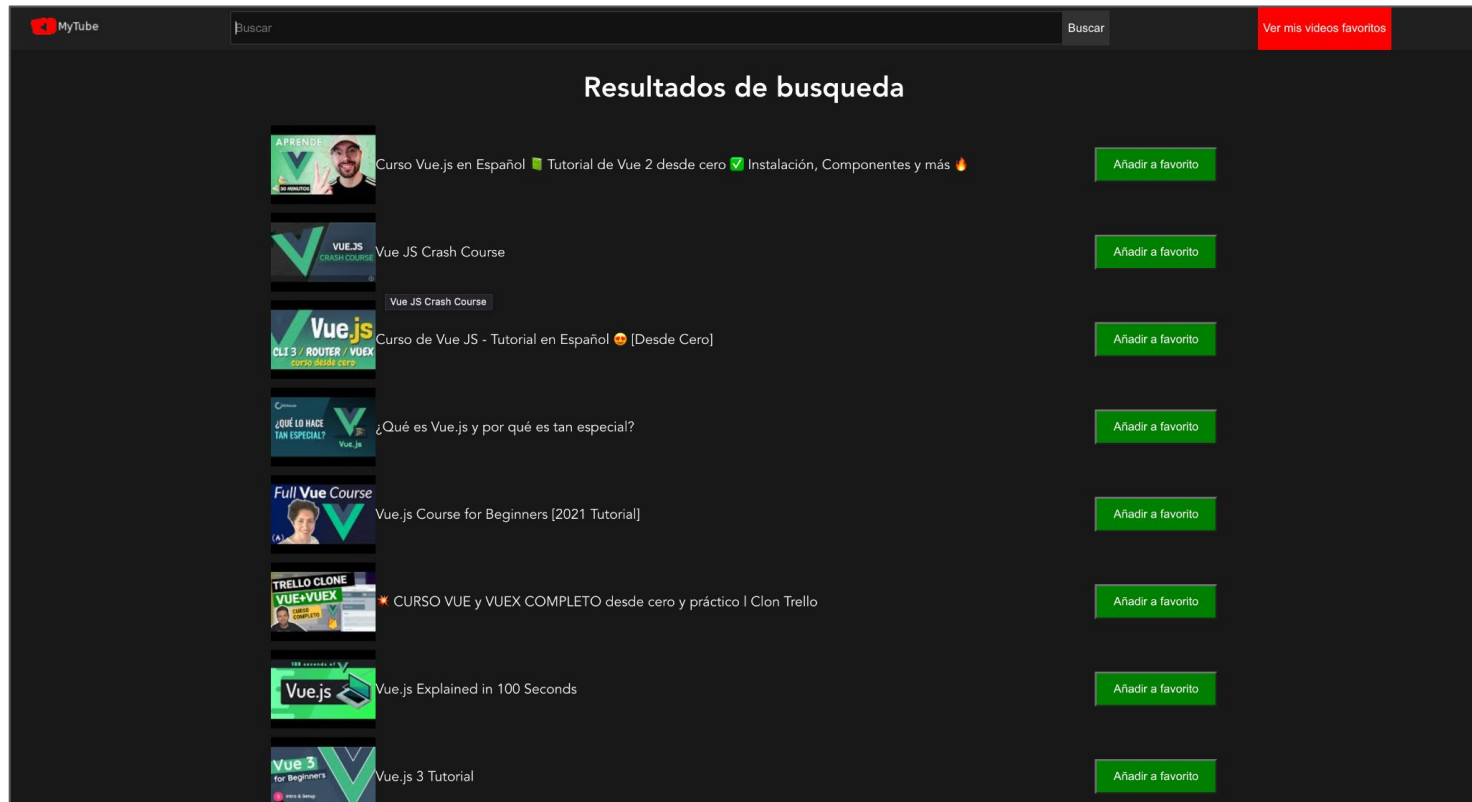




Framework para frontend: Vue.js

Grupo 1:
Joaquín Cáceres
Julián García
Mathias Valdebenito



01

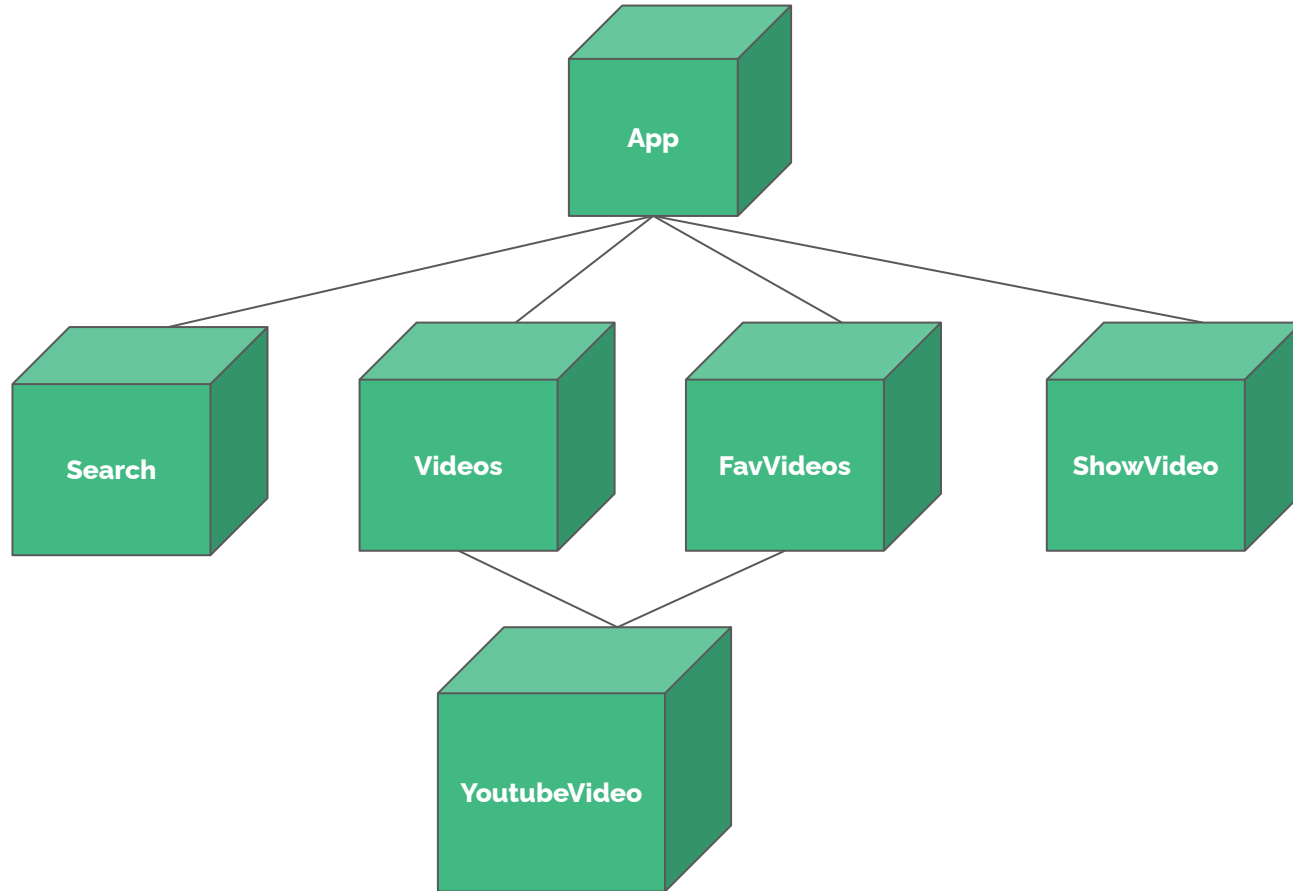
Demo: MyTube



02

Estructura de componentes





03

Análisis del código



```
import { createApp } from 'vue';
import { createStore, Store } from 'vuex';
import { Video, StoreState, Tab } from './StoreState'
import App from './App.vue';

const app = createApp(App);

const store = createStore<StoreState>({
  ...
})

app.use(store);

app.mount('#app');
```

Uso de store con
createStore



```
state: {  
  videos: [],  
  currentVideo: undefined,  
  currentTab: Tab.FAV,  
  favVideos: window.localStorage.getItem('favVideos') !== null  
    ? JSON.parse(window.localStorage.getItem('favVideos')) : [],  
},
```

State

- **videos:** Lista de videos resultado de la búsqueda actual
- **currentVideo:** Video actualmente reproduciendose
- **currentTab:** Pestaña que está siendo mostrada (FAV / VIDEOS)
- **favVideos:** Videos favoritos guardados por el usuario


```
mutations: {
  SET_VIDEOS(state, payload: Array<Video>) {
    state.videos = payload;
  },
  CHANGE_VIDEO(state, payload: string) {
    console.log(payload)
    state.currentVideo = payload;
  },
  ADD_FAV_VIDEO(state, payload: Video) {
    state.favVideos.push(payload);
    window.localStorage.setItem('favVideos', JSON.stringify(state.favVideos));
  },
  DELETE_FAV_VIDEO(state, payload: Video) {
    state.favVideos = state.favVideos.filter((video) => video.id.videoId !== payload.id.videoId)
    window.localStorage.setItem('favVideos', JSON.stringify(state.favVideos));
  },
  CHANGE_TAB(state, payload: Tab) {
    state.currentTab = payload
  },
}
```

Mutations

- **SET_VIDEOS:** permite guardar los videos obtenidos en una búsqueda
- **CHANGE_VIDEO:** permite cambiar el video que actualmente se reproduce
- **ADD_FAV_VIDEO:** permite añadir videos a la lista de videos favoritos
- **DELETE_FAV_VIDEO:** permite eliminar videos de la lista de videos favoritos
- **CHANGE_TAB:** permite cambiar la pestaña actual que se está mostrando

```
actions: {
  async searchYoutube({ commit }, query: string) {
    const result = await fetch(`https://www.googleapis.com/youtube/v3/search?q=${query}&
part=snippet&maxResults=50&key=AIzaSyA-tvEokrKF-vdJuqA-MXucQclYYiivAXI`)
    const response = await result.json();
    commit("SET_VIDEOS", response.items);
    commit("CHANGE_TAB", Tab.VIDEOS);
    commit("CHANGE_VIDEO", undefined);
  },
  back({commit}) {
    commit("SET_VIDEOS", []);
    commit("CHANGE_TAB", Tab.FAV);
    commit("CHANGE_VIDEO", undefined);
  },
  showVideo({commit}, videoId: string) {
    commit("CHANGE_VIDEO", videoId);
  },
  addFavVideo({commit}, video: Video) {
    commit("ADD_FAV_VIDEO", video);
  },
  deleteFavVideo({commit}, video: Video) {
    commit("DELETE_FAV_VIDEO", video);
  }
},
```

Actions

Vuex nos sirvió mucho para manejar contexto sin tener que pasar props



```
<script setup lang="ts">
import Search from "../components/Search.vue"
import Videos from "../components/Videos.vue"
import ShowVideo from "../components/ShowVideo.vue"
import { reactive, ref, computed } from 'vue'
import { useStore } from "vuex";
import { StoreState, Tab } from "../StoreState";
import FavVideos from "../components/FavVideos.vue";
const store = useStore<StoreState>();
const currentVideo = computed(() => {
  return store.state.currentVideo
})
const currentTab = computed(() => {
  return store.state.currentTab
})
</script>

<template>
<Search />
<div class="centered">
  <Videos v-if="currentVideo === undefined && currentTab === Tab.VIDEOS" />
  <FavVideos v-if="currentVideo === undefined && currentTab === Tab.FAV" />
  <ShowVideo v-if="currentVideo !== undefined" />
</div>
</template>
```

Barra de navegación y
búsqueda

1 de entre la lista de videos
buscados, la lista de videos
favoritos o el video que se
reproduce actualmente

Componente Search

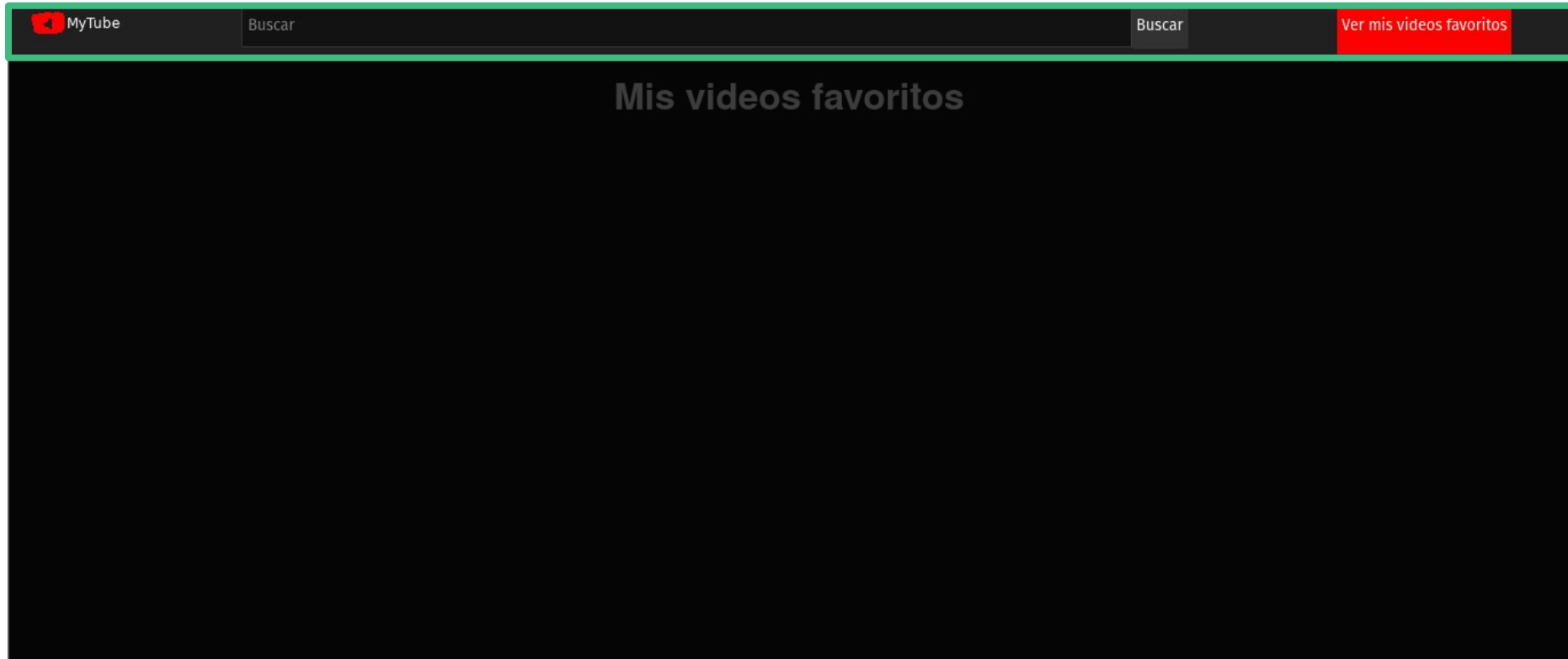
```
<script setup lang="ts">
  import { ref } from "vue";
  import { useStore, mapActions } from "vuex";
  import { StoreState, Tab } from "../StoreState";
  const searchQuery = ref("");
  const store = useStore<StoreState>();
  function search() {
    if(searchQuery.value) {
      store.dispatch('searchYoutube', searchQuery.value);
      searchQuery.value = ''
    }
  }
  function back() {
    store.dispatch('back');
  }
</script>
<template>
<div class="search-bar flex">
  <div class="left">
    <button @click="back" class="left-button">
      
    </button>
  </div>
  <div class="right flex">
    <div class="search-input" >
      <input v-model="searchQuery" class="no-display" @keyup.enter="search" placeholder="Buscar" />
    </div>
    <button @click="search" class="search-button">Buscar</button>
  </div>
  <div class="flex">
    <button @click="back" class="fav-button">Ver mis videos favoritos</button>
  </div>
</div>
</template>
```

Uso de dispatch para
llamar acciones
definidas en el store

Uso de ref y v-model
para manejar el
cambio de estado
automáticamente al
cambiar el input

Uso de listeners con @
para hacer el llamado
de la función search

Componente Search



```
<script setup lang="ts">
  import { computed } from "vue"
  import { useStore } from "vuex";
  import { StoreState } from "../StoreState";
  import YoutubeVideo from "../YoutubeVideo.vue";

  const store = useStore<StoreState>();
  const videos = computed(() => {
    return store.state.videos;
  })
</script>
<template>
<div class="w-full">
  <h1 class="title">Resultados de busqueda</h1>
  <YoutubeVideo v-for="video in videos" :video="video" />
</div>
</template>

<style scoped>
div.w-full {
  width: 100%;
  overflow-wrap: break-word;
}

h1.title {
  color: white;
  width: 100%;
  text-align: center;
  display: block;
}
</style>
```

Uso de v-for para iterar por todos los videos encontrados en la búsqueda y mostrar cada uno con el componente YoutubeVideo

 MyTube

one piece episodio 1015

Buscar

Ver mis videos favoritos

Resultados de busqueda



One Piece Capítulo 1015 Sub Español Completo

Eliminar de favoritos



El Sueño de Luffy y Roger One Piece Episodio 1015

Añadir a favorito



One Piece Episode 1015/6 English Sub HD1080

Añadir a favorito



Luffy golpea 🍷 a Kaido- One piece Cap 1015 - SUB ESP

Añadir a favorito



¡¡UNA OBRA MAESTRA DE LA ANIMACIÓN!! | ONE PIECE 1015 | ANÁLISIS Y REVIEW.

Añadir a favorito



```
<script setup lang="ts">
  import { computed } from "vue"
  import { useStore } from "vuex";
  import { StoreState } from "../StoreState";
  import YoutubeVideo from "../YoutubeVideo.vue";

  const store = useStore<StoreState>();
  const favVideos = computed(() => {
    return store.state.favVideos;
  })
</script>
<template>
<div class="w-full">
  <h1 class="title">Mis videos favoritos</h1>
  <YoutubeVideo v-for="video in favVideos" :video="video" />
</div>
</template>

<style scoped>
div.w-full {
  width: 100%;
  overflow-wrap: break-word;
}

h1.title {
  color: white;
  width: 100%;
  text-align: center;
  display: block;
}
</style>
```

Igual que el componente anterior pero cambiando videos por favVideos


 MyTube

Buscar

Buscar


Ver mis videos favoritos

Mis videos favoritos




One Piece Opening 13 "One Day" 1080p [Creditless!!!]

Eliminar de favoritos



😭😭 Naruto Shippuden Jiraiya vs Pain(Jiraiya Muere) SUB ESPAÑOL HD 😭😭

Eliminar de favoritos



JUJUTSU KAISEN - Opening | Kaikai Kitan

Eliminar de favoritos

```
<script setup lang="ts">
  import { computed } from 'vue';
  import { useStore } from 'vuex';
  import { StoreState, Video, Tab } from '../StoreState';
  const props = defineProps<{video: Video}>();
  const store = useStore<StoreState>();
  function showVideo() {
    store.dispatch('showVideo', props.video.id.videoId);
  }
  function addFavVideo(e: MouseEvent) {
    e.stopPropagation()
    store.dispatch('addFavVideo', props.video);
  }
  function deleteFavVideo(e: MouseEvent) {
    e.stopPropagation()
    store.dispatch('deleteFavVideo', props.video);
  }
  const favVideos = computed(() => {
    return store.state.favVideos;
  })
</script>
```

Agregar video a los
favoritos

Eliminar video de los
favoritos

Componente YoutubeVideo

```
<template>
  <div class="flex youtube-thumbnail" @click="showVideo" :title="props.video.snippet.title">
    
    <div class="video-title">{{props.video.snippet.title}}</div>
    <button class='add-fav'
      v-if="!favVideos.map(video => video.id.videoId).includes(props.video.id.videoId)"
      @click="addFavVideo"
    > Añadir a favorito</button>
    <button class='delete-fav'
      v-else @click="deleteFavVideo"
    > Eliminar de favoritos</button>
  </div>
</template>
```

Uso de :property para pasarle datos de nuestro objeto Video a los elementos html.

Uso de v-if con v-else para mostrar diferentes botones dependiendo de si el video ya está en favoritos

 MyTube

one piece episodio 1015

Buscar

Ver mis videos favoritos

Resultados de búsqueda



One Piece Capítulo 1015 Sub Español Completo

Eliminar de favoritos



El Sueño de Luffy y Roger One Piece Episodio 1015

Añadir a favorito



One Piece Episode 1015/6 English Sub HD1080

Añadir a favorito



Luffy golpea 🍷 a Kaido- One piece Cap 1015 - SUB ESP

Añadir a favorito



¡¡UNA OBRA MAESTRA DE LA ANIMACIÓN!! | ONE PIECE 1015 | ANÁLISIS Y REVIEW.

Añadir a favorito

Componente ShowVideo

```
<script setup lang="ts">
  import { computed } from "vue"
  import { useStore } from "vuex";
  import { StoreState } from "../StoreState";

  const store = useStore<StoreState>();
  const currentVideo = computed(() => {
    return store.state.currentVideo;
  })
</script>
<template>
<div class="w-full">
  <div class="yt-container">
    <iframe width="640" height="360" :src="`https://www.youtube.com/embed/${currentVideo}?autoplay=1`"
    title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe>
  </div>
</div>
</template>

<style scoped>
div.w-full {
  margin-top: 20px;
  width: 100%;
  display: flex;
}
div.yt-container {
  margin: 0 auto;
}
</style>
```

Iframe con video de youtube especificado en la propiedad currentVideo del estado (por id)

Componente ShowVideo



04

Aprendizajes



- Fácil desarrollar una aplicación simple en Vue
- Typescript ayuda mucho a no equivocarse
- Rapidez para montar el servidor en desarrollo.
- Menos engorroso que react, pero menor flexibilidad (no se puede especificar las dependencias)

