Trabajo Práctico Especial 2024

Parte II

Consigna

Para la segunda entrega, se debe continuar el trabajo de la primera etapa. El objetivo es agregar nueva funcionalidad detallada a continuación a una tabla dinámica.

NOTA: En caso que la tabla ya existente no tenga sentido una carga dinámica de información se puede agregar un segunda tabla que tenga los requerimientos descritos a continuación.

Detalle de Requerimientos

Tabla dinámica por API REST

La tabla dinámica que exista en sus páginas tiene que estar sincronizada en todo momento con la información almacenada en un servicio web (API Rest).

Agregar en su página el código necesario para crear la tabla y llenarla con la información que obtienen del servicio. Cada “fila de la tabla” estará entonces asociado a un “item” o “elemento” del servicio web consultado. A su vez se debe agregar un formulario para crear información, dicha tabla debe mantenerse actualizada cuando el usuario agregue/edite/borre información, además de mantener actualizados los datos en el servicio..

El servicio REST deben generarlo utilizando https://mockapi.io/, creando la API con la colección de datos que corresponden a cada trabajo. En la unidad de 3 de Moodle, está disponible el video donde se explica cómo crear un servicio en mockApi.

Ejemplo: si la tabla almacena “cervezas” el endpoint será similar a
https://5f8sd3342342ca.mockapi.io/api/v1/cervezas

Requerimientos Funcionales

La tabla se debe cargar automáticamente al mostrarse en la página: Al entrar por el nav a la página que tiene la tabla, automáticamente debe mostrarse la tabla cargada sin que el usuario tenga que hacer ningún click adicional, con los mismos datos que inicialmente existan en el servicio Rest.

La información se carga a través de un formulario, que se debe implementar. Estos datos se deben agregar usando la API Rest. Es decir los datos que se agregan mediante el formulario deben actualizar el servicio y la tabla.

Permitir eliminar filas de la tabla de a una. Cada fila tiene que tener una forma de indicarle “Borrar” que elimine la fila que uno desea borrar y también se debe eliminar el elemento correspondiente del servicio.

Permitir editar filas de la tabla individualmente de alguna forma. Cada fila tiene que tener una forma de indicarle “Editar” que permita editar los valores de esa fila y los actualice en el servicio.

Limitar tamaño de imágenes y peso del sitio. La carpeta completa del sitio no puede superar los 5 Mb.

Opcionales

1. Hacer deploy del sitio completo en un HOST (**+2**)

   Subir la página completa a algún servicio de hosting gratuito, anexar a la entrega del proyecto un archivo urls.txt que contenga la url de acceso de la página y del servicio de mockApi:

   Algunas opciones

   - https://glitch.com/
   - https://www.000webhost.com/

2. SPA / Partial Render (**+2**) Usar AJAX para la navegación de la página (técnica de partial render). La página no debe refrescarse completamente, cuando hago click en un link de la navigation bar se refresca sólo la porción de la información que cambia.
3. Paginación (**+2**): se debe poder traer la información del servicio rest de forma paginada desde el servidor. Leer documentacion https://www.mockapi.io/docs
   Recomendación: solo botones “anterior” y “siguiente”.
4. Creación de varios Items (**+1**): Debe haber un botón que permita crear varios ítems automáticamente (al menos 3 items), esos datos deberán agregarse en el servicio y verse en la tabla.
5. Filtrado (**+1**): Agregar filtros de búsqueda en la tabla desde JS (local). Al filtrar los datos, se mostrarán solo los datos que cumplan ese criterio, los demás datos no se ven, pero no se borran realmente. Sugerimos dos formas de hacer el filtro (otras opciones debatir con los docentes):
   - filtros por combo (tag select) para las columnas con opciones fijas
   - filtrar con un input por al menos una columna.

   En el filtrado no es requisito que el resultado se actualice permanentemente con el cambio de caracteres por ejemplo en el input.

Aclaración partial render de navegación (solo aplica si hace OPCIONAL SPA / PARTIAL RENDER):
La página principal debe tener el contenido en blanco, pero al abrirla el usuario se debe mostrar el contenido completo (se descarga con AJAX automáticamente)

Aclaración Tabla REST: No se permite el uso de JQuery para REST

ENTREGA - MOODLE

La presentación es de acuerdo a los grupos definidos en la planilla correspondiente. Todos los grupos deben tener el contenido definido en la planilla de trabajos y estar en el lugar acordado. **Los grupos pueden ser hasta 2 personas como máximo**

- Planilla Comisiones A
- Planilla Comisiones B

- Planilla Comisiones C
- Planilla Sede RAUCH

El proyecto debe entregarse y subirse como archivo comprimido con nombre según número (Columna “N° Trab”) de grupo, comisión y nombres y subirlo a la tarea de entrega de MOODLE, un solo trabajo por grupo.
Ejemplo 1 (“C009 - C1 Jose Perez y Juan García”), ambos en comisión C1
Ejemplo 2 (“C010 - C1 Raul Gomez y C2 Fernando Hernandez”), en comisiones C1 y C2

La entrega del trabajo se hace por Moodle, ingresando en la sección “Entrega Segunda Parte”, que aparecerá próximo a la fecha .

Se debe subir como entrega un archivo compactado que incluya : El proyecto completo del sitio web y un archivo urls.txt que incluya la url del servicio Rest y la url del sitio (en caso que se haya implementado el opcional 1. del Host).

Fecha de entrega :

TANDIL:
- Entrega: Lunes 24/06 13.00 hs.
- Defensa: A coordinar semana 24/06

RAUCH
Entrega: Lunes 24/06 13.00 hs.
Defensa: A coordinar semana 24/06

Defensa
La defensa se coordinará según cronograma, es sobre entrega 1 y 2.

Criterios de evaluación

La entrega es obligatoria. La no entrega de este TPE implica pérdida de cursada.

La estructura HTML, CSS y JS debe ser de acuerdo a las diferentes pautas explicadas. En caso de errores graves se descontará puntaje.

Se evaluará la correcta implementación del TPE: no repetición de código, identificadores (nombres de clases, variables, etc) descriptivos, etc. Los trabajos deben implementar la totalidad de los requerimientos funcionando correctamente, sin tolerancia a bugs.

Se considera fundamental la aplicación de buenas prácticas, y la elección apropiada de cada tecnología para cada punto a resolver.

El sitio debe ejecutar correctamente desde otra computadora (tener cuidado en las rutas de los archivos y dimensiones demasiado rígidas de los elementos, etc).

La nota se calculará partiendo de 10, a partir de ahí se suma/resta:

- Item sin implementar (o mal implementado, por ej, no se corresponde con la planilla): -4.
- Item que no anda: -2.
- Item con bug menor: -1.
- Poca Prolijidad General: (código difícil de leer, mal indentado / alineado, código desordenado etc): hasta -4.
- Malos nombres (nombres no representativos, error general, no casos puntuales): pierde promoción y -2
- Hacer prácticas marcadas como malas durante la cursada: hasta -6
- Item Opcionales: suman según lo especificado
- Defensa con calificación individual: se puede desaprobar o disminuir la nota por una defensa insuficiente