| Mostrando Marca_Txurdinaga_Color_H_SUP_01_01.jpg | Curso / *Kurtsoa* | Fecha / *Data* | Nivel / *Maila* | Eval. /*Ebal.* |
| --- | --- | --- | --- | --- |
| 2023-24 | 02/02/2024 | 2º | 2ª |
| Módulo / *Modulua* | *Kodea / Código* | *U.Didak /Unid didác* | *tipo* | Calificación/*Kalifikazioa* |
| DWC | 2DW3 | 1, 2, 3, 4, 5, 6, Vue.js | P |  |
| Nombre Alumno/a / Ikaslearen izena |  | | | |

**EXAMEN MÍNIMOS**

* Este examen es de tipo procedimental.
* Tiene un valor total de 10 puntos.
* **En cada uno de los ejercicios es necesario sacar un 40% de la nota (0,8 sobre 2 en el primero y 4 sobre 10 en el segundo)**
* Si la propuesta del examen se resuelve de forma correcta se le aplica el valor especificado en cada apartado.
* Descargar el scafolding desde Github Classroom. en vuestro equipo, haced los commits necesarios. Cuando quieras entregar sube el repositorio a Github.

**Ejercicio 1. Javascript: (2 ptos)**

Diseña un programa en Javascript que permita validar el formulario adjunto (formulario.html) de manera que realice lo siguiente:

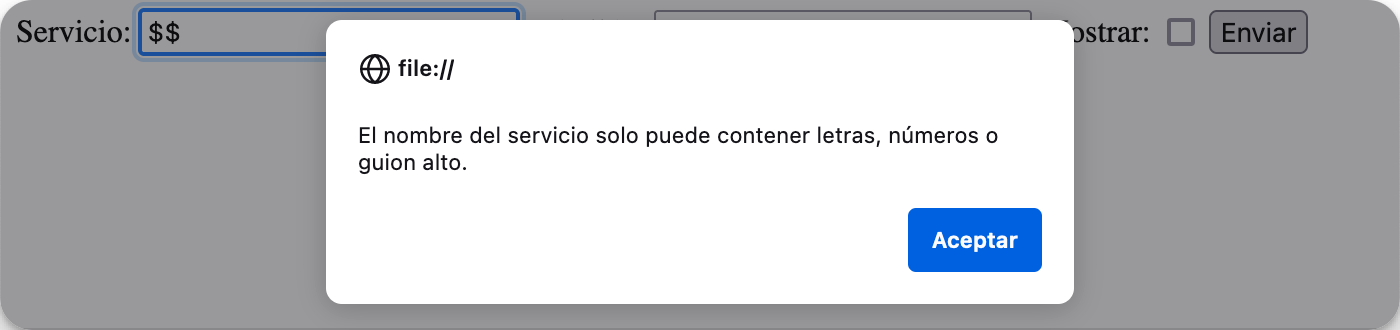
1. Dispone de un archivo Servicio.js que contiene una clase. Los datos que almacenará serán:
   1. servicio (texto).
   2. pabellón (número, correspondiente con la posición del pabellón en el array pabellones). Este array tendrá cuatro elementos: Arrupe, Areilza, Aztarain y Guturbai.
   3. mostrar (booleano).

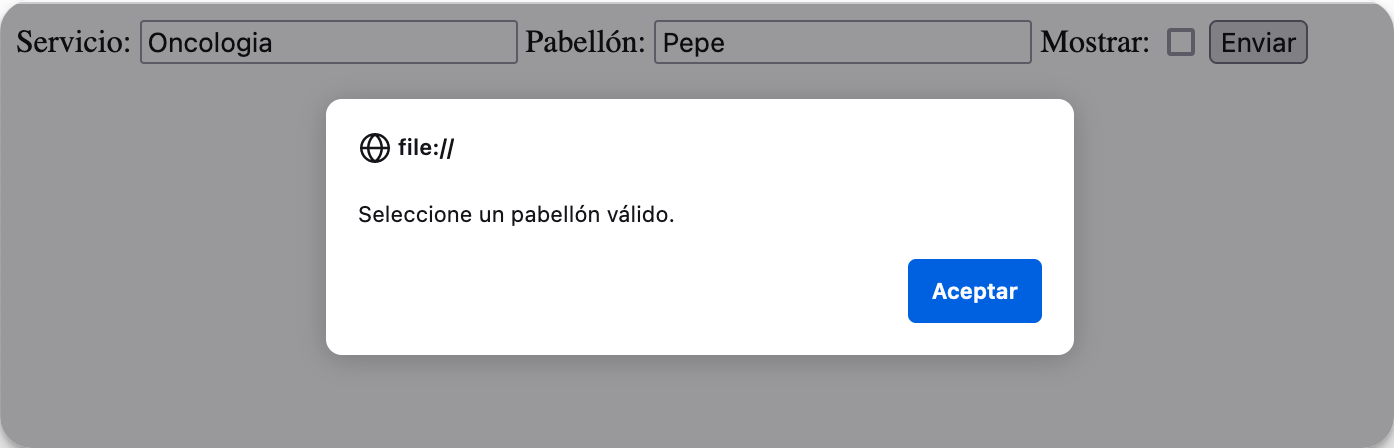
Además, tendrá un constructor y los getters y setters correspondientes. Igualmente incluirá un método *mostrar* que devuelva una línea del tipo: servicio – número de pabellón: nombre de pabellón – visible/no visible. Tiene que devolver el texto, no hacer un console.log.

*Ej: Resonancias – 2: Areilza – Visible.*

1. Validar los datos introducidos utilizando Javascript en el archivo validar.js:
   1. El nombre está formado únicamente por letras, números o guión alto.
   2. El pabellón forma parte del array *pabellones* adjunto en el archivo.
2. Si los datos son válidos, el formulario no se enviará, pero se realizarán las siguientes operaciones:
   1. Se crea un objeto Servicio.
   2. Se añade el objeto a un array llamado *servicios*.
   3. Se muestra por consola el servicio creado.

**Ejemplos de funcionamiento:**







| **Total ejercicio - 1** | **2** |
| --- | --- |
| 1. **Objeto** |  |
| * 1. Todos los getters y setters siguen los estándares | 0,20 |
| * 1. El método constructor ha sido definido correctamente | 0,15 |
| * 1. El método mostrar ha sido definido correctamente | 0,30 |
| 1. **Validaciones** |  |
| * 1. La validación del nombre funciona correctamente | 0,15 |
| * 1. La validación del nombre se hace mediante una expresión regular | 0,25 |
| * 1. La validación del pabellón es correcta y se hace teniendo en cuenta el array | 0,30 |
| 1. **Tratamiento de validación correcta (sólo se corregirá si el punto 2 es correcto)** |  |
| * 1. Se crea un objeto servicio | 0,25 |
| * 1. Se añade objeto al array de servicios | 0,30 |
| * 1. Se muestra por consola el objeto creado | 0,10 |
|  |  |

**Ejercicio 2. Vue.js: (8 ptos)**

El hospital de Basurto ofrece una lista de **servicios** que se encuentran repartidos entre los diferentes **pabellones** que componen el complejo hospitalario.

Los y las pacientes y las visitas que acuden a Basurto, en muchas ocasiones, no tienen claro a qué pabellón/entrada deben acudir.

Por eso, debemos diseñar una aplicación que facilite al usuario toda la **información de los diferentes pabellones/entradas y servicios** que hay en el complejo de Basurto.

Además, el usuario podrá marcar **cuáles son sus servicios favoritos**, que se quedarán guardados y estarán disponibles entre los diferentes componentes Vue tal y como se detalla más adelante.

**En este ejercicio vamos a tomar como base el proyecto ya creado y que podrás descargar del repositorio de Github Classroom.**

Para la carga de datos utilizaremos la **librería Axios** que está en la carpeta *./Datos/Axios/axios.min.js* y el fichero json que está en *./src/js/basurto.json*

El **archivo JSON** contiene toda la información sobre los diferentes servicios que hay en el hospital incluyendo los siguientes campos:

* **SERVICIO**: identifica el servicio, como Anatomía patológica, Extracciones, Psiquiatría, etc.
* **ENTRADA**: se refiere a la entrada o pabellón en el que se encuentra dicho servicio.
* **ICONO**: es el nombre del archivo que contiene el icono que representa el servicio.
* **MOSTRAR**: indica mediante un 1 o un 0 si el servicio debe estar visible para todo el público o únicamente para el personal del hospital.

Como veis en este proyecto ya incluye 2 rutas que apuntan a las vistas ***HomeView.vue***(en la que se carga el componente **HelloWorld.vue**), y la vista ***AboutView.vue***. Tendréis que hacer lo siguiente

1. Modificar la página principal y crear las **rutas necesarias** para gestionar la navegabilidad entre las diferentes páginas. Esta página tiene el siguiente aspecto con dos columnas.



El componente que se carga por defecto a la derecha cuando se accede a la raíz será Bienvenida y tiene este aspecto:

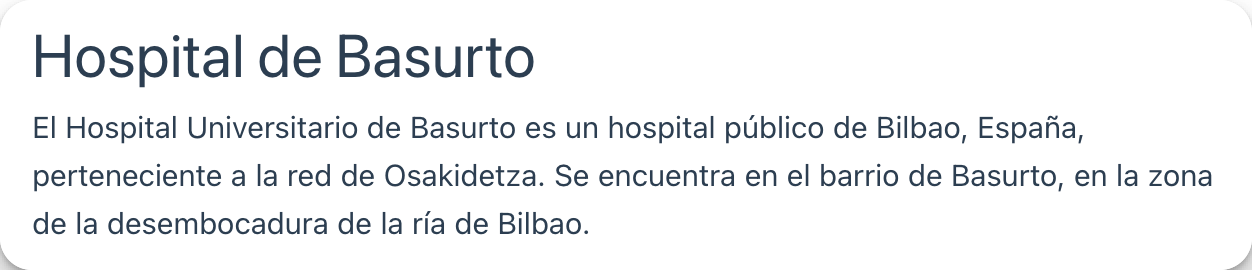


El resto de componentes se cargarán a la derecha cuando se haga click en los diferentes enlaces, sustituyendo Bienvenida, de modo que deberás crear las siguientes rutas:

* /
* /todosServicios
* /serviciosPabellon
* /favoritos

***Nota****: Los logos están disponibles en la carpeta de ./src/assets.*

1. Crear una nueva vista ***ServiciosPublicos.vue***. Esta vista incluye el componente Basurto, que únicamente tiene un pequeño título y un texto.



Además, hay una tabla con cuatro columnas: las tres primeras son Icono, Servicio, Pabellón. **Solamente se muestran los servicios cuyo campo MOSTRAR es 1.**

Permite hacer un **filtrado por nombre de servicio** utilizando un campo de texto teniendo en cuenta que no es sensible a las mayúsculas, de modo que según se van añadiendo letras en el campo de texto, se realiza el filtrado en la tabla inferior. Del mismo modo, si se eliminan letras del campo, la tabla se vuelve a rellenar. Si el campo está vacío, deberán mostrarse todos los servicios.



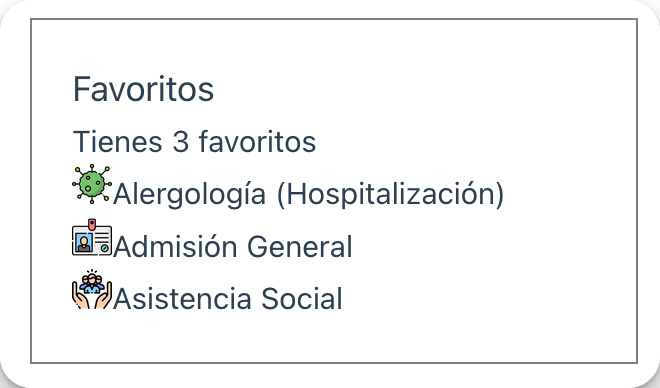
Además de las columnas indicadas en el componente anterior, tiene una columna más llamada **Fav**. Incialmente, esta columna contiene un **icono de un corazón blanco** por cada servicio. Si el usuario hace clic en el corazón, éste **se colorea de rojo**.

Además, **se guarda el servicio en una lista de favoritos** que es la que posteriormente se muestra **CuadroFavoritos.vue**.

Si el usuario vuelve a hacer clic, el corazón vuelve a ponerse blanco y el **elemento se elimina de la lista de favoritos**.

1. El componente **CuadroFavoritos.vue** será un componente independiente. Se trata de un pequeño cuadro que muestra el número de servicios favoritos que ha marcado el usuario y la lista de los mismos.

Para ello, los servicios marcados como favoritos por el usuario se almacenarán en un array que se gestiona como **store** y cuya información se comparte entre todos los componentes que la utilizan (**TodosServicios.vue** y **Favoritos.vue**).



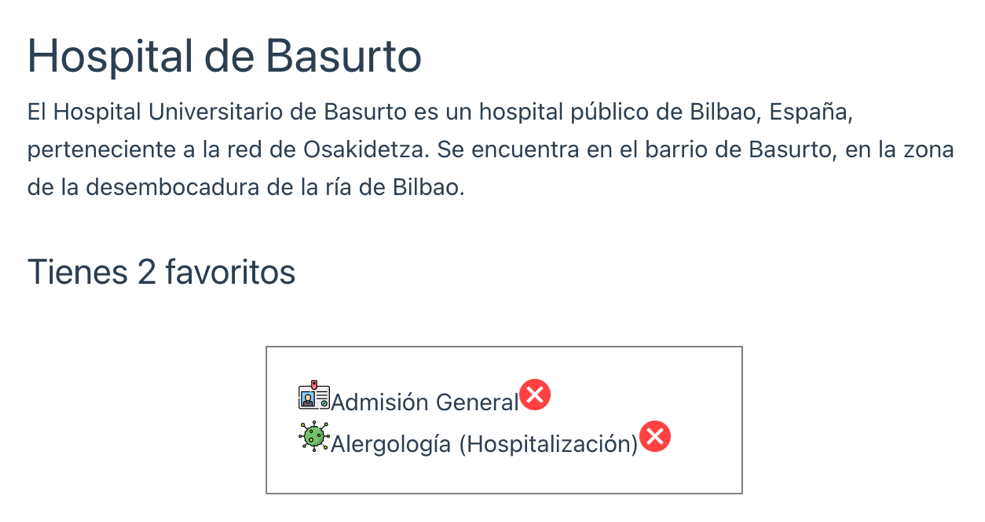
| **Información sobre la store: favoritos.js**  Esta *store* dispondrá de dos *states*:   * **numeroFavoritos**, que almacenará el número de elementos que se han marcado como favoritos (en lugar de utilizar el tamaño del array de favoritos, se utilizará este state). * **favoritos[]**, un array que contendrá todos los elementos Servicio que marque el usuario con el corazón rojo.   A su vez, dispondrá de dos actions:   * **anadirFavoritos**, al que se le pasará un servicio y añadirá al array un el elemento correspondiente y actualizará la variable numeroFavoritos. * **eliminarFavoritos**, al que también se le pasará un servicio, que será el que se elimine del array; también se actualizará la variable numeroFavoritos. |
| --- |

1. Crea una vista llamada ***Favoritos.vue.*** Esta vista incluye en la parte superior el componente Basurto.

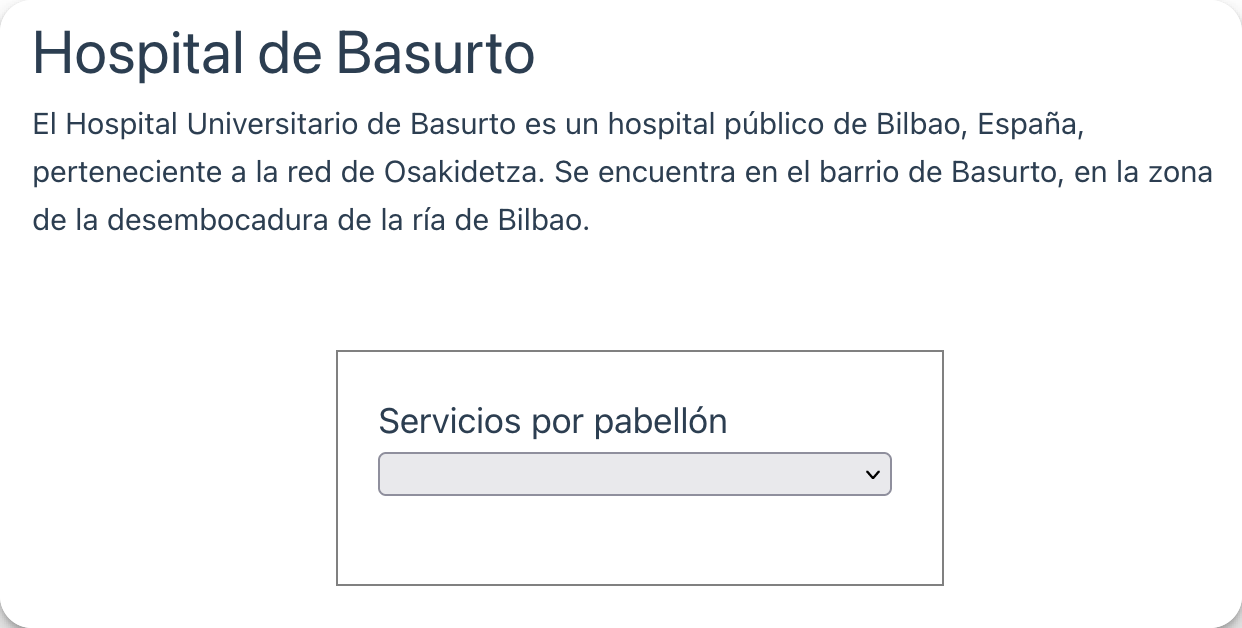
Además, **muestra un texto con el número de favoritos** que ha marcado el usuario, y un pequeño recuadro con la lista de esos favoritos.

Por cada uno de ellos también aparece un pequeño icono con una X de manera que, **al ser marcado, elimina el elemento correspondiente.**

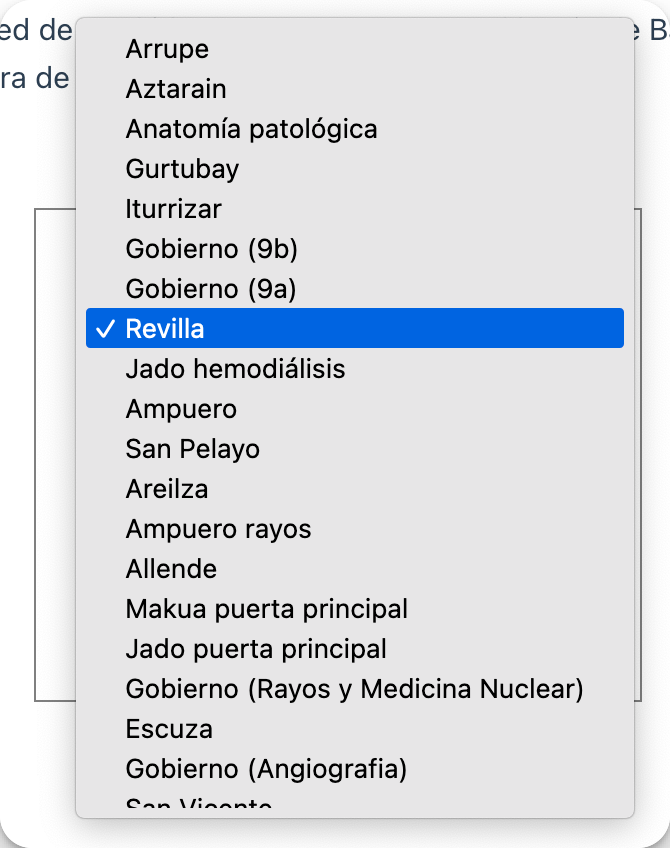
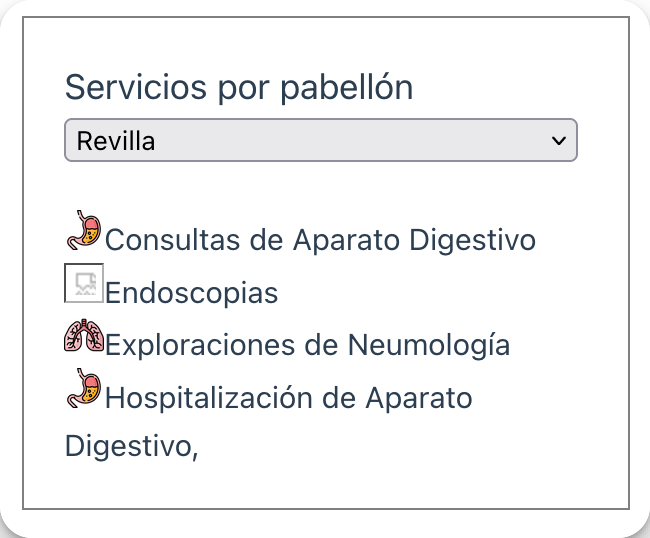
Al igual que en CuadroFavoritos, los servicios favoritos se almacenarán en un array que se gestiona como **store** y cuya información se comparte entre los componentes TodosServicios y Favoritos.



1. Crea una vista llamada ***ServiciosPabellón.vue*:** se trata de un componente que incluye a su vez el componente Basurto.



Dispone de un cuadro con una **lista desplegable que almacena todos los pabellones una única vez**. Cuando el usuario elige un pabellón (entrada) se muestran debajo una lista de servicios que se llevan a cabo en dicho pabellón.

🡪

**Valoraciones:**

| **Total ejercicio - 2** | **10** |
| --- | --- |
| 1. **Home** |  |
| * 1. Se ha modificado la home para que cargue el componente Bienvenida.vue a la derecha. | 0,25 |
| * 1. Se han creado las 4 rutas propuestas y cargan los componentes correctamente. | 0,5 |
| * 1. Todas las vistas contienen el componente Basurto.vue cargado correctamente. | 0,5 |
| 1. **TodosServicios.** |  |
| * 1. Se cargan correctamente los datos del JSON en la tabla. | 0,5 |
| * 1. Los datos cargados contienen únicamente las columnas cuyo campo mostrar es 1. | 0,25 |
| * 1. El filtro funciona tanto al escribir como al borrar letras. | 0,5 |
| * 1. El filtro no es sensible a las mayúsculas. | 0,25 |
| * 1. Al pulsar el icono del corazón, se pone rojo y blanco. | 0,25 |
| * 1. Al pulsar el icono del corazón, el elemento se añade a la lista de favoritos una sola vez. | 0,25 |
| * 1. Al pulsar el icono del corazón, el elemento se elimina de la lista de favoritos. | 0,25 |
| 1. **CuadroFavoritos** |  |
| * 1. El cuadro favoritos muestra una lista con los elementos del array favoritos. | 0,25 |
| * 1. El cuadro favoritos utiliza una store para gestionar los datos. | 0,25 |
| * 1. La store favoritos tiene los states y actions propuestos y funcionan correctamente. | 2 |
| 1. **Favoritos** |  |
| * 1. La vista muestra el texto con el número de favoritos extraído de la store. | 0,25 |
| * 1. La vista muestra la lista de favoritos extraídos de la store. | 0,25 |
| * 1. Es posible eliminar un favorito al hacer clic sobre el botón rojo con el aspa y se elimina ESE favorito. | 1,5 |
| 1. **Pabellones** |  |
| * 1. La lista desplegable muestra todos los pabellones una única vez (no se repiten). | 1 |
| * 1. Al pulsar sobre un elemento, se carga debajo la lista de servicios del pabellón correctamente. | 1 |