18/19

Lab 1

DESARROLLO WEB – 3TI11

JUAN LUIS HERNÁNDEZ GARCÍA – JUAHERG7@INF.UPV.ES
JUAN CARLOS CANTOS AGUDO – JUACANAG@INF.UPV.ES

UNIVERSIDAD POLITECNICA DE VALENCIA

La maquina virtual en la que se encuentra la practica es: dew-juaherg7-1819.dsci.cloud

Arreglar HTML	Completo
Estilo Monserrat	Completo
Estilo Fondamento	Completo
Bootstrap Sketchy	Completo
Bootstrap Superhero	Completo
Formulario JS	Completo
Formulario HTML	Bastante

Arreglar HTML:

El primer fallo es la cabecera que estaba diseñada para un HTML 4.1, la cambiamos para que fuese compatible para HTML 5. Algunas palabras utilizaban una fuente que se quito en la versión HTML5 (<tt>), la cambiamos por otra igual pero que se puede utilizar en HTML 5 (<code>), también habían problemas con listas, se creaba una dentro de otra si decir que era un elemento de esta, es decir, no se ponía el <li> antes de crearla, aun así cuando pones el <li> sale un círculo que queremos quitar, esto lo arreglamos añadiendo el tag: style= ”list-style: none. Este problema con las listas era el más usual.

Estilo Monserrat y Estilo Fondamento:

En el <header> del html enlazamos el archivo con una hoja de estilo css con esta línea de código:

<link rel="stylesheet" type="text/css" href="css/estiloMontserrat.css"> y <link rel="stylesheet" type="text/css" href="css/estiloFondamento.css"> (estos archivos css los tenemos en una carpeta llamada css).

Además cambiamos las fuentes de las letras para que quede como nos pide el ejercicio.

Bootstrap Sketchy y Bootstrap Superhero:

Descargamos la carpeta de Bootstrap y los estilos superhero y sketchy de Bootswatch, creamos un par de carpetas de Bootstrap y a cada uno le ponemos un estilo diferente de los que descargamos de Boostwatch, cono esto solo queda poner la línea de código para unir el html con el css como hicimos con Estilo Monserrat y Estilo Fondamento:

<link rel="stylesheet" type="text/css" href="css/Boostrap.min.css">

Además para que quede exactamente como la página de ejemplo la inspeccionamos para saber su código y añadir el que nos faltaba, por ejemplo el nav:

<nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">

Y todo el demás contenido lo metemos en dos div:

* *

* *

Además tuvimos un problema con una tabla, las líneas no eran igual al ejemplo porque eran más finitas, esto lo arreglamos forzando que hiciera el borde más grueso añadiendo un tag a la tabla:

style="border:1px solid white

Formulario JS:

Para que se revisen los campos con javascript metemos un <script> en el <header> y creamos una string donde se irán añadiendo los errores que detectemos en los condicionales de cada campo, posteriormente saltara una alerta con todos los errores que detectemos cuando le demos al botón “Enviar”, si no detecta errores te llevara a la página de memex y escribirá los valores que los campos del formulario tenían. Un ejemplo de comprobación es este, que utilizamos para ver si el DNI es correcto:

if (f.dni.value == "") mensaje += "El NIF no puede estar vacio.\n";

else { if ((dni.length != 10) || !(esEntero(dni.substr(0, 8)))) mensaje += "El NIF no se corresponde con el formato esperado 8 digitos, separador y letra.\n";

else { letras = "TRWAGMYFPDXBNJZSQVHLCKET";

var u = letras[dni.substr(0, 8) % 23];

if (!dni.endsWith(u)) mensaje += "La letra del NIF no coincide (deberia ser " + u + ")\n"; } }

Formulario HTML:

Como en este caso solo podíamos comprobar errores mediante html añadimos tags especiales como:

required minlength="1" (para asegurarnos de que haya algo escrito y el campo no esté vacío)

*required pattern="[0-9]**"* (para asegurarnos de que solo hayan números)

required minlength="12" required maxlength="12" (para asegurarnos de que hay un total de 12 caracteres justos)

Con HTML hay bastantes más limitaciones y por tanto no se puede comprobar de manera eficiente todos los campos.