucción **=lorem()**. Acompaña tu código con comentarios para mayor claridad.

Crea una carpeta llamada ejercicio1 para guardar los siguientes ficheros:

- Index.html
- Estilos.css (si lo ves necesario)
- Script.js (si lo ves necesario)
- Carpeta **images** para las imágenes (si lo ves necesario)

Se aporta una captura para tener una idea **aproximada** de lo que se pide en este ejercicio.



Pregunta 2 [4 puntos]

El departamento de personal de la empresa nos hace llegar un archivo JSON con los nombres de los usuarios externos que tendremos que gestionar. El listado es el siguiente:


```
{
  "usuarios_externos": {
    "usuario": [
      {
        "id": "Felipe",
        "nombre": "Felipe Lotas",
        "telefonos": {
          "casa": "952124567"
        }
      },
      {
        "id": "Alberto",
        "nombre": "Alberto Cadiscos",
        "telefonos": {
          "casa": "912382722",
          "movil": "678234567"
        }
      },
      {
        "id": "Borja",
        "nombre": "Borja Món de York",
        "telefonos": {
          "movil": "678234567"
        }
      },
      {
        "id": "Aitor",
        "nombre": "Aitor Tilla",
        "telefonos": {
          "casa": "912382722",
          "movil": "678234567",
          "oficina": "927121212"
        }
      },
      {
        "id": "Diego",
        "nombre": "Diego Norrea",
        "telefonos": {
          "oficina": "927121213"
        }
      },
      {
        "id": "Jesús",
        "nombre": "Jesús Piros de España",
        "telefonos": {
          "casa": "912382722",
          "movil": "678234567"
        }
      },
      {
        "id": "Rubén",
        "nombre": "Rubén Tosidad",
        "telefonos": {
          "casa": "952124567"
        }
      }
    ]
  }
}
```

Diseña un servicio web de tipo REST para que se puedan realizar las siguientes operaciones:

- **Obtener todos los objetos**: Devuelve toda la agenda en formato JSON.

- **Obtener un objeto a partir de su índice**. Realizar esta operación con URL paramétrica:
  Ejemplo de la petición: http://localhost:3000/contactos/2

- **Añadir un objeto**: El nuevo contacto vendrá en formato JSON:
  Módulo de express a utilizar: app.use(express.json());

- **Eliminar un objeto**. Utilizar el método splice de un array. Realizar esta operación utilizando el cuerpo de la petición:
  Ejemplo de petición: http://localhost:3000/contactos

- **Actualizar un objeto.** Actualiza los datos de un objeto identificado mediante su índice.

Desarrollar todas las operaciones mencionadas utilizando un route llamado crudUsuarios donde se gestionen todas las operaciones CRUD.

El servicio web deberá detectar los errores en la entrada de las peticiones (p.e el índice del elemento a buscar) por lo que deberá contar con los middleware necesarios que detecten errores 404 y 500.

El front-end del sitio consistirá en una página con todos los controles para probar todas las operaciones.

Puede que quieras utilizar funciones como **splice** o **filter** para el borrado de elementos en una matriz.

En este ejercicio es obligatoria la gestión de los errores 404 y 500 mediante las implementación de middlewares. Se valorará muy positivamente la refactorización del código así como la visualización de os resultados.

Crea una carpeta ejercicio 2 con toda la estructura del servidor **exceptuando la carpeta node-modules.**

Pregunta 3 [3 puntos]

En el desarrollo de la nueva Intranet de la Empresa, se te ha encomendado la parte del registro de usuarios. Para ello se espera que desarrolles una página de registro accesible desde la página principal de la Intranet de la empresa (interpretarlo como la definición del contexto de trabajo).

La página de registro sería accesible en la URL **http://localhost:3000/usuarios**. En esta primera página se mostrarán todos los usuarios registrados, sólo se mostrará la foto y el nombre. La foto será un fichero con extensión **.png** que se guardará en la parte pública de la aplicación, directorio **uploads**.

Los registros de usuarios se almacenarán en una variable llamada **usuarios** de tipo array que contendrá los objetos con los datos de los usuarios que se vayan registrando. La página del registro será accesible en **http://localhost:3000/registro** y solicitará los siguientes campos:

- Nombre (obligatorio)
- Apellidos (obligatorio)
- Correo electrónico (con dominio ucm.es)
- Contraseña (obligatoria - no hace falta ningún tipo de cifrado)
- Foto (opcional, si no se introduce ningún fichero se cogerá el fichero por defecto llamado noUser.png)

En el momento de hacer clic en el botón **Registro** se procederá a añadir ese nuevo objeto al array usuarios y se mostrará la página **http://localhost:3000/usuarios** de usuarios con la relación actualizada de usuarios registrados.

Ejemplo de la variable usuarios:

```
var usuarios =  [
    {nombre: "Aitor",
     apellidos: "Tilla Patata",
     correo: aitor@ucm.es,
     pass: "ATP01",
     foto: "aitorTilla.png"}
]
```

Esta página de registro tendrá un diseño responsivo que se adapte a diferentes tamaños de pantalla. Utiliza para ello las “herramientas para desarrolladores” del navegador.

En este ejercicio es obligatoria la gestión de los errores 404 y 500 mediante las implementación de middlewares. Se valorará muy positivamente la refactorización del código así como la visualización de os resultados.