Definición Entrega ET3
Interfaces de Usuario
Curso 2024-2025

Tipología

VOLUNTARIA (Si se presenta deben seguirse las normas de aplicación establecidas en la guía docente)

Realización en grupo de como máximo cuatro miembros, de los cuales uno es el líder. El líder puntúa sobre 12 en vez de 10 que lo hacen el resto de miembros del grupo.

Entrega individual por parte del líder del grupo.

La ET3 tiene una dedicación entre teoría y práctica de 40 horas por alumno y se plantea con una duración estimada de 120 horas en una holgura por grupo de 4 personas de 40 horas.

Definición

Siguiendo la arquitectura de directorios definida en la ET2, (js_core, js_base, locale, iconos, css, index.html), colocando Validaciones_Atomicas.js y Test_Class.js en js_core, EntidadAbstracta en js_base, idioma en locale.

Se solicita que en un fichero estructura_nombreentidad.js en js_app se definan dos estructuras de datos que permitan rellenarlas con la información necesaria para poder generar automaticamente las tablas de presentación de información, los formularios de acciones y las validaciones de los campos de formulario. Estas estructuras serían:

variable def_html_nombreentidad que defina para una tabla la información necesaria para definir los elementos html que representan dentro de un formulario los atributos de la tabla

variable def_test_nombreentidad que defina las validaciones por acción junto con los mensajes de respuesta de error de las validaciones

Una vez que se establezca la estructura de datos, esta debe poder rellenarse con la información necesaria para cada tabla a gestionar en la interfaz.

Se solicita también:

Utilizar la clase test para que verifique el formato de test y pruebas y ejecute todas las pruebas de test de las definiciones de test definidas para cada atributo (incluidas las de fichero) y saque su resultado en un componente modal con scroll para ser visualizado por el usuario. La acción de test debe estar disponible como en la ET2 en la parte superior del index cuando se entra en la gestión de una entidad.

Debe utilizarse la clase validacionesatomicas para la implementación de los métodos estándar de validación de campo que se indican en la definición de la ET2.

Implementar un método createForm() en la clase EntidadAbstracta (y métodos accesorios si es necesario) que, a partir de los datos de las estructuras de datos, genere los formularios de ADD, SEARCH, EDIT, DELETE y SHOWCURRENT, compruebe la información introducida en los campos del formulario y si es correcta permita realizar la llamada a BACK. Este método debe permitir que si existe un método cargar_formulario_html() en la clase de la entidad se ejecute y cargue el contenido html del formulario y si no existe se cree dinámicamente el formulario a partir de la estructura de datos con la información html de cada campo del formulario.

Debe permitirse una modificación especial del valor de presentación de un atributo en la tabla de presentación mediante un método que se declare en la clase de la entidad.

Debe permitirse una validación especial de valores de campos que no corresponda con las estándar definidas en la ET2 que se coloque en la clase de la entidad y que pueda ser invocada en las pruebas de la entidad.

Debe hacerse los ficheros de definición de test y de pruebas para los campos de los formularios correspondiente al nuevo código desarrollado.

En general, debe permitirse en la clase entidad la personalización de la misma para poder adecuar fuera del estándar el comportamiento de la entidad.

Debe existir una clase para la construcción del formulario a partir de la estructura de datos de definición html del formulario.

Debe existir una clase para la colocación de los valores de los atributos para las acciones de EDIT, DELETE y SHOWCURRENT

Debe existir una clase para la construcción de las validaciones de campo y submit en los formularios para las acciones.

En la página index.html, al abrirse, debe mostrarse (no estático en html) la información de un fichero ET3_Datos_*NombreGrupo*.js con una variable de tipo array con el nombre def_grupo_*nombreGrupo* con la siguiente información (este fichero debe estar a nivel de index.html) :

Entrega,
nombre de grupo,
integrantes,
horas dedicadas por integrante,
horas totales

En la página index.html debe existir un icono que lleve a una página estructuras.html en donde se describan las estructuras de datos realizadas y un ejemplo de su uso. Este fichero debe estar a nivel de index.html.

En la página index.html debe existir un icono que lleve a una página API.html en donde se describen las funciones desarrolladas indicando y describiendo para cada una de ellas los parámetros que utilizan y el output que devuelven a nivel de nombre, tipo de dato y

descripción. Las funciones deben estar separadas en función de si se utilizan para los test, para la tabla de presentación o para la creación de los formularios. Este fichero debe estar a nivel de index.html.

Debe existir un fichero IU.css en el directorio /css que contenga todas las reglas de estilo aplicables en el interfaz. La interfaz de las páginas web debe poder visualizarse desde 1920x1080 píxeles hasta 960×540 píxeles de forma dinámica. A partir de 960×540 píxeles debe cambiar al formato móvil de una columna con la resolución 640×480 píxeles como tamaño mínimo. El formato de index.html debe tener un encabezado, un pie de página, un icono de menú debajo de la cabecera (para mostrar las entidades que se pueden seleccioinar) y una zona de trabajo. Se mostrará en un modal los resultados de test y se mostrarán la tabla y los formularios (modales) y los mensajes de acción (modales).

El líder debe mandar por correo electrónico a jriglesias@esei.uvigo.es la composición del grupo con copia a cada uno de los miembros del mismo. Los miembros del grupo deben responder a todos al correo del líder indicando que están de acuerdo en formar parte del grupo. Todo esto debe hacerse antes del día 1 de diciembre de 2024. Posteriormente a esta fecha no se permitirán modificaciones en la composición de los grupos ni nuevos grupos.

Objetivos

1) Debe funcionar para cualquier tabla de una base de datos. Toda la información necesaria debe estar en las estructuras de datos definidas.

2) Deben poder definirse definiciones de test y pruebas de test para todos los campos de la tabla incluyendo los campos de tipo file.

3) La realización de los test debe realizarse en index.html

4) Debe utilizar los códigos definidos en textos_ES y textos_EN para la información de interfaz.

5) Las acciones deben estar representadas por iconos.

6) Los datos del grupo mostrados en la página index.html debe leerse el fichero ET3_Datos_*NombreGrupo*.js

7) Debe utilizarse todo lo desarrollado para implementar los test y la gestión y acceso a BACK de las tablas analysis_preparation, project y characteristic (definida al final del documento), los cuales deben ser accesibles desde index.html.

Historias de usuario a cumplir

Particulares de entrega (Obligatorio. Si se incumple alguno de estos criterios la nota será 0)

1. Los ficheros tienen el nombre, formato y tipo indicado en la entrega
2. El directorio a entregar existe y tiene el nombre indicado en la entrega

3. Se han indicado el número de horas utilizadas en la realización de la entrega por cada alumno.

Por cada error en la estructura de directorios, ficheros o variables solicitadas : 0,1

Por cada error en lo solicitado en los ficheros de información mostrados desde el index.html : 0,1

Por cada error en los test : 0,1

Por cada error en los formularios : 0,1

Por cada error en la interacción en interfaz : 0,1

Por cada error en la apariencia de la interfaz : 0,1

Forma de entrega

1) Colocar todo el código en un directorio con nombre CODIGO

2) Crear un directorio con el nombre ET3_*NombreGrupo* y poner en su interior el directorio CODIGO

3) Comprimir el directorio ET3_*NombreGrupo* en formato .rar con el nombre ET3_*NombreGrupo.*rar

4) Subir el fichero ET3_*NombreGrupo*.rar al ejercicio ET3 en Moovi.

Modo de corrección

Se descomprime el fichero .rar proporcionado por el grupo.
Se coloca en un directorio de la máquina virtual. NO necesariamente en el directorio raíz.
Se verifica la información proporcionada a nivel de grupo, estructuras de datos, test y api.
Se verifica el funcionamiento para las entidades propuestas.
Se verifica el funcionamiento para una tabla nueva usando las estructuras de datos de la entrega.
Se verifica la presentación de la interfaz conforme a las indicaciones de corrección proporcionadas.

(SE ENTREGA ANTES DEL VIERNES DÍA 10 DE ENERO A LAS 23:59 HORAS)

```
CREATE TABLE `characteristic` (
  `id_characteristic` int(11) NOT NULL AUTOINCREMENT,
  `name_characteristic` varchar(100) NOT NULL,
  `description_characteristic` varchar(5000) NOT NULL,
  `data_type_characteristic` enum('number','text','set') NOT NULL,
  `category_characteristic` enum('soil_site','soil_chem','soil_bio') NOT NULL,
  `bibref_characteristic` varchar(200) NOT NULL,
  `file_characteristic` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
```

id_characteristic	dígitos min 1 max 11
name_characteristic	alfabéticos y espacios sin acentos ni ñ, min 8 max 100
description_characteristic	alfabéticos y espacios sin acentos ni ñ, min 80 max 5000
data_type_characteristic	
category_characteristic	
bibref_characteristic	alfabéticos con acentos, ñ, espacios y signos de puntuación, min 16 max 200
file_characteristic	alfabéticos con punto sin acentos ni ñ ni espacios min 7 max 100. Solo los mime type pdf, doc o docx y tamaño de fichero menor de 200000 bytes.