Boletín de Ejercicios - Sprint 1

Descripción del ejercicio

Para asentar los conocimientos mostrados en la parte teórica, se tendrá que proceder a resolver los siguientes ejercicios/problemas en el orden correcto. Para ello, primero pasamos a explicar los criterios que debemos seguir para la resolución de los ejercicios/problemas.

Fases de la resolución de problemas

1. **Análisis del problema:** Se debe indicar en el directorio específico de la asignatura el problema que se va a resolver de una forma adecuada, es decir, no debe contener ambigüedades, debe ser simple y autocontenido.

2. **Diseño de la propuesta de solución del problema:** Como todo aquel problema que se quiere resolver, es necesario realizar el diseño de la o las soluciones que se procederá a implementar en el siguiente paso. Para esto nos debemos ayudar de las herramientas para realizar esquemas gráficos (UML, Diagramas de flujos, etc…), además de incluir la conversación con cualquier herramienta que potencie la productividad para resolver el boletín de ejercicios (para cada ejercicio).

3. **Implementación del diseño propuesto:** En este punto ya se procederá a implementar todo el diseño establecido en el punto anterior.

4. **Pruebas de la resolución del problema:** Es indispensable el realizar pruebas para verificar la integridad y correcto funcionamiento de la implementación realizada, para ello simplemente compararemos si el comportamiento esperado del análisis del problema se ha implementado de forma adecuada. Vídeos en formato GIF que se puedan visualizar desde github web.

Ejercicio 1: Cambio de Color con Botón

- Crear una página web que contenga un botón etiquetado "Cambiar color".

- Al hacer clic en el botón, el color de fondo de la página debe cambiar a un color aleatorio.

- Pista: Utiliza `Math.random()` para generar valores RGB aleatorios.

ENTREGAR: ejercicio1.html y ejercicio1.js

Prueba: Vídeo GIF en Readme.md que muestre que al pulsar sobre el botón varias veces, este hace que cambie el color del fondo únicamente.

Ejercicio 2: Calculadora de Área

- Diseña una página web con dos campos de entrada (input) para introducir el ancho y el alto de un rectángulo.

- Agrega un botón etiquetado "Calcular Área".

- Al hacer clic en el botón, calcula el área del rectángulo y muestra el resultado en un elemento `<p>` en la página.

- Pista: Área del rectángulo = ancho x alto.

ENTREGAR: ejercicio2.html y ejercicio2.js

Prueba: Vídeo GIF en Readme.md que muestre que al rellenar ambos campos con un 2, este muestre un 4 y que al rellenarlo con un 2 y un 9, este muestre un 18.

Ejercicio 3: Listado Dinámico

- Crea una página con un campo de entrada y un botón etiquetado "Añadir a la lista".

- También debes tener una lista vacía (`<ul>` o `<ol>`).

- Cuando el usuario escribe algo en el campo de entrada y hace clic en el botón, entonces el contenido del campo debe agregarse como un nuevo ítem (`<li>`) a la lista.

- Pista: Utiliza el método `.createElement()` y `.appendChild()` del DOM.

ENTREGAR: ejercicio3.html y ejercicio3.js

Prueba: Vídeo GIF en Readme.md que muestre como se añaden 3 elementos a la lista.

Ejercicio 4: Hover y Estilo Dinámico

- Diseña una página con varios elementos div, cada uno con un texto diferente.

- Al pasar el ratón sobre un div, cambia su color de fondo a azul y el texto a blanco.

- Al mover el ratón fuera del div, restaura sus estilos originales.

- Pista: Considera usar eventos como "mouseover" y "mouseout".

ENTREGAR: ejercicio4.html y ejercicio4.js

Prueba: Vídeo GIF en Readme.md que muestre como al pasar por encima cambia a fondo azul y texto blanco y cuando se sale restaura sus valores.

Ejercicio 5: Detección de Clics y Generación de XPath

Descripción:

Desarrolla una página web que, al hacer clic en cualquier elemento, muestre el XPath único de ese elemento en un cuadro de alerta o en una sección dedicada de la página.

Especificaciones:

1. Detección de Clics:

- Añade un evento de escucha a todo el documento (`document`) para detectar cualquier clic realizado.

- Al detectar un clic, determina el elemento exacto que fue clickeado usando el objeto `event.target`.

2. Generación de XPath:

- Una vez identificado el elemento, genera su XPath.

- Muestra el XPath generado en un cuadro de alerta o en una sección específica de la página.

Se adjunta el HTML para el ejercicio 5 llamado 'Ejercicio5.html' (no modificable) Hay que hacer click sobre cada uno de los botones y tiene que venir reflejado que se ha hecho click sobre cada uno con una alerta que indique el XPATH relativo del que se ha realizado el click.

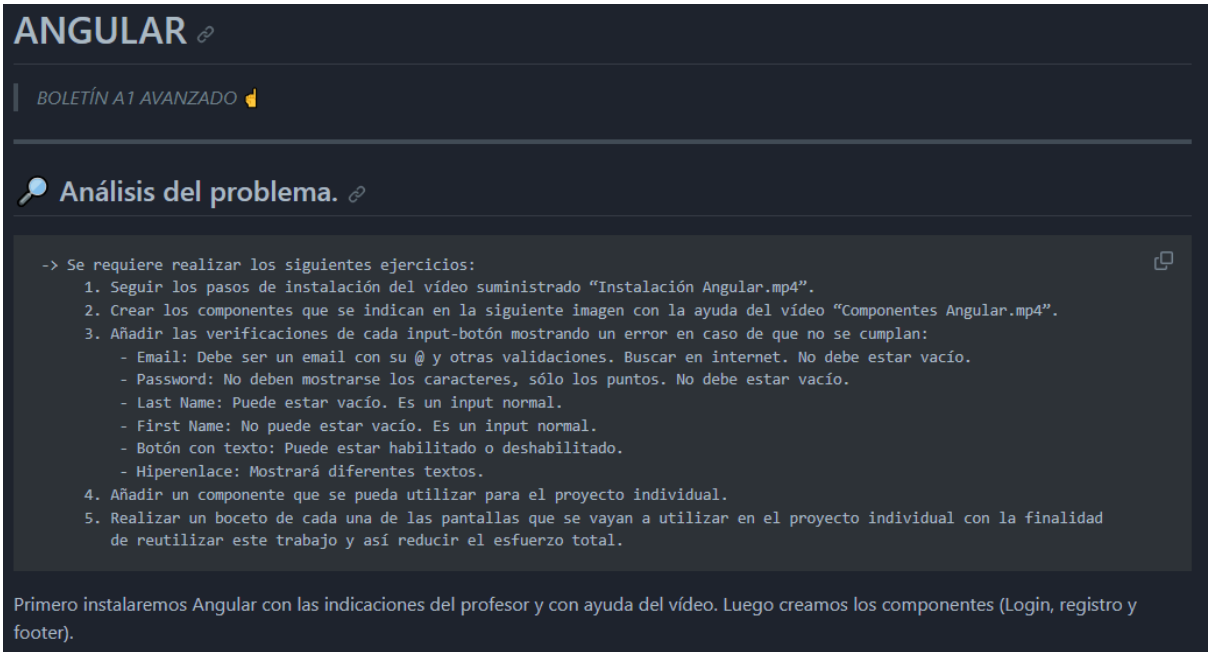
ENTREGAR: ejercicio5.js

Criterios en el formato de entrega

Todos estos ejercicios se deberán entregar en el formato establecido en clase o tablón de classroom, respetando las horas de entrega de cada uno de ellos indicados en la tarea de classroom.

¿Qué y cómo se entrega?

- Hay que realizar cada apartado de ejercicios en HTML diferentes y subirlo al repositorio a la carpeta SPRINT 1
- Hay que realizar un vídeo en formato .gif para cada ejercicio en el que se interactúe de manera dinámica con la web y adjuntarlo en el README.md del repositorio GIT en la carpeta del sprint correspondiente.
- Hay que realizar una captura de pantalla de aquellos ejercicios que sean estáticos y adjuntarlos en el README.md del repositorio GIT en la carpeta del sprint correspondiente.
- La estructura de carpetas del repositorio de GitHub debe seguir las normas establecidas de “NombreApellido-EC” -> “T1” -> “SPRINTX”, siendo X en la palabra SPRINTX el correspondiente al sprint actual.
- Los GIF se tiene que ver dentro de la previsualización del README.md.
- Para los boletines de sólo HTML y Javascript NO se podrán desplegar, sólo podrán consultarse mediante el .html abriendo el archivo con el navegador.
- Ejemplo de un README a continuación:



ANGULAR

BOLETÍN A1 AVANZADO ☝

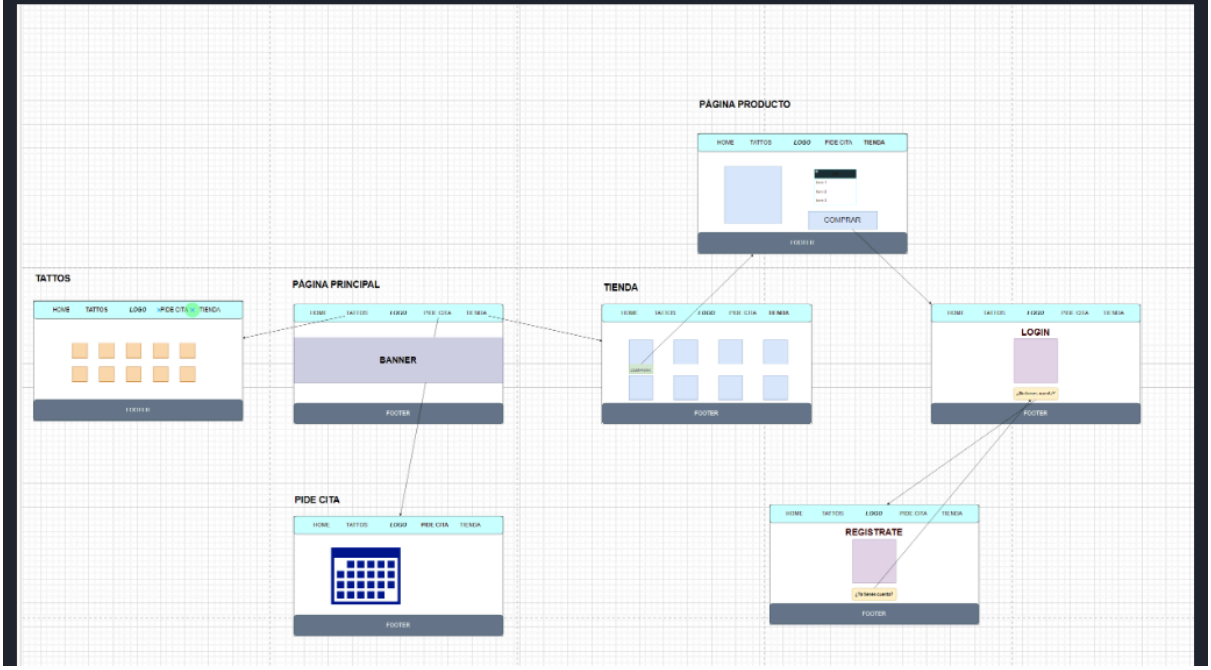
🔎 Análisis del problema.

```
-> Se requiere realizar los siguientes ejercicios:
    1. Seguir los pasos de instalación del vídeo suministrado “Instalación Angular.mp4”.
    2. Crear los componentes que se indican en la siguiente imagen con la ayuda del vídeo “Componentes Angular.mp4”.
    3. Añadir las verificaciones de cada input-botón mostrando un error en caso de que no se cumplan:
        - Email: Debe ser un email con su @ y otras validaciones. Buscar en internet. No debe estar vacío.
        - Password: No deben mostrarse los caracteres, sólo los puntos. No debe estar vacío.
        - Last Name: Puede estar vacío. Es un input normal.
        - First Name: No puede estar vacío. Es un input normal.
        - Botón con texto: Puede estar habilitado o deshabilitado.
        - Hiperenlace: Mostrará diferentes textos.
    4. Añadir un componente que se pueda utilizar para el proyecto individual.
    5. Realizar un boceto de cada una de las pantallas que se vayan a utilizar en el proyecto individual con la finalidad
       de reutilizar este trabajo y así reducir el esfuerzo total.
```

Primero instalaremos Angular con las indicaciones del profesor y con ayuda del vídeo. Luego creamos los componentes (Login, registro y footer).

📝 Implementación de la solución.

En este apartado vamos a ponernos a implementar todos los apartados anteriores, vamos a hacer el ejercicio completo y los gifs de cada prueba.



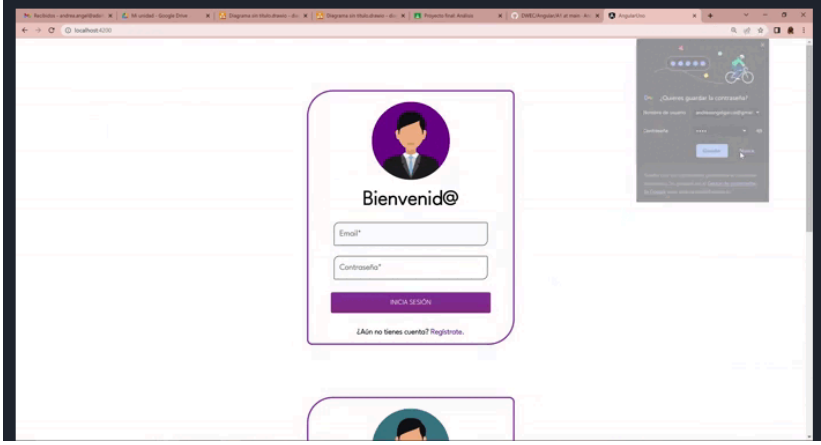
💡 Pruebas.

-> Plan de pruebas:

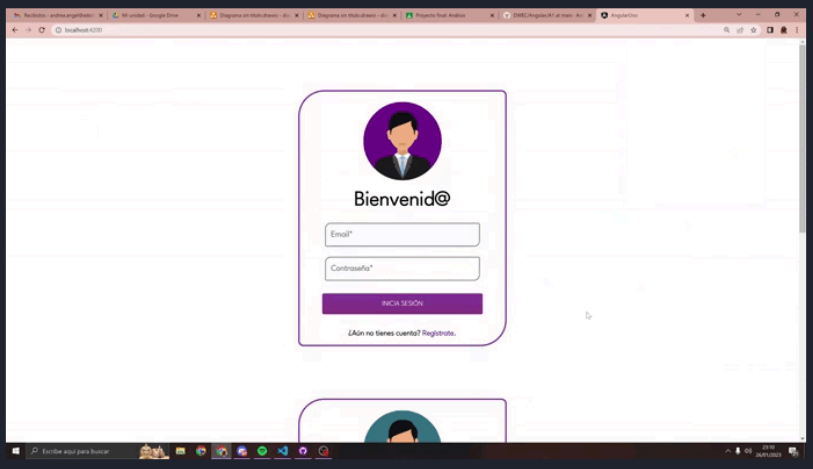
TestID	TestName	Description	StepNumber	StepAction	StepCondition
1	CASO OK - LOGIN	Entramos en la página, escribimos el correo y la contraseña correctamente y nos deja darle al botón.	1	Abrimos la URL	Aparece una página web con login, registro y footer
			2	Escribimos el correo correctamente	Se muestra el correo
			3	Escribimos la contraseña	Aparece la contraseña oculta y podemos darle al botón
2	CASO KO1 - LOGIN	Entramos en la página, escribimos el correo y no ponemos ninguna contraseña.	1	Abrimos la URL	Aparece una página web con su logo, buscador y una tabla
			2	Escribimos el correo correctamente	Se muestra la tabla vacia y el mensaje de cue no hay ningún pokémon
			3	No escribimos ninguna contraseña	No nos deja darle al botón
3	CASO KO2 - LOGIN	Entramos en la página, escribimos mal el correo y no nos deja darle al botón.	1	Abrimos la URL	Aparece una página web con su logo, buscador y una tabla
			2	Escribimos mal el correo	Se muestra la tabla vacia y el mensaje de cue no hay ningún pokémon
			3	No escribimos ninguna contraseña	No nos deja darle al botón
4	CASO OK - REGISTRO	Entramos en la página, escribimos los datos correctamente y nos deja darle al botón.	1	Abrimos la URL	Aparece una página web con login, registro y footer
			2	Escribimos el correo correctamente	Se muestra el correo
			3	Escribimos la contraseña	Aparece la contraseña oculta y podemos darle al botón
5	CASO KO- REGISTRO	Entramos en la página, escribimos los datos incorrectamente y no nos deja darle al botón.	1	Abrimos la URL	Aparece una página web con login, registro y footer
			2	Escribimos los datos incorrectamente	No nos deja darle click
			3	No nos deja darle al boton	Aparece la contraseña oculta y no podemos darle al botón

--

-> LOGIN:

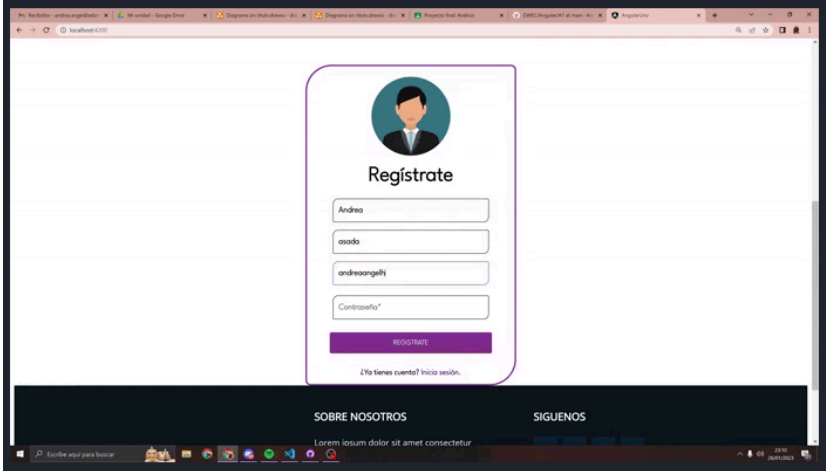


-> LOGIN:




```
CASO OK:
Entramos en la página,
ponemos el correo correctamente
e introducimos la contraseña.
```

-> REGISTRO:




```
CASO OK:
Entramos en la página,
ponemos los datos correctamente
e introducimos la contraseña.
```

Para los que quieran poner vídeos de la ejecución:

Convertidor Vídeo a GIF

- Hacer click a *Elegir Archivo* y elige el vídeo.
- Una vez elegido, click en *Upload video!*
- Seleccionar *Convert*.
- Puedes recortar el GIF para hacerlo más corto.

Video to animated GIF converter (ezgif.com)

Criterios de evaluación

Ejercicio 1	No se cumple con alguno de los criterios del formato de entrega o no se realizan las pruebas o no se realiza el gif que verifique el correcto funcionamiento del ejercicio. **0 puntos**	Se cumple con lo establecido en los criterios del formato de entrega, se realizan las pruebas con éxito y se evidencia mediante un gif. Se encuentra todo documentado en el README.md **1 punto**
Ejercicio 2	No se cumple con alguno de los criterios del formato de entrega o no se realizan las pruebas o no se realiza el gif que verifique el correcto funcionamiento del ejercicio. **0 puntos**	Se cumple con lo establecido en los criterios del formato de entrega, se realizan las pruebas con éxito y se evidencia mediante un gif. Se encuentra todo documentado en el README.md **1 punto**
Ejercicio 3	No se cumple con alguno de los criterios del formato de entrega o no se realizan las pruebas o no se realiza el gif que verifique el correcto funcionamiento del ejercicio. **0 puntos**	Se cumple con lo establecido en los criterios del formato de entrega, se realizan las pruebas con éxito y se evidencia mediante un gif. Se encuentra todo documentado en el README.md **2 puntos**
Ejercicio 4	No se cumple con alguno de los criterios del formato de entrega o no se realizan las pruebas o no se realiza el gif que verifique el correcto funcionamiento del ejercicio. **0 puntos**	Se cumple con lo establecido en los criterios del formato de entrega, se realizan las pruebas con éxito y se evidencia mediante un gif. Se encuentra todo documentado en el README.md **2 puntos**
Ejercicio 5	No se cumple con alguno de los criterios del formato de entrega o no se realizan las pruebas o no se realiza el gif que verifique el correcto funcionamiento del ejercicio. **0 puntos**	Se cumple con lo establecido en los criterios del formato de entrega, se realizan las pruebas con éxito y se evidencia mediante un gif. Se encuentra todo documentado en el README.md **4 puntos**