



**Máster Universitario en Ingeniería Informática**

**Curso :** 2020 - 2021

**Asignatura :** Redes Sociales, Colaboración en Red

**Práctica :** SCNI Videoclub

**Alumno :** Daniel Serena Sanz, Víctor García Bermejo

**Madrid, 2020**

Contenido

[Contenido 2](#_Toc71491621)

[Introducción 3](#_Toc71491622)

[Aplicación base 3](#_Toc71491623)

[Análisis 3](#_Toc71491624)

[Diseño 4](#_Toc71491625)

[Pantallas 5](#_Toc71491626)

[Index 5](#_Toc71491627)

[Login / Create 5](#_Toc71491628)

[Profile 5](#_Toc71491629)

[AddItem 6](#_Toc71491630)

[Supermercado 6](#_Toc71491631)

[Base de datos 7](#_Toc71491632)

[Market 7](#_Toc71491633)

[Usuarios 8](#_Toc71491634)

[Conclusión 8](#_Toc71491635)

[Referencias 8](#_Toc71491636)

[Repositorio 8](#_Toc71491637)

[Hosting 8](#_Toc71491638)

[Otras referencias 9](#_Toc71491639)

# Introducción

Se entiende como aplicación web, aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor web a través Internet o una intranet mediante un navegador. Uno de los motivos más importantes por los que se utilizan, es debido a la actuación como un cliente ligero por parte del navegador independientemente del sistema operativo en el que se utilice y por ello las aplicaciones web ganan una gran compatibilidad con todos los sistemas.

El sistema clásico que siguen las aplicaciones web es sencillo: El navegador como cliente realiza una petición al servidor donde recopila la información y los datos necesarios para crear una página con una lógica por detrás, y la envía al cliente en forma de respuesta. En este momento el navegador interpreta la respuesta y la muestra al usuario por pantalla.

A lo largo del tiempo se han ido mejorando las aplicaciones web. Actualmente los problemas a tratar dentro de este mundo son la latencia ya que la mayor parte de los cálculos y cambios se hacen en el servidor y el cliente se ve obligado a descargar constantemente recursos; y la escalabilidad del servidor en función de la cantidad de usuarios accediendo a los recursos y la cantidad de datos que deba manejar el servidor.

Una de las soluciones que se plantean en esta asignatura para resolver estos problemas, son el uso de aplicaciones web progresivas[1] en las que, todo lo que es la presentación se encarga el lado cliente, se ejecutan en segundo plano e incluso se pueden instalar a partir de los navegadores.

En esta práctica usaremos Nuxt[2] utilizando VUEX y Nodejs (nvm)[3], con la idea de generar una aplicación web que gestione y genere páginas y el estado de las sesiones utilizando componentes. En el primer apartado que compondrá la aplicación base, haremos un breve análisis donde se marcarán unos pequeños objetivos del sistema a implementar. Seguidamente en el apartado de diseño se propondrá un diseño del flujo que seguirán los usuarios que utilicen la aplicación. En el apartado de pantallas se dará una breve explicación de los elementos más relevantes de cada página. Para finalizar se presentarán la base de datos con las estructuras donde se guardarían los datos y el documento se cerrará con unas conclusiones.

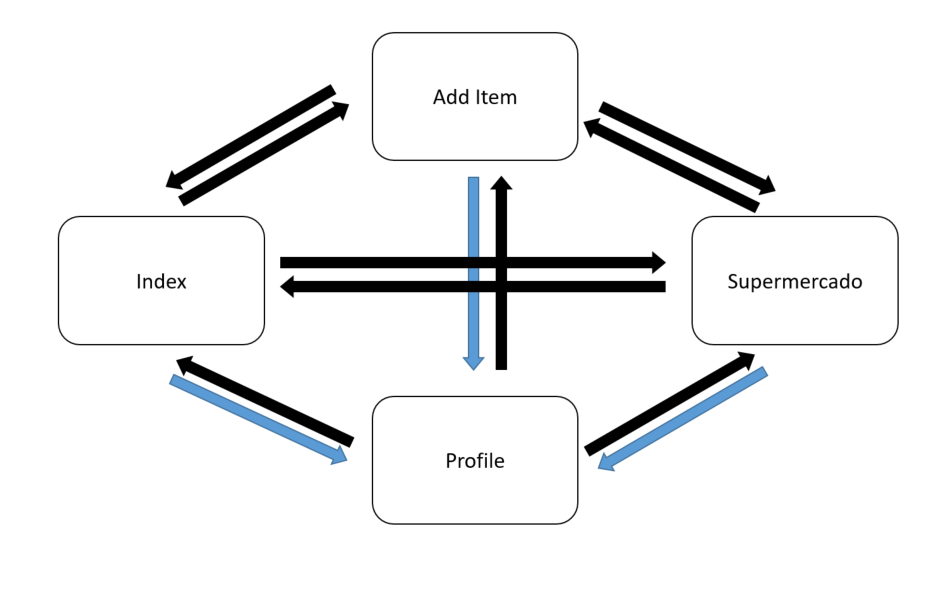
# Aplicación base

## Análisis

Para la realización de esta práctica, se plantean los objetivos

## Diseño

Para el diseño de la plataforma se planteó una interacción sencilla entre las diferentes páginas de la misma. Acorde a esto, creamos el siguiente flujo:



La aplicación web, constaría así de cuatro páginas que serían la página principal o Index, el Supermercado, el Add Item y la página de Profile. Tal y como muestran las flechas propuestas en el diagrama, se puede distinguir dos tipos de interacciones entre las páginas:

* En el caso de las flechas negras, indican un acceso entre las páginas sin necesidad de haber iniciado sesión. Cabe destacar que, se han marcado las interacciones de salida de la página del perfil, como que no tienen necesidad de tener la sesión iniciada puesto que en las páginas entrantes, no existe ningún tipo de restricción de entrada.
* En el caso de las flechas azules, indican la necesidad de haber iniciado sesión antes de poder acceder a una página.

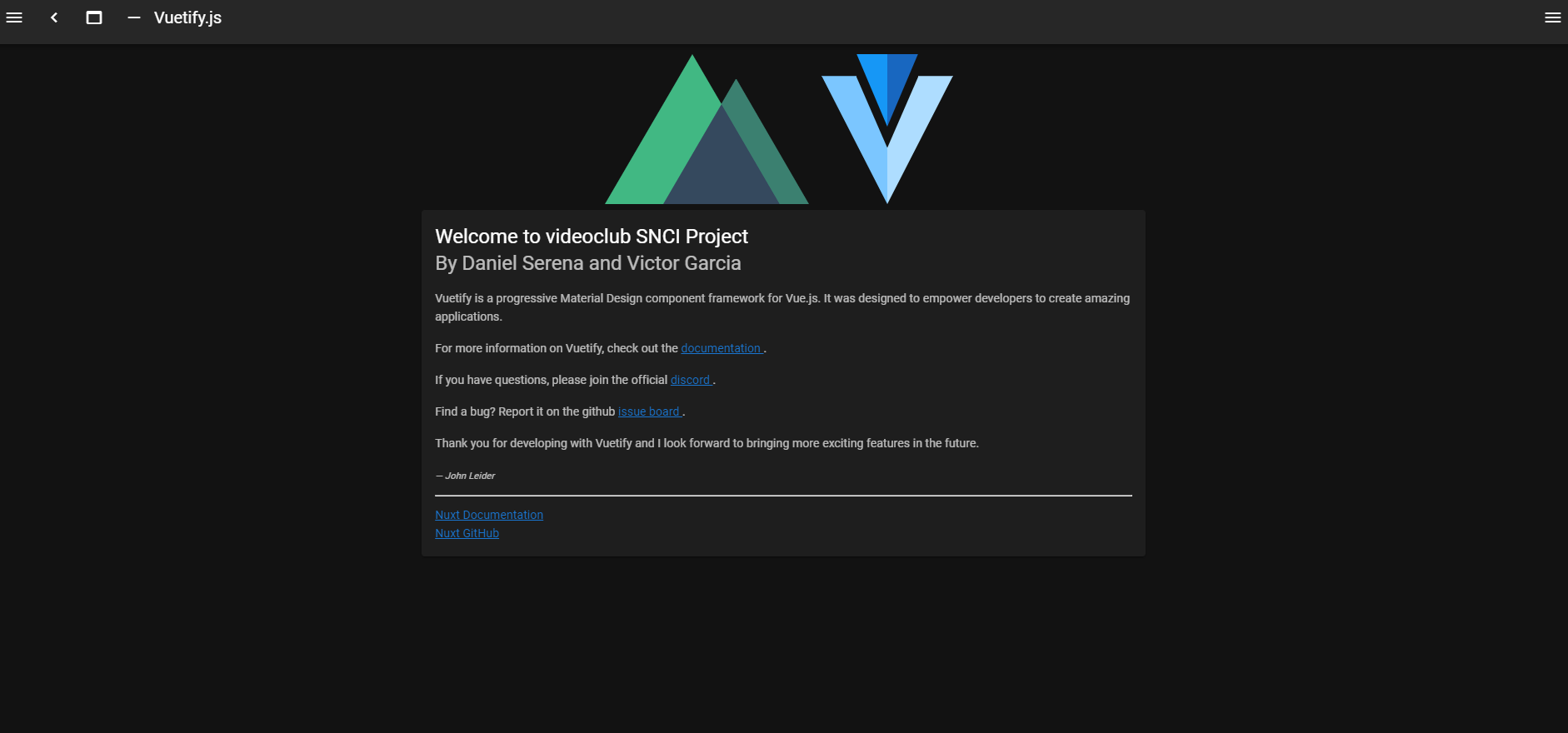
Por este segundo caso y conociendo que no hay una página de inicio de sesión, en el menú lateral izquierdo de todas las páginas, estará un componente que servirá para iniciar sesión dentro de la página de dos formas, la primera mediante un correo de google y la segunda mediante un correo en el momento. Si el usuario no existe, tras la solicitud del email, se creara una nueva cuenta pidiendo al nuevo usuario su nombre y una contraseña.

Adicionalmente, la funcionalidad de alquilar películas de la página del supermercado no será utilizable a no ser que el usuario esté registrado y haya iniciado sesión. Sin embargo, el resto de funcionalidades que tiene la aplicación web como valorar las películas o añadir películas a la base de datos sí que serán usables sin necesidad es estar dentro de una sesión.

## Pantallas

### Index

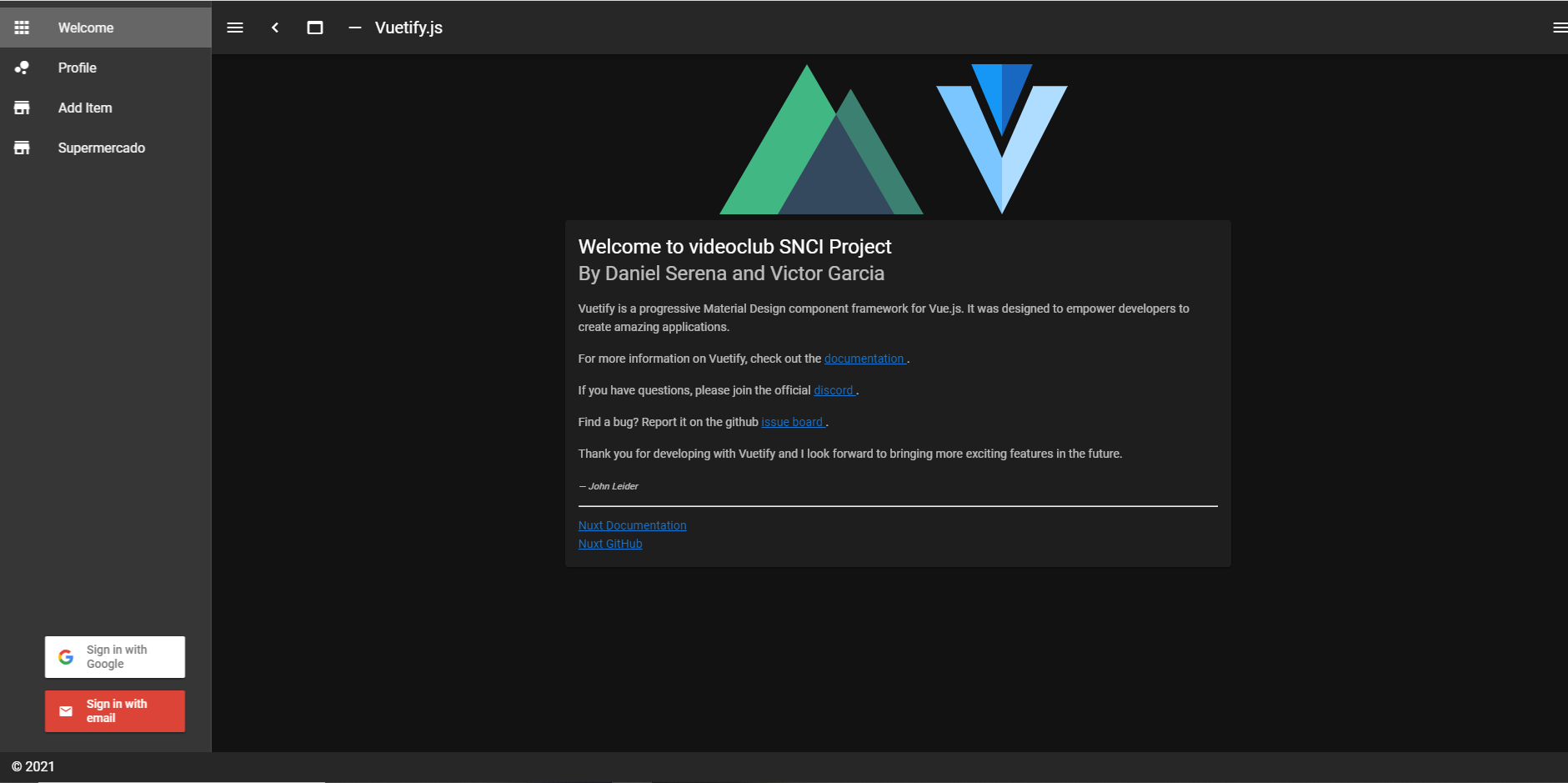
La página principal es una página básica de presentación del equipo y explicación del funcionamiento de la aplicación y de las funciones implementadas.



### Login / Create

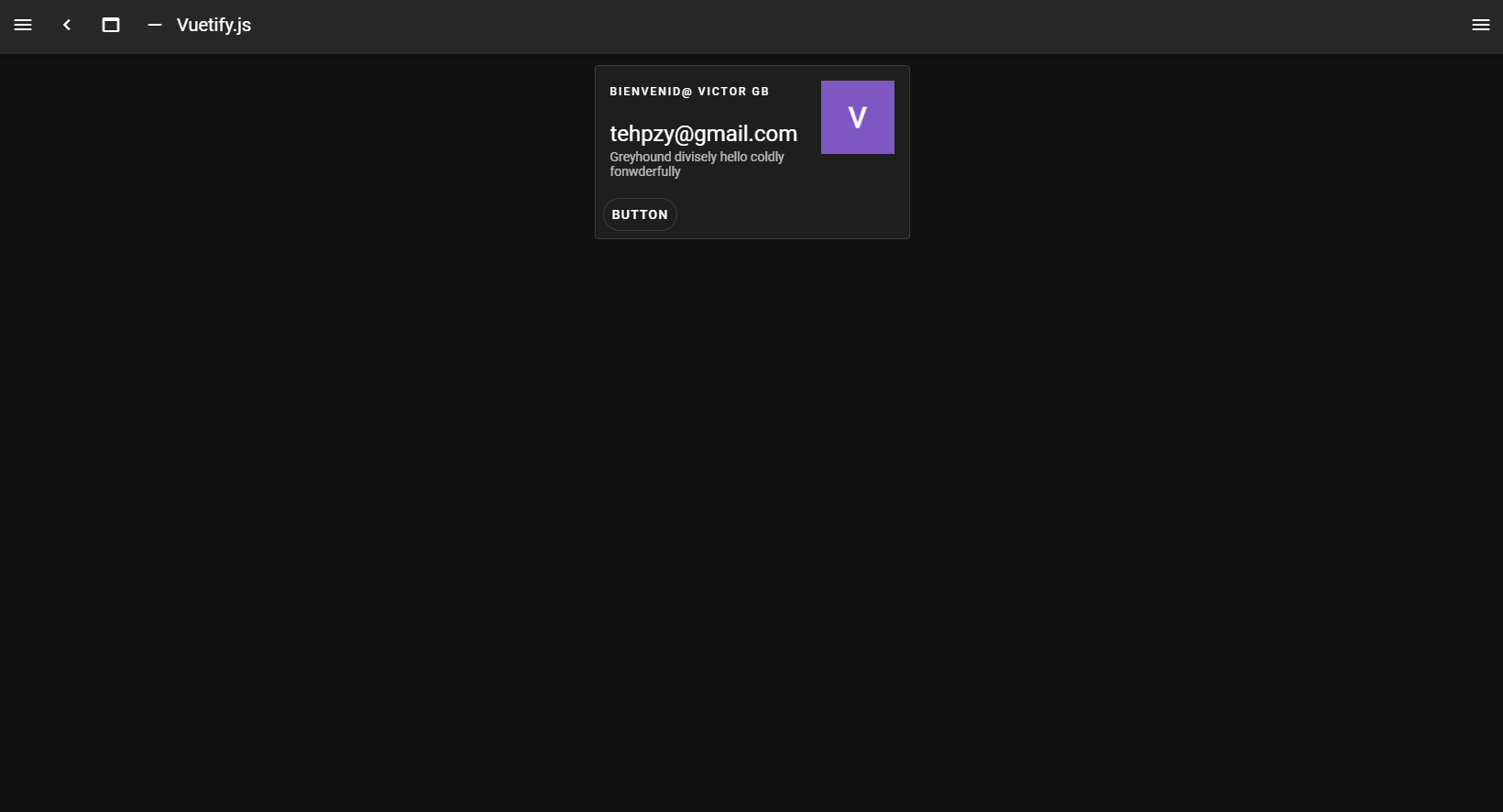
En un principio contábamos con una página de login y otra de create que posteriormente fueron unificadas en login, en caso de no estar registrado en la base de datos se modificaba de forma automática para registrarte y si ya tenias un usuario inicias sesión en la aplicación. Esta implementada mediante la interfaz de firebase, permitiendo incorporar autenticaciones mediante tokens como Google, github y correo-contraseña, entre otros.

Finalmente se ha incorporado en el menú lateral de la aplicación permitiendo una mayor facilidad a la hora de iniciar sesión desde cualquier página.



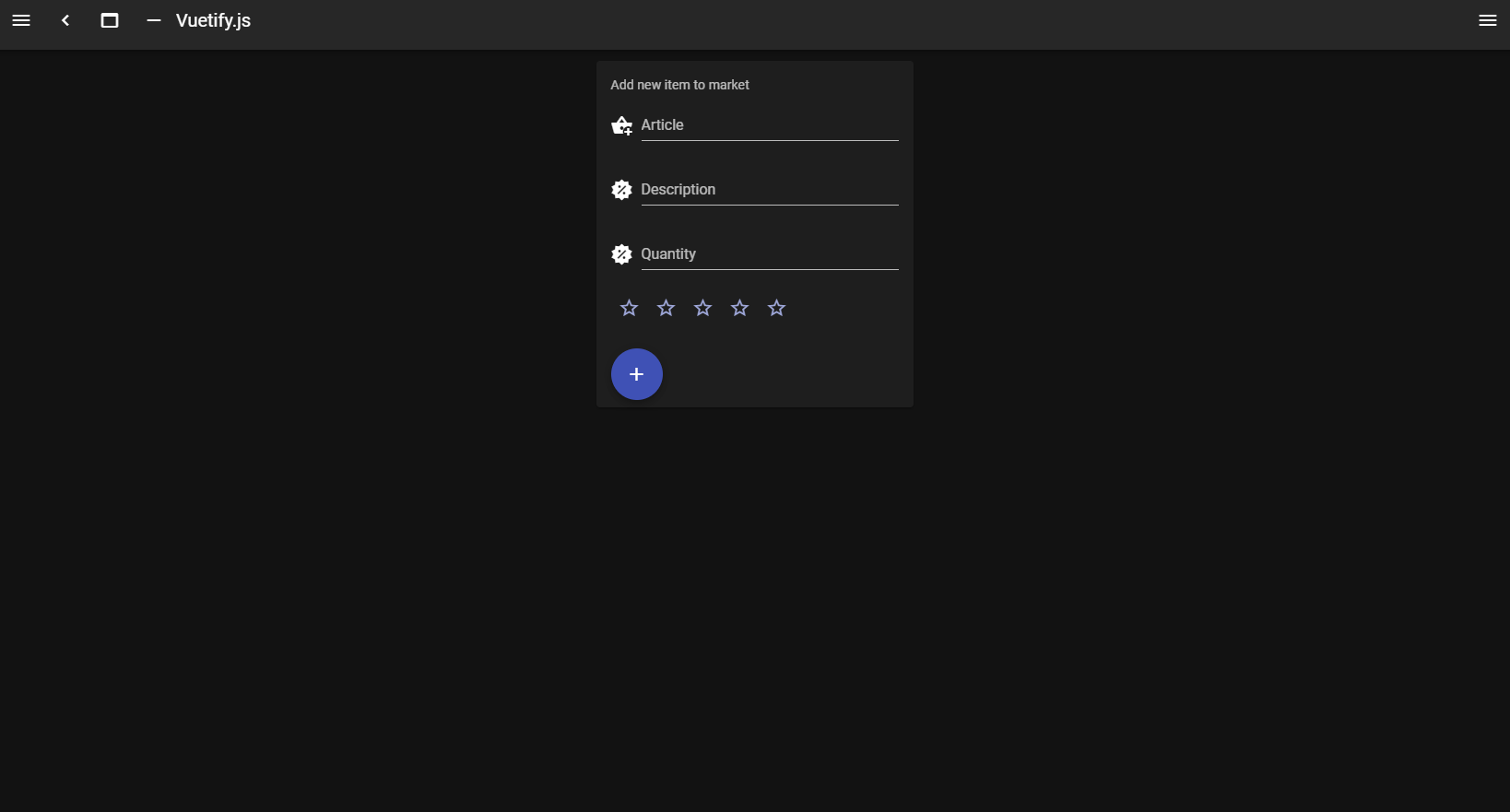
### Profile

En la página de profile se muestra los datos del usuario como su correo o su nombre de cara al resto de usuarios, en caso de iniciar sesión con los proveedores por defecto de firebase se importará una imagen de contacto, si se inicia sesión con email y contraseña, esta imagen será una por defecto.



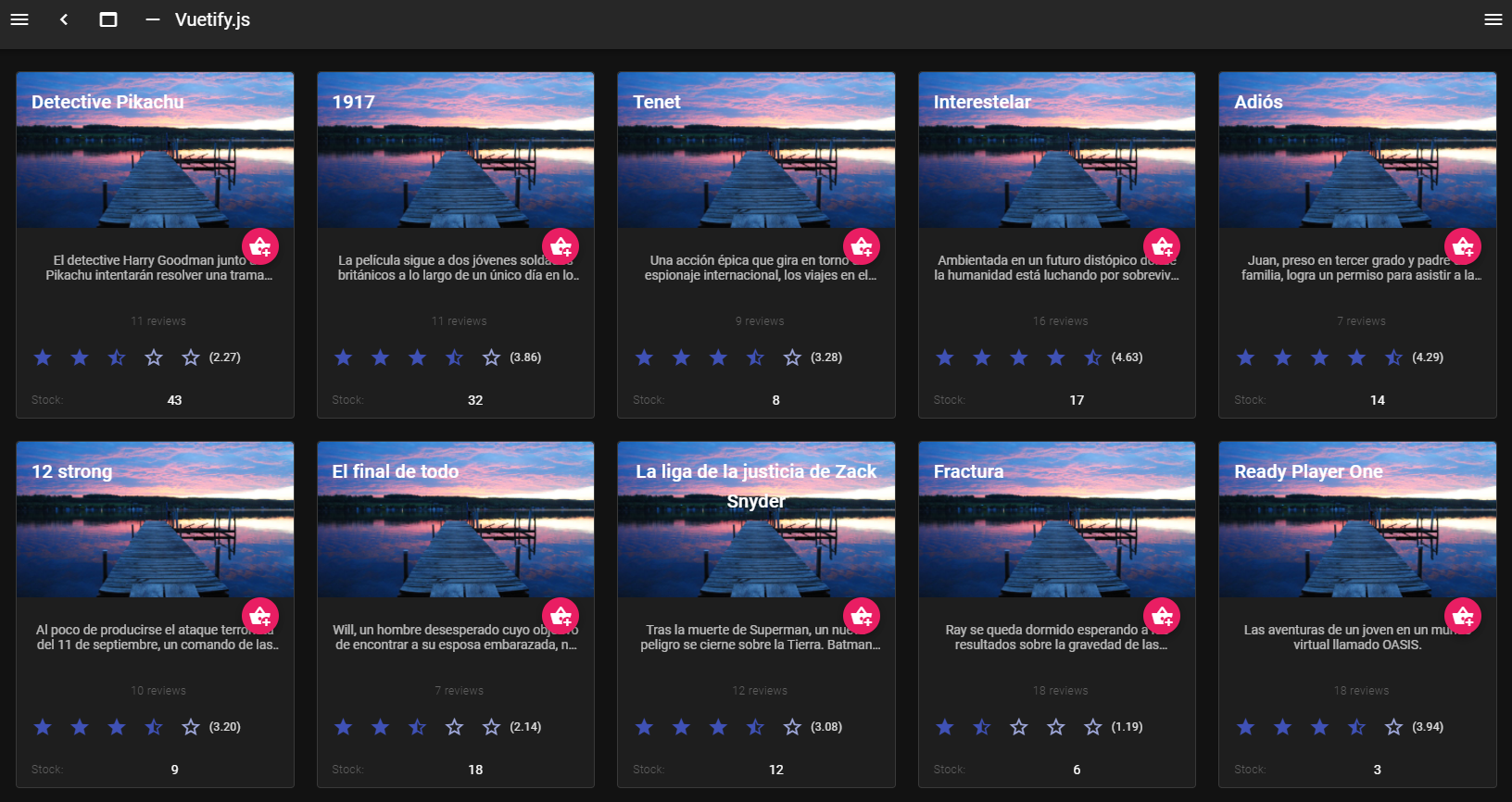
### AddItem

En esta página se muestra un formulario con los campos de la película que se quiere añadir y con un campo de valoración, en este caso se ha incluido para probar su funcionamiento, no obstante, este campo puede ser modificado más adelante por parte de los usuarios. Los campos del formulario tienen reglas de escritura para que no haya problemas de registro a posteriori.



### Supermercado

En la página de supermercado, se muestran todas las peliculas registradas en el sistema. Esta pagina es la mas completa y la que recoge la funcionalidad requerida para el proyecto, como por ejemplo la sincronizacion de datos y la actualizacion de los mismos en tiempo real. En caso de que se añada una pelicula, el usuario que este visualizando esta pagina vera una nueva pelicula al final de la lista, lo mismo ocurre cuando se produce una valoracion de cualquier pelicula, los usuarios que esten visualizando esta pagina veran modificado el valor de dicha pelicula.

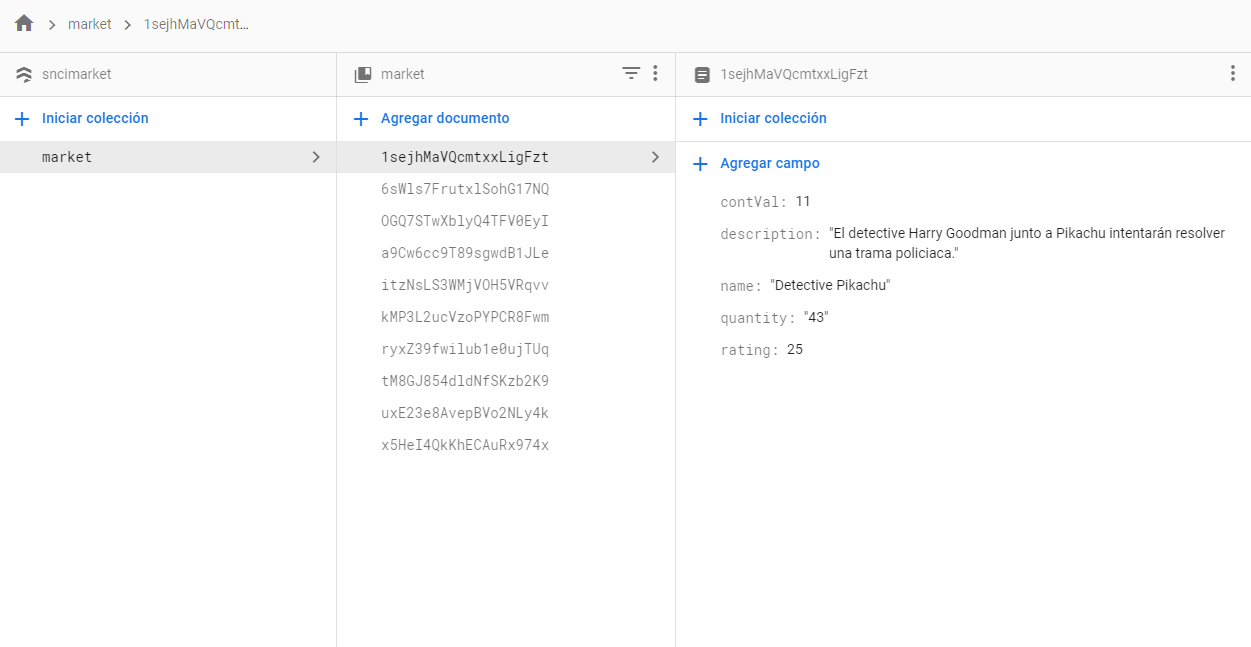


# Base de datos

Para la base de datos, hemos recurrido a una base datos no relacional, en este caso usamos Firebase con el servicio de Firestore. Esta plataforma nos permite hacer una serie de llamadas a la base de datos para añadir, actualizar y previsualizar las diferentes colecciones y documentos.

## Market

Contamos con una colección base para las peliculas, la cual contiene una serie de documentos con los datos de cada una de las peliculas, como se muestra en la siguiente imagen:

En esta colección base se introducen las peliculas que posteriormente se guardan en vuex para su facil acceso, en vuex se incluye como campo el id del propio elemento para facilitar su acceso a firestore y poder modificar facilmente sus valores.

## Usuarios

Por ultimo, contamos con una base de datos o registro de usuario pero en este caso solo en vuex, esto nos permite tener todos los datos del servicio de autenticacion de firebase en vuex y con ello la estructura de usuario que deseemos, como se puede ver a continuacion:

Con esta estructura de usuario podemos acceder al campo que queramos desde cualquier punto de la aplicación, como por ejemplo si el usuario esta logeado o no, estado necesario para proceder al alquiler cuando se implemente en un futuro.

# Conclusión

En esta práctica hemos creado una aplicación web con la herramienta de npm y la tecnología de nuxt. Se han propuesto diversos problemas a paliar y en base a esos problemas se ha generado una aplicación web. Esta, es un servicio de alquiler de películas donde destacan las funciones de crear ítems, valorarlos y alquilarlos. Lo más particular de esta aplicación web es que en el uso de estas funciones, la página se actualiza de forma directa para todos los usuarios ya que cada cliente está suscrito a cambios en la base de datos y cuando suceden se visualizan por cómo funcionan las aplicaciones que utilizan Vuex.

Respecto al trabajo futuro, la página es muy sencilla y se debería perfeccionar algunas funcionalidades o añadir otras como control de precios en los alquileres. Por otro lado, respecto a los problemas planteados en la introducción, sería una buena idea comprobar tiempos de actualización en base a una gran cantidad de usuarios utilizando a la vez la aplicación web.

# Referencias

## Repositorio

En este enlace se muestra el código desarrollado, la metodología de trabajo llevada a cabo así como todos los recursos y requisitos para correr el sistema.

<https://github.com/danielss24/SNCI_VUE>

## Hosting

En este enlace se muestra la aplicación web desplegada.

<https://sncimarket.web.app/>

## Otras referencias

[1] <https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_web>

[2] <https://www.xataka.com/basics/que-es-una-aplicacion-web-progresiva-o-pwa>

[3] <https://nuxtjs.org/>

[4] <https://github.com/nvm-sh/nvm>

[5] <https://firebase.google.com/?hl=es>

[6] <https://firebase.google.com/docs/reference/js>