Prueba - My Store

- Para realizar esta prueba debes haber estudiado previamente todo el material disponibilizado correspondiente al módulo.
- Una vez terminada la prueba, comprime la carpeta y sube el `.zip`

Descripción

Se presenta una aplicación web Vue llamada "MyStore" que permite:

- Entrar al home y ver una lista de productos
- Buscar productos por nombre
- Añadir productos a un carrito de compra indicando la cantidad
- Ver el carrito de compras
- Eliminar productos del carrito
- Limpiar completamente el carrito
- Iniciar sesión con mail y contraseña
- Ver menú de usuario si inició sesión
- Ver un menú link al login si no inició sesión
- Cerrar sesión

El estado del usuario y el carrito se almacena en un Vuex Store.

La lista de productos se consulta a través de un módulo javascript externo que simula una API.

La respuesta del servicio es asíncrona a través de una promesa.

Instrucciones

Parte I

Implementar test suite de pruebas unitarias para la aplicación MyStore.

1. Verificar que `Products.vue` filtre los productos por nombre (Verifique que exista solo un producto y los demás no estén visibles luego de buscar). (**1 Punto**)
2. Verificar que los productos en `Products.vue` se puedan añadir al carro (Utilice localVue y Vuex y verifique que los productos añadidos están en el store). (**2 Puntos**)
3. Verificar que `Navbar.vue` muestre el menú item "Login" cuando el usuario NO está autenticado (Utilice localVue y Vuex). (**2 Puntos**)
4. Verificar que `Navbar.vue` muestre el menú ítem "Usuario" cuando el usuario SI está autenticado (Utilice localVue y Vuex). (**1 Punto**)
5. Verificar que el formulario de login en `Login.vue` muestra un error cuando las credenciales de acceso son falsas (Use Mocks ó Stubs en el servicio Auth y/o método login). (**2 Puntos**)
6. Verificar que el formulario de login en `Login.vue` intente redireccionar a `/home` si el usuario se loguea exitosamente (Use Mocks ó Stubs o Spies para Auth y $router). (**2 Puntos**)

Detalles de la aplicación

Existen los siguientes componentes en `src/components`:

- **Products.vue:** Carga, Muestra y filtra la lista de productos
- **Navbar.vue:** Barra de navegación con los menúes
- **CartDetail.vue:** Ventana Popup con el detalle del carrito

Y las siguientes Vistas en `src/views`:

- **Home.vue:** Vista principal y contiene el componente de products
- **Login.vue:** Vista del form de Login

Además de los siguientes módulos JavaScript implementando los servicios de "API" en `src/services`

- **API:** Simula una API REST que obtiene la lista de productos. `Products.all()` retorna una promesa que resuelve a un array de productos.
- **Auth:** Simula una API REST que loguea al usuario. `Auth.login(creds)` retorna una promesa que resuelve a un objeto 'User' o Lanza un error (`usuario1@mitienda.com` y `password` son credenciales válidas)

Y finalmente, Mock objects para el Vuex Store y las rutas en `tests/unit/mocks`

- **routes.js**: Tiene un objeto con las rutas y los componentes asociados a estas para "home" y "login"
- **store.js:** Implementación simple del Vuex store de la app con actions y getters para "user" y "cart"

Instrucciones

Parte II

Implementar test suite End-to-end para la aplicación MyStore.

1. Verifique la funcionalidad `"Buscar productos por nombre"` (**2 Puntos**)
2. Verifique que sea posible `"Añadir productos a un carrito de compra y ver el carrito"` (**3 Puntos**)
3. Verifique que sea posible `"Eliminar productos del carrito"` (**2 Puntos**)
4. Verifique la funcionalidad de `"Iniciar sesión con mail y contraseña"` (Debe comprobar que la sesión del usuario está activa) (**3 Puntos**)

Detalles de la aplicación

Existen los siguientes componentes en `src/components`:

- **Products.vue**: Carga, Muestra y filtra la lista de productos
- **Navbar.vue**: Barra de navegación con los menúes
- **CartDetail.vue**: Ventana Popup con el detalle del carrito

Y las siguientes Vistas en `src/views`:

- **Home.vue**: Vista principal y contiene el componente de products
- **Login.vue:** Vista del form de Login

Además de los siguientes módulos JavaScript implementando los servicios de "API" en `src/services`:

- **API:** Simula una API REST que obtiene la lista de productos. `Products.all()` retorna una promesa que resuelve a un array de productos.
- **Auth:** Simula una API REST que loguea al usuario. `Auth.login(creds)` retorna una promesa que resuelve a un objeto 'User' o Lanza un error (`usuario1@mitienda.com` y `password` son credenciales válidas)

Y finalmente, Mock objects para el Vuex Store y las rutas en `tests/unit/mocks`:

- **routes.js**: Tiene un objeto con las rutas y los componentes asociados a estas para "home" y "login"
- **store.js:** Implementación simple del Vuex store de la app con actions y getters para "user" y "cart"

Recursos

Material complementario para la realización de la prueba:

- Descomprima el archivo `Apoyo Prueba - My Store.zip` para la aplicación en versión Mocha o Jest.
- Instale sus dependencias con `npm install`