

Informe

Parcial N° 1 | Aplicaciones para dispositivos móviles



Prof. Mabel García
Alumna Karina Serrano

Se ha desarrollado un sitio web en el que el usuario puede realizar un test orientativo sobre que juego Eurogames (Juegos de Mesa) jugar segun sus preferencias.

con el uso de html, CSS y **Vue.js**:

Componente <nav-eurogame>

The screenshot shows the main landing page with a title '¿Qué es un Eurogame?' and a brief description. Below it is the 'Test' component, which includes a form for entering 'Nombre' and 'Alias', and three sections of questions with radio button options. At the bottom right of the test component is a 'Ver mi resultado' button.

<Componente-test (padre)>

↓

Contiene componente <respuesta-test> (Hijo)

Se utiliza vue.js para desarrollar la estructura del sitio compuesto por cuatro componentes:

Componente **nav-eurogame** (individual), **componente-portada**(individual),

componente-test (padre) y su componente hijo **respuesta-test**.

El **footer** a excepción del resto del sitio se encuentra codificado en el index.html .

The screenshot shows the first question of the test: '1. ¿Prefieres jugar en solitario o en equipo?'. It has three radio button options: 'Me gusta jugar en solitario.', 'Disfruto de juegos que implican construcción.', and 'Preferio jugar en equipo.'

El **componente-test** es el elemento principal del sitio.

sus Props definidas permiten la comunicación con el **.data** de la instancia Vue principal.

Contiene su propio **data()** con los datos de uso local.

El template cuenta con generación dinámica gracias a sus directivas **v-for**, **v-bind** y **v-if**.

A su vez este componente contiene su propio methods con funciones que se ejecutan en cada uno de sus botones y el botón input submit.

El usuario va a interactuar con dos campos de input tipo texto **Nombre**, **Alias** y campos input tipo radio **Opciones**(generados dinámicamente según la cantidad de elementos de la propiedad **.opciones** del **.data**).

! No me importa si la partida es de duración media.

Prefiero partidas cortas y rápidas.

Ver mi resultado

Limpia todas las variables reactivas

(las de su propio **data()**) atadas a la

información de lo que ingreso el usuario

en el formulario y desactiva la variable

booleana (**mostrarResultado** atada al

v-if del componente hijo **respuesta-test**).

Al hacer click sobre al Submit se ejecuta la función **verResultado()** (método del <componente-test>).

Dentro de dicha función se ejecuta una función importante

(**determinarCategoria()**) que recorrerá

el valor de los radios seleccionados y

con una determinada lógica , obtener la

categoria adecuada para el usuario.

Mostrará además el componente hijo

<respuesta-test> con el juego resultante y guardará los datos ingresados

en la **localStorage**.

Se muestra el juego seleccionado por el <componente-test> de la propiedad **.juego** del data principal.

Los datos se pasan a través del **props** desde el componente padre al hijo.

Botones de interacción para el usuario (like y dislike), los mismos

generados por el componente hijo (<respuesta-test>).

Cuenta con su propia lógica en su **.methods**.

También se guarda en la **localStorage**.

componente <respuesta-test> (Hijo)

Al hacer click sobre el like o dislike se ejecuta la función **guardarLocalStorage()** (método del <respuesta-test>).

Guarda los datos en la **localStorage** y actualiza el resultado.

Ahora con datos disponibles en la **localStorage**, el botón parte del <componente-test>

estará **habilitado**.

El mismo parará a través de la función **traerUltimo()** los datos de la **localStorage** a los input y

también generará en base a estos mismos el último resultado generado.