



A <PROGRAMAÇÃO> PRECISA DE TI /

Laboratório Web

Web

Front-end Frameworks

- Desenvolvimento web front-end descreve o processo de transformar dados numa interface gráfica, através do uso de CSS, HTML e JavaScript de forma que os utilizadores possam observar e interagir com esses dados.
- Enquanto frameworks de backend (e.g., Django) ajudam no desenvolvimento de lógica e funcionalidade, frameworks de front-end como Vue e React ajudam no desenvolvimento de acessibilidade do utilizador.

Front-end Frameworks

- Uma framework é um produto de software que simplifica o desenvolvimento e a manutenção de projetos complexos.
- As estruturas contêm módulos de software básicos com bocados de código prontos para a serem utilizados
- As frameworks ditam as regras para construir a arquitetura do sistema: um esqueleto que deve ser estendido e alterado de acordo com os requisitos especificados

10 Frontend Frameworks

- Vue.js
- React
- Angular
- JQuery
- Backbone.js
- Ember.js
- Semantic-UI
- Svelte
- Foundation
- Preact

Vue.js

- Fácil de aprender — a sintaxe do framework é comprehensível até para iniciantes; conhecimento prévio de JavaScript, HTML, CSS é suficiente.
- Alto desempenho — devido à sua simplicidade e tamanho pequeno, o Vue é uma ferramenta incrivelmente rápida e escalável.
- Integrações de terceiros — O Vue pode se integrar facilmente a outras bibliotecas e projetos existentes.
- DOM virtual — uma abstração do DOM tradicional é um recurso padrão
- Documentação detalhada

Vue.js

- O Vue é uma estrutura progressiva de código aberto destinada a ser usada de forma incremental.
- Ao contrário do React (Facebook) e do Angular (Google), o VueJS é desenvolvido e financiado exclusivamente a partir de doações dos utilizadores
- VueJs é descrita como uma estrutura padronizada MVVM, Model-View View-Model, que é baseada no conceito de ligação bidirecional de dados.

Vue.js

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Printing hello world</title>
</head>
<body>
    <div id="app">
        <h1>{{ message }}</h1>
    </div>
<script src="https://cdn.jsdelivr.net/npm/vue@2.6.14/dist/vue.js"></script> ←
<script>
    new Vue({
        el: "#app",
        data: {
            message: "Hello World From Vue"
        }
    });
</script>
</body>
</html>
```

Usar a Framework Vue.js

- Para começar a usar o Vue.js, adicione o seguinte script a sua nova página HTML

```
<script src="https://cdn.jsdelivr.net/npm/vue@2.6.0"></script>
<script src="https://unpkg.com/vue-router@2.0.0"></script>
```

Manipulando o input do utilizador

HTML

```
<div id="app">  
  {{ message }}  
</div>
```

JS

```
new Vue({  
  el: '#app',  
  data: {  
    message: 'Hello Vue.js!'  
  }  
})
```

Manipulando o input do utilizador

HTML

```
<div id="app">  
  {{ message }}  
<input v-model="message">  
</div>
```

JS

```
new Vue({  
  el: '#app',  
  data: {  
    message: 'Hello Vue.js!'  
  }  
})
```

Componentes

- Props - propriedades. Registrado numa componente para passar dados da componente pai para uma componentes filha.
- Data
- Methods
- Computed - lógica complexa que inclui dados reativos

Exemplo - usando apenas Data

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <script src="https://cdn.jsdelivr.net/npm/vue@2.6.0"></script>
  <script src="https://unpkg.com/vue-router@2.0.0"></script>
  <title>Document</title>
</head>
<body>
  <div id="app">
    <button @click="count++>Count is: {{ count }}</button>
  </div>
  <script>
    var vm = new Vue({
      el: "#app",
      data: {
        count: 0,
      },
    });
  </script>
</body>
</html>
```

Exemplo - usando Methods

```
<body>
  <div id="app">
    <button @click="increment()">
      Count is: {{ count }}
    </button>
  </div>
  <script>
    var vm = new Vue({
      el: "#app",
      data: {
        count: 0,
      },
      methods: {
        increment() {
          this.count++;
        },
      },
    });
  </script>
</body>
```

Exemplo - input

Text

HTML

```
<input v-model="message" placeholder="edit me">
<p>Message is: {{ message }}</p>
```

edit me

Message is:

Exemplo - checkbox

Multiple checkboxes, bound to the same Array:

HTML

```
<input type="checkbox" id="jack" value="Jack" v-model="checkedNames">
<label for="jack">Jack</label>
<input type="checkbox" id="john" value="John" v-model="checkedNames">
<label for="john">John</label>
<input type="checkbox" id="mike" value="Mike" v-model="checkedNames">
<label for="mike">Mike</label>
<br>
<span>Checked names: {{ checkedNames }}</span>
```

JS

```
new Vue({
  el: '...',
  data: {
    checkedNames: []
  }
})
```

Jack John Mike

Checked names: ["Jack", "John"]

Exemplo - radio

Radio

HTML

```
<input type="radio" id="one" value="One" v-model="picked">
<label for="one">One</label>
<br>
<input type="radio" id="two" value="Two" v-model="picked">
<label for="two">Two</label>
<br>
<span>Picked: {{ picked }}</span>
```

- One
 - Two
- Picked:

Exemplo – select box

Single select:

```
HTML
<select v-model="selected">
  <option disabled value="">Please select one</option>
  <option>A</option>
  <option>B</option>
  <option>C</option>
</select>
<span>Selected: {{ selected }}</span>
```

```
JS
new Vue({
  el: '...',
  data: {
    selected: ''
  }
})
```

B ▼ Selected: B

Exemplo – select – listas/arrays

```
HTML  
<ul id="example-2">  
  <li v-for="(item, index) in items">  
    {{ parentMessage }} - {{ index }} - {{ item.message }}  
  </li>  
</ul>
```

```
JS  
var example2 = new Vue({  
  el: '#example-2',  
  data: {  
    parentMessage: 'Parent',  
    items: [  
      { message: 'Foo' },  
      { message: 'Bar' }  
    ]  
  }  
})
```

Result:

- Parent - 0 - Foo
- Parent - 1 - Bar

Exemplo – Iterações

```
<ul>
  <li v-for="n in countdown">{{n}}</li>
</ul>
```

- 1
- 2
- 3
- 4
- 5

Here `countdown` is an array defined inside the `data` property of our Vue instance.

```
var app = new Vue({
  el: '#app',
  data: {
    countdown: [5, 4, 3, 2, 1]
  }
});
```

Copy

Exemplo – Iterações c/ posição

```
<ul>
  <li v-for="(fruit, i) in fruits">{{i}} - {{fruit}}</li>
</ul>
```

```
var app = new Vue({
  el: '#app',
  data: {
    fruits: ["Apple", "Banana", "Orange", "Mango", "Grapes"]
  }
});
```

- 0 - Apple
- 1 - Banana
- 2 - Orange
- 3 - Mango
- 4 - Grapes

Exemplo – Iterações - propriedades de um objecto

```
<ul>
  <li v-for="(value, key, index) in profile">{{key}} - {{value}}</li>
</ul>
```

```
var app = new Vue({
  el: '#app',
  data: {
    profile : {
      name : 'Tushar',
      age : '31',
      degree : 'Masters',
      position : 'Full Stack Developer'
    }
  }
});
```

- **name** - Tushar
- **age** - 31
- **degree** - Masters
- **position** - Full Stack Developer

Exemplo – Condições

```
var app = new Vue({  
  el: '#app',  
  data: {  
    count: 10  
  }  
});
```

```
<div id="app">  
  <div v-if="count > 10">  
    Count is greater than 10  
  </div>  
  <div v-else-if="count == 10">  
    Count is 10  
  </div>  
  <div v-else>  
    Count is less than 10  
  </div>  
</div>
```

Exemplo – Showing/hiding

```
<h3> Using v-show directive</h3>
<div v-show="count > 10">
    Count is greater than 10
</div>
<div v-show="count == 10">
    Count is 10
</div>
<div v-show="count < 10">
    Count is less than 10
</div>
```

Exemplo – Eventos

```
<div id="app">
  <h2>{{count}}</h2>
  <button v-on:click="count += 1">Increase</button>
  <button v-on:click="count -= 1">Decrease</button>
</div>
```

Exemplo – Métodos/funções

```
<div id="app">
  <h2>Greet</h2>
  <button v-on:click="greet('Hello')">Say Hello</button>
  <button v-on:click="greet('Hi')">Say Hi</button>
  <button v-on:click="greet('Namaste')">Say Namaste</button>
  <br/><br/>
  {{greeting}}
</div>
```

Greet

Say Hello Say Hi Say Namaste

Namaste, How Are You?

```
var app = new Vue({
  el: '#app',
  data: {
    greeting: '',
  },
  methods: {
    greet: function(greeting){
      this.greeting = greeting + ", How Are You?";
    }
  });
});
```

Exemplo – Computed

```
<div id="app">  
  <label>Type your name: </label>  
  <input type="text" v-model="name"/> <br/><br/>  
  Your name is {{name}}<br/>  
  And is {{charcount}} characters long.  
</div>
```

Type your name: Tushar

Your name is Tushar
And is 6 characters long.

```
var app = new Vue({  
  el: '#app',  
  data: {  
    name: '',  
  },  
  computed: {  
    charcount: function () {  
      return this.name.length;  
    }  
  }  
});
```

Exercício 1

Contact US

Name

Your name

Email address

name@example.com

Gender

Male Female

From where did you hear about us?

You are a :

Developer
Graphic Designer
Manager
Company Head / CEO
Other

Which of our service are you interested in?

Online Tests Paper Based Tests Customized Tests

Your message

How satisfied are you with our service?

Agree to Terms & Conditions

Submit

- **name:** John Doe
- **email:** john@doe.com
- **gender:** male
- **refer:** newspaper
- **profession:** ["manager", "ceo"]
- **message:** Please reply back asap
- **satisfaction:** 8
- **interested:** ["customized", "online"]
- **terms:** true

Exercício 2

Add

```
addItem: function () {
    this.itemsList.push({
        quantity: quantityIN,
        itemName: itemNameIN,
    });
},
```

Remove

```
removeItem: function (index) {
    this.itemsList.splice(index, 1);
},
```

Shopping List		
Quantity	Item	Actions
3	Apples	<button>Edit</button> <button>Delete</button>
6	Pears	<button>Edit</button> <button>Delete</button>
Add new item		
Quantity	<input type="text"/>	Name <input type="text"/>
	<button>Add</button>	

Setting Vue

```
npm install -g @vue/cli
```

Criar um projeto -> vue create nomeProjeto

Setting Vue

Vue CLI v4.3.1

Failed to check for updates

? Please pick a preset:

 default (babel, eslint)

> Manually select features

Setting Vue

Vue CLI v4.3.1

Failed to check for updates

? Please pick a preset: Manually select features

? Check the features needed for your project: (Press <space> to select, <a> to toggle all, <i> to invert selection)

>● Babel

- TypeScript
- Progressive Web App (PWA) Support
- Router
- Vuex
- CSS Pre-processors
- Linter / Formatter
- Unit Testing
- E2E Testing

Setting Vue

```
Vue CLI v4.3.1
Failed to check for updates
? Please pick a preset: Manually select features
? Check the features needed for your project: Babel, Router, Linter
? Use history mode for router? (Requires proper server setup for index fallback in production) (Y/n) █
```

Setting Vue

Vue CLI v4.3.1

Failed to check for updates

- ? Please pick a preset: Manually select features
- ? Check the features needed for your project: Babel, Router, Linter
- ? Use history mode for router? (Requires proper server setup for index fallback in production) No
- ? Pick a linter / formatter config: (Use arrow keys)
 - > ESLint with error prevention only
 - ESLint + Airbnb config
 - ESLint + Standard config
 - ESLint + Prettier

Setting Vue

Vue CLI v4.3.1

```
Failed to check for updates
? Please pick a preset: Manually select features
? Check the features needed for your project: Babel, Router, Linter
? Use history mode for router? (Requires proper server setup for index fallback in production) No
? Pick a linter / formatter config: Basic
? Pick additional lint features: (Press <space> to select, <a> to toggle all, <i> to invert selection)
❯◉ Lint on save
  ○ Lint and fix on commit
```

Setting Vue

Vue CLI v4.3.1

Failed to check for updates

- ? Please pick a preset: Manually select features
- ? Check the features needed for your project: Babel, Router, Linter
- ? Use history mode for router? (Requires proper server setup for index fallback in production) No
- ? Pick a linter / formatter config: Basic
- ? Pick additional lint features: (Press `<space>` to select, `<a>` to toggle all, `<i>` to invert selection)
 - Lint on save
 - Lint and fix on commit

Setting Vue

Vue CLI v4.3.1

```
Failed to check for updates
? Please pick a preset: Manually select features
? Check the features needed for your project: Babel, Router, Linter
? Use history mode for router? (Requires proper server setup for index fallback in production) No
? Pick a linter / formatter config: Basic
? Pick additional lint features: Lint on save
? Where do you prefer placing config for Babel, ESLint, etc.? (Use arrow keys)
> In dedicated config files
In package.json
```

Setting Vue

Vue CLI v4.3.1

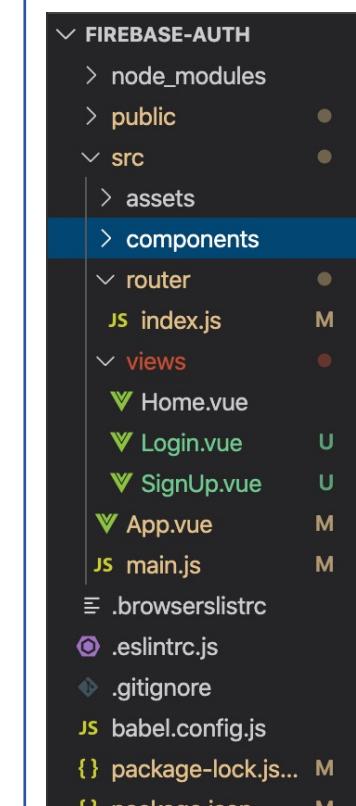
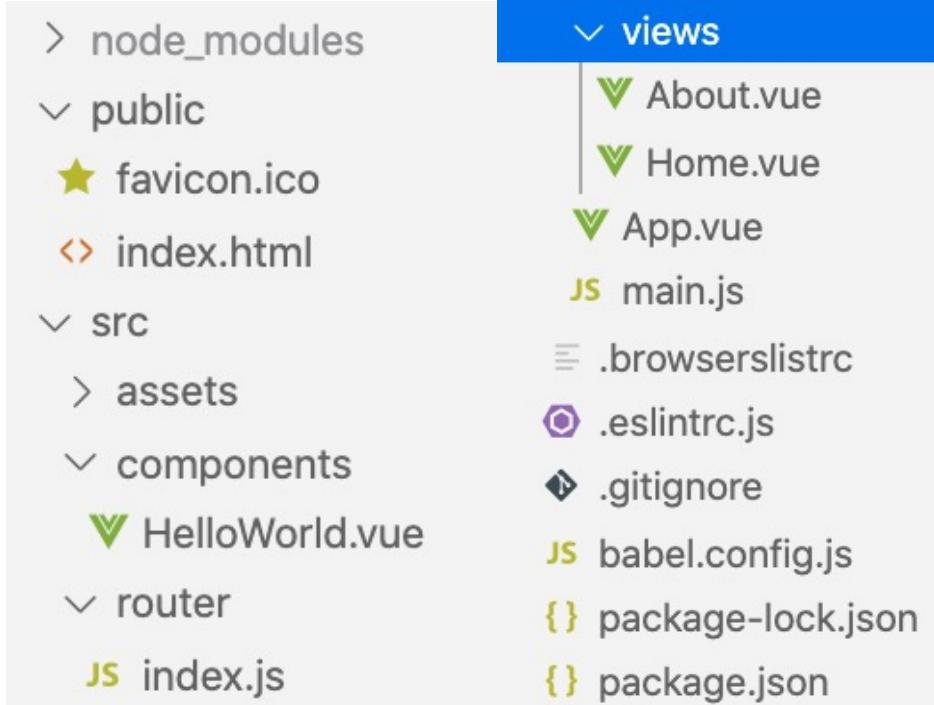
Failed to check for updates

- ? Please pick a preset: Manually select features
- ? Check the features needed for your project: Babel, Router, Linter
- ? Use history mode for router? (Requires proper server setup for index fallback in production) No
- ? Pick a linter / formatter config: Basic
- ? Pick additional lint features: Lint on save
- ? Where do you prefer placing config for Babel, ESLint, etc.? In dedicated config files
- ? Save this as a preset for future projects? (y/N) N

Projeto Vue

<https://vuejs.org/#rule-categories>

O projeto criado por Default tem a seguinte estrutura:



Acrescentar nas views, uma página para o Login e outra para o Registo. Deverá copiar os seus formulários de login e registo para as respetivas páginas.

A estrutura das novas páginas, deve replicar a página Home (criada por default). Ver slide seguinte

Registo e Login

- Adicionar ao script as variáveis email e password

```
64      </template>
65      <script>
66
67      export default {
68          name: "Signup",
69          data() {
70              return {
71                  email: "",
72                  password: "",
```

Projeto Vue

```
▼ Home.vue ● ▼ Login.vue

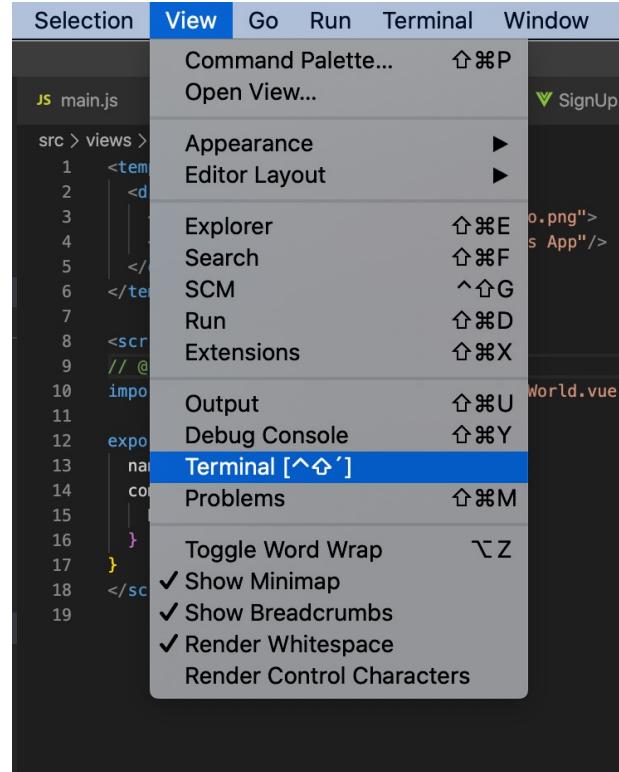
src > views > ▼ Home.vue > {} "Home.vue"
1  <template>
2  | <div class="home">
3  |   
4  |   <HelloWorld msg="Welcome to Your Vue.js App"/>
5  | </div>
6  </template>
7
8  <style>
9  |   body {
10 |     background-color: #cecece;
11 |   }
12 </style>
13
14 <script>
15 // @ is an alias to /src
16 import HelloWorld from "@/components/HelloWorld.vue";
17
18 export default {
19   name: "Home",
20   components: {
21     HelloWorld,
22   },
23 };
</script>
25
```

```
▼ Home.vue ● ▼ Login.vue ● ▼ SignUp.vue ×

src > views > ▼ SignUp.vue > {} "SignUp.vue" > template > div.row > div.col-sm-5.m-auto > form#sign
1  <template>
2  | <div class="row">
3  |   <div class="col-12 text-center mb-4">
4  |     <h1>Sign Up</h1>
5  |   </div>
6  |   <div class="col-sm-5 m-auto">
7  |     <div v-if="errorMessage !== ''" class="alert alert-danger" role="alert">
8  |       {{ errorMessage }}
9  |     </div>
10 |     <div>
11 |       <input type="text" v-model="email" placeholder="Email" />
12 |       <input type="password" v-model="password" placeholder="Password" />
13 |     </div>
14 |     <button type="button" @click="signupRequest()" class="btn btn-primary w-100">Sign Up</button>
15 |   </div>
16 </div>
17
18 <script>
19   export default {
20     name: "Signup",
21     data() {
22       return {
23         email: "",
24         password: "",
25       };
26     },
27     methods: {
28       signupRequest() {
29         const user = {
30           email: this.email,
31           password: this.password,
32         };
33         axios
34           .post("https://jsonplaceholder.typicode.com/users", user)
35           .then((response) => {
36             console.log(response);
37           })
38           .catch((error) => {
39             this.errorMessage = error.message;
40           });
41       },
42     },
43   };
44 </script>
45
46 <style>
47   .alert {
48     margin-bottom: 1rem;
49   }
50   .form-group {
51     margin-bottom: 1.5rem;
52   }
53   .form-control {
54     border-radius: 0.25rem;
55   }
56   .form-control:focus {
57     border-color: #007bff;
58     outline: none;
59   }
60   .btn-primary {
61     border-color: #007bff;
62     background-color: #007bff;
63     color: white;
64   }
65   .btn-primary:hover {
66     border-color: #007bff;
67     background-color: #007bff;
68     color: white;
69   }
70   .btn-primary:active {
71     border-color: #007bff;
72     background-color: #007bff;
73     color: white;
74   }
75   .btn-primary:disabled {
76     border-color: #007bff;
77     background-color: #007bff;
78     color: white;
79   }
80   .btn-primary:disabled:active {
81     border-color: #007bff;
82     background-color: #007bff;
83     color: white;
84   }
85   .w-100 {
86     width: 100% !important;
87   }
88 </style>
```

Testar o projeto vue

Para testar o projeto primeiro abra o terminal

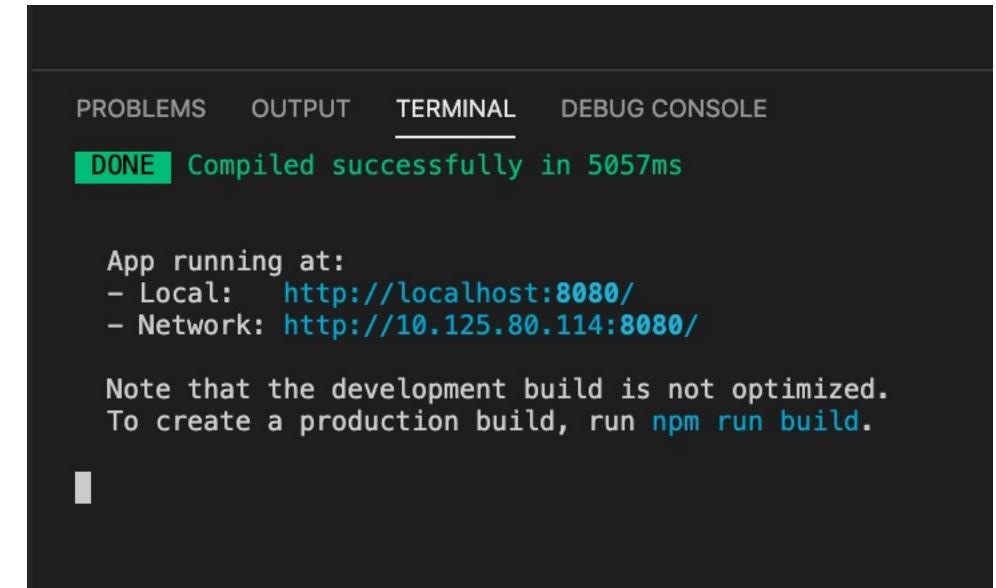


Testar o projeto vue

Guarde o projeto e as alterações e escreva no terminal o comando: **npm run serve** (copiar)



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE  
anacaraban@Anas-MacBook-Pro firebase-auth % npm run serve
```



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE  
DONE Compiled successfully in 5057ms  
  
App running at:  
- Local: http://localhost:8080/  
- Network: http://10.125.80.114:8080/  
  
Note that the development build is not optimized.  
To create a production build, run npm run build.
```

Caso o seu projeto não possua nenhum erro de sintaxe, na consola aparecerá o endereço onde pode testar. Basta clicar no primeiro link (Local) que abrirá uma página no browser

Ligaçāo a uma base de dados externa (API Firebase)

Avance para os passos seguintes caso tenha conseguido implementar a página de registo e de login no seu projeto.

[Não necessário para a implementação da funcionalidade de login e registo] Poderá criar a navbar no ficheiro App.vue que a aplicará em todas as restantes páginas. Para fazer a ligação entre as páginas, deverá manter a directiva <router-view/> e efetuar as ligações como na imagem abaixo

```
<router-link to="/">Home</router-link> |  
<router-link to="/login">Login</router-link> |  
<router-link to="/signup">SignUp</router-link>  
</div>  
<router-view />
```

O endereçamento a estas novas páginas deverá ser feita no ficheiro [index.js](#) dentro da pasta [router](#), replicando o que foi feito para a página Home (fazer o import, copiar o bloco de código {} e substituir pelo nome da página)

Firebase

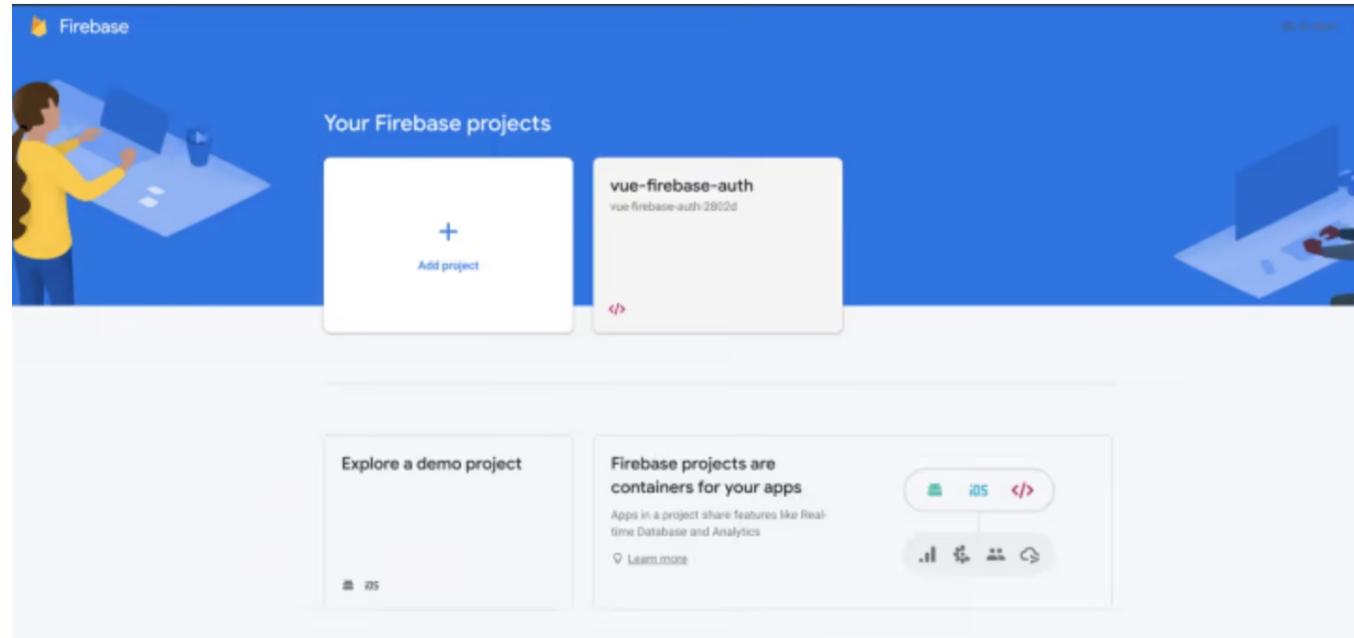
O Firebase é uma plataforma para desenvolvimento de aplicações para web e dispositivos móveis. Está disponível para diferentes plataformas (iOS, Android e web), o que agiliza o trabalho de desenvolvimento. Disponibiliza uma API que permite que possamos usar serviços de base de dados ou serviços de autenticação. Podemos também inserir dados que podem ser acedidos através de um ficheiro json

<https://console.firebaseio.google.com/u/0/>

Firebase

Criar um projeto

<https://console.firebaseio.google.com/u/0/>

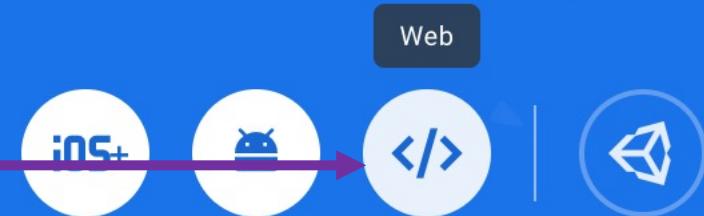


Firebase

Registrar aplicação Web

ReqDigital Spark plan

Get started by adding
Firebase to your app



Add an app to get started

Firebase

Registrar aplicação Web

Add Firebase to your web app

1 Register app

App nickname ?

My web app

! An app nickname is required.

Also set up **Firebase Hosting** for this app. [Learn more](#) ↗

Hosting can also be set up later. There is no cost to get started anytime.

Register app

2 Add Firebase SDK

Firebase

Instalar firebase no terminal
do Visual Studio

API Key
(guardar num ficheiro
de texto). Será
necessário para a
configuração

Use npm [?](#) Use a <script> tag [?](#)

If you're already using [npm](#) and a module bundler such as [webpack](#) or [Rollup](#), you can run the following command to install the latest SDK:

```
$ npm install firebase
```

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyC2Hi_vDI78e4ghEHhd6jqHfH0stEH5o8",
  authDomain: "reqdigital-55ddc.firebaseio.com",
  projectId: "reqdigital-55ddc",
  storageBucket: "reqdigital-55ddc.appspot.com",
  messagingSenderId: "627375122032",
  appId: "1:627375122032:web:d0aa68be3822b8c82ff6b5"
};

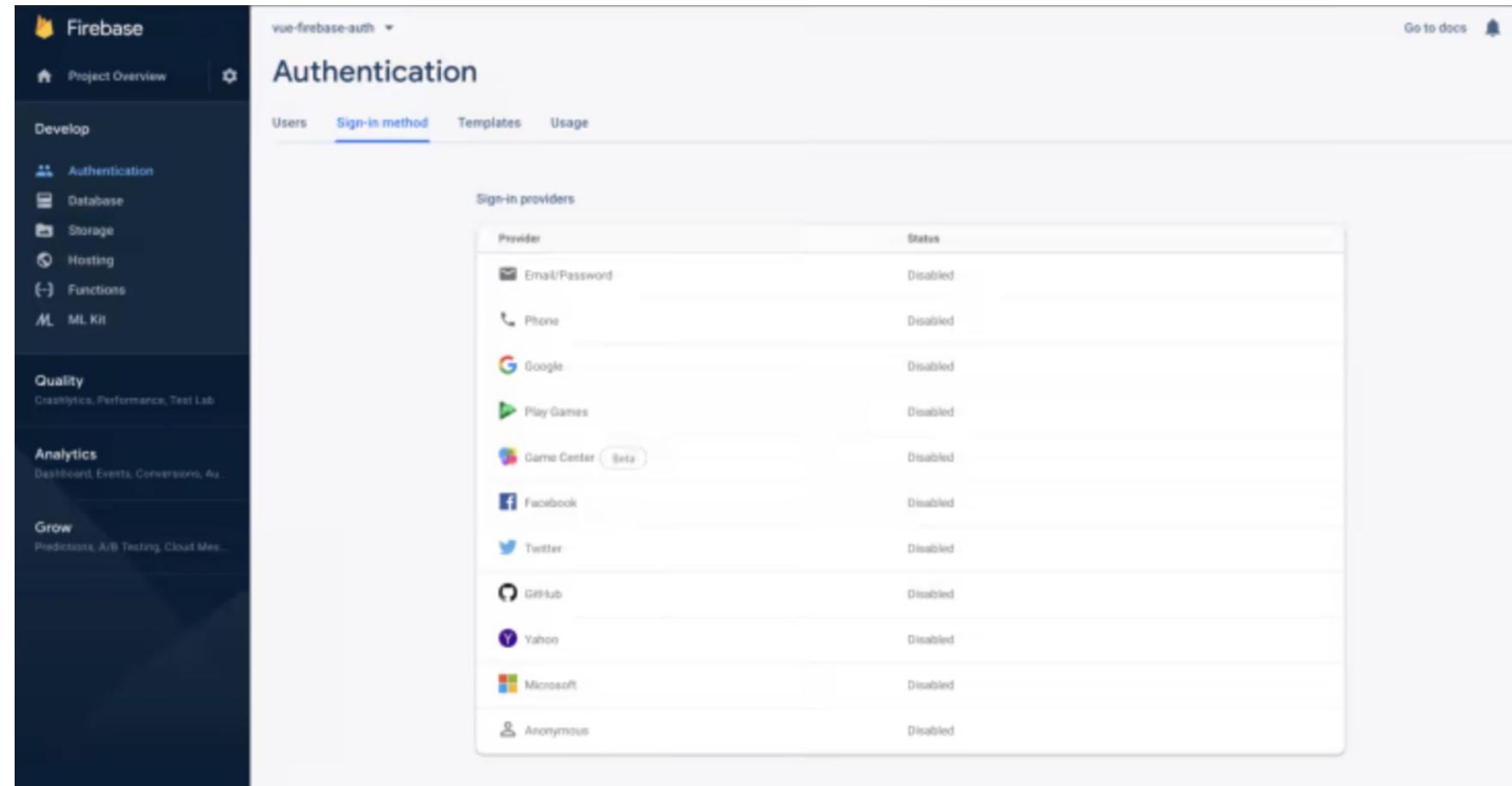
// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Note: This option uses the [modular JavaScript SDK](#), which provides reduced SDK size.

Learn more about Firebase for web: [Get Started](#), [Web SDK API Reference](#), [Samples](#)

Firebase

Auth: email e passe



The screenshot shows the Firebase console's Authentication interface. On the left, a sidebar lists 'Develop' (Authentication, Database, Storage, Hosting, Functions, ML Kit), 'Quality' (Crashlytics, Performance, Test Lab), 'Analytics' (Dashboard, Events, Conversions, Automação), and 'Grow' (Predictions, A/B Testing, Cloud Messag...). The main area is titled 'Authentication' and has tabs for 'Users', 'Sign-in method' (which is selected), 'Templates', and 'Usage'. Under 'Sign-in method', there is a table titled 'Sign-in providers' with columns 'Provider' and 'Status'. The providers listed are: Email/Password (Disabled), Phone (Disabled), Google (Disabled), Play Games (Disabled), Game Center (Beta, Disabled), Facebook (Disabled), Twitter (Disabled), GitHub (Disabled), Yahoo (Disabled), Microsoft (Disabled), and Anonymous (Disabled).

Provider	Status
Email/Password	Disabled
Phone	Disabled
Google	Disabled
Play Games	Disabled
Game Center <small>(Beta)</small>	Disabled
Facebook	Disabled
Twitter	Disabled
Github	Disabled
Yahoo	Disabled
Microsoft	Disabled
Anonymous	Disabled

Firebase

Auth: email e passe

Authentication

Sign-in method

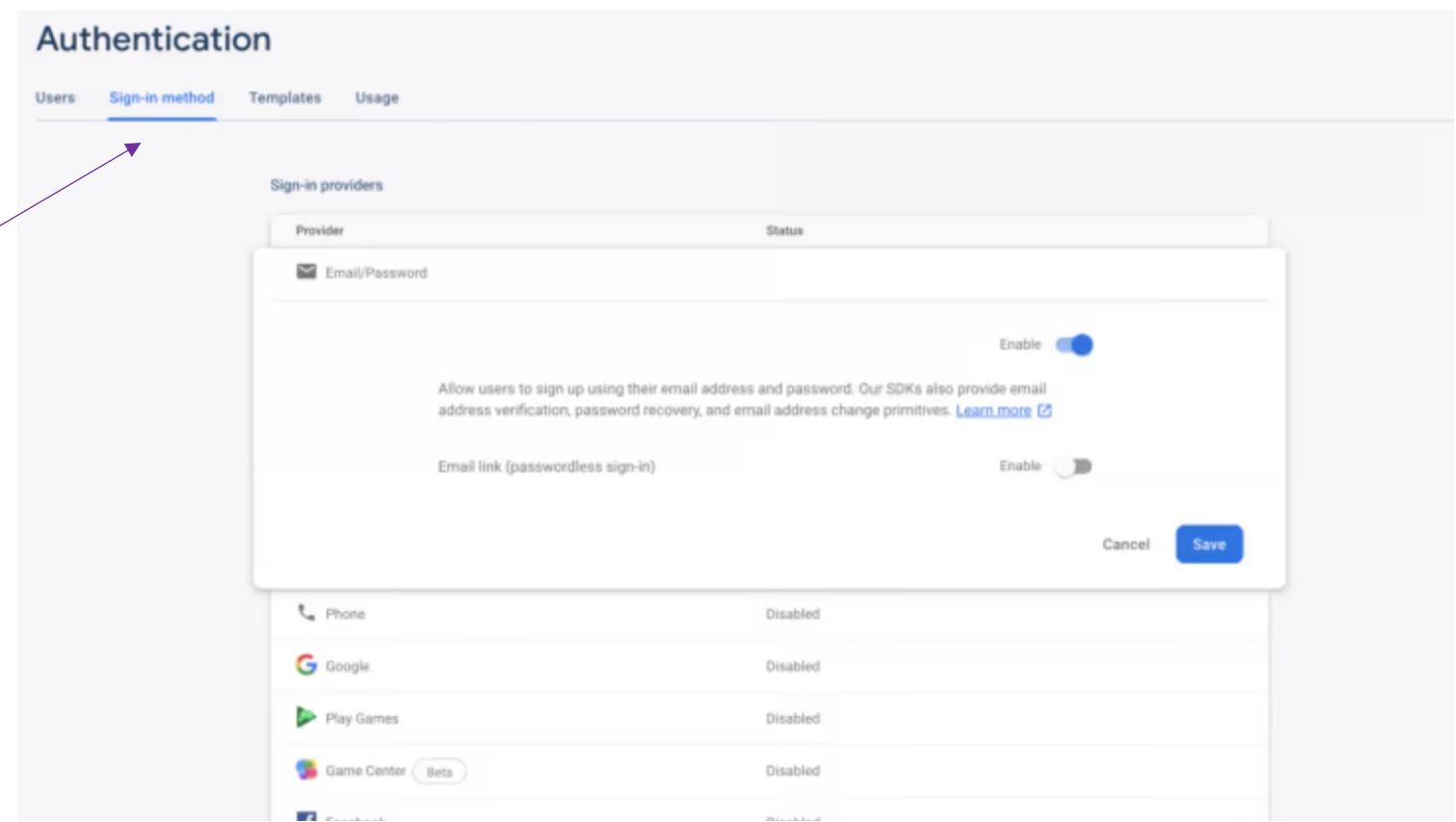
Users Sign-in method Templates Usage

Sign-in providers

Provider	Status
Email/Password	Enable <input checked="" type="checkbox"/>
Email link (passwordless sign-in)	Enable <input type="checkbox"/>
Phone	Disabled
Google	Disabled
Play Games	Disabled
Game Center <small>Beta</small>	Disabled
Facebook	Disabled

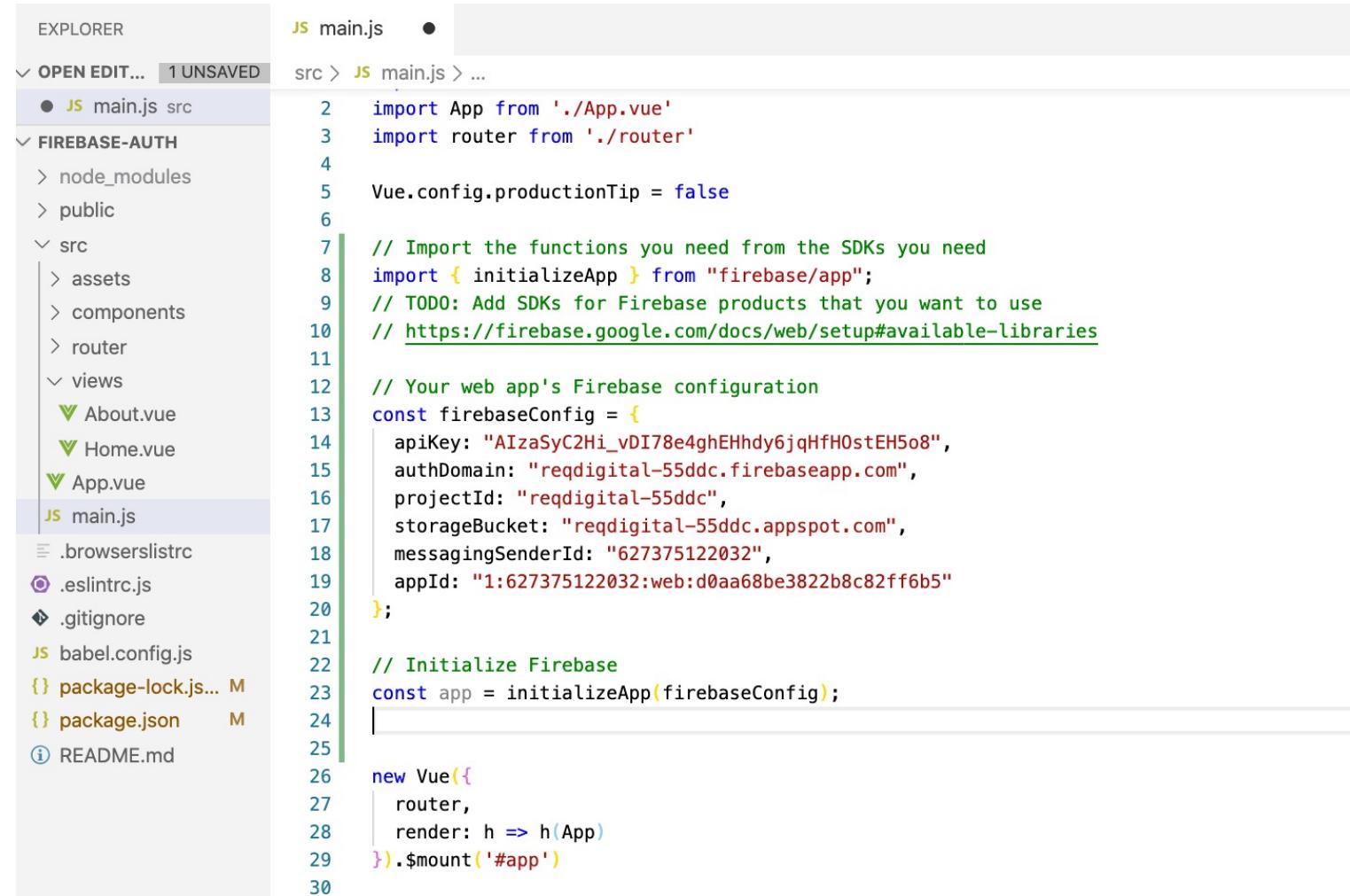
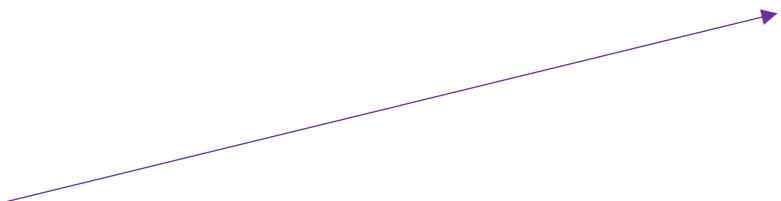
Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery, and email address change primitives. [Learn more](#)

Cancel Save



Ligaçāo Vue-Firebase

Copiar o código da API key
Colocar no ficheiro main.js



```
EXPLORER JS main.js • JS main.js > ...  
OPEN EDIT... 1 UNSAVED  
● JS main.js src  
FIREBASE-AUTH  
> node_modules  
> public  
> src  
> assets  
> components  
> router  
> views  
  ▼ About.vue  
  ▼ Home.vue  
  ▼ App.vue  
JS main.js  
.browserslistrc  
.eslintrc.js  
.gitignore  
babel.config.js  
{} package-lock.js... M  
{} package.json M  
README.md  
  
JS main.js  
src > JS main.js > ...  
2 import App from './App.vue'  
3 import router from './router'  
4  
5 Vue.config.productionTip = false  
6  
7 // Import the functions you need from the SDKs you need  
8 import { initializeApp } from "firebase/app";  
9 // TODO: Add SDKs for Firebase products that you want to use  
10 // https://firebase.google.com/docs/web/setup#available-libraries  
11  
12 // Your web app's Firebase configuration  
13 const firebaseConfig = {  
14   apiKey: "AIzaSyC2Hi_vDI78e4ghEHhd6jqHfH0stEH5o8",  
15   authDomain: "reqdigital-55ddc.firebaseio.com",  
16   projectId: "reqdigital-55ddc",  
17   storageBucket: "reqdigital-55ddc.appspot.com",  
18   messagingSenderId: "627375122032",  
19   appId: "1:627375122032:web:d0aa68be3822b8c82ff6b5"  
20 };  
21  
22 // Initialize Firebase  
23 const app = initializeApp(firebaseConfig);  
24  
25  
26 new Vue({  
27   router,  
28   render: h => h(App)  
29 }).$mount('#app')
```

Registo e Login

- Ir ao site <https://firebase.google.com/docs/auth/web/password-auth#web-version-8>

confirmar o tipo de autenticação pretendido

Copiar o código

Colocar dentro da Função correspondente
Neste caso é a do registro

```
createUserWithEmailAndPassword .
```

Versão 9 para a Web (modular) Versão 8 para a Web (namespaced)

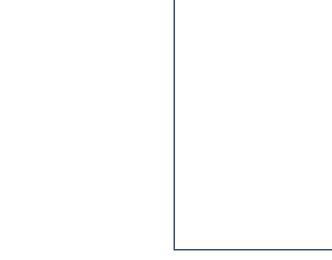
```
firebase.auth().createUserWithEmailAndPassword(email, password)
  .then((userCredential) => {
    // Signed in
    var user = userCredential.user;
    // ...
  })
  .catch((error) => {
    var errorCode = error.code;
    var errorMessage = error.message;
    // ...
});
```

email.js ↗

Se a nova conta for criada, o usuário será conectado automaticamente. Confira a seção "Próximas etapas" abaixo para obter os detalhes do usuário conectado.

Registo e Login

- Adicionar import a biblioteca do firebase



```
64 </template>
65 <script>
66 import firebase from "firebase";
67 export default [
68   name: "Signup",
69   data() {
70     return {
71       email: "",
72       password: "",
```

Registo e Login

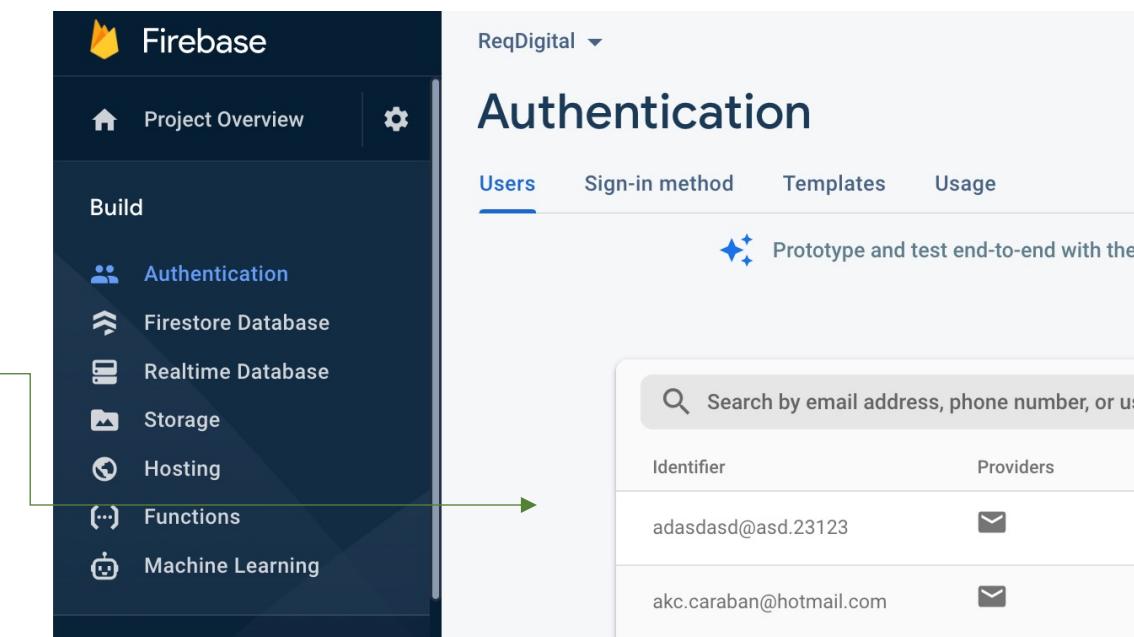
- Criar uma função login cujo evento é desencadeado quando o utilizador carrega no botão correspondente (login/registo). Registar essa função nos methods. Dentro dessa função registo, copiar o seguinte código

```
methods: {
  signupRequest() {
    firebase
      .auth()
      .createUserWithEmailAndPassword(this.email,
        this.password)
      .then(
        () => {
          this.successMessage = "Register Successfully.";
        },
        error => {
          let errorResponse = JSON.parse(error.message);
          this.errorMessage = errorResponse.error.message;
        }
      );
  },
},
```

Registo e Login

- No terminal -> npm add [firebase@^8.10.0](#)
- Guardar ficheiro
- No terminal -> npm run serve
- Testar e verificar se efectuo o registo no firebase

Se quiser enviar para outra página (em caso de sucesso na função) usar:
`this.$router.push('Home')`



The screenshot shows the Firebase console's Authentication section. On the left, a sidebar lists various services: Project Overview, Build, Authentication (which is selected and highlighted in blue), Firestore Database, Realtime Database, Storage, Hosting, Functions, and Machine Learning. A green arrow points from the 'Authentication' link in the sidebar to the corresponding tab in the main content area. The main content area is titled 'Authentication' and contains tabs for 'Users', 'Sign-in method', 'Templates', and 'Usage'. Under the 'Users' tab, there is a search bar at the top with the placeholder 'Search by email address, phone number, or user ID'. Below the search bar is a table with two rows of user data. The columns are 'Identifier' and 'Providers'. The first row shows 'adasdasd@asd.23123' and an envelope icon. The second row shows 'akc.caraban@hotmail.com' and an envelope icon.

Identifier	Providers
adasdasd@asd.23123	
akc.caraban@hotmail.com	

Registo e Login

No caso da função do login, replicar a função do registo efetuando a seguinte alteração

```
firebase  
  .auth()  
  .createUserWithEmailAndPassword(t
```

```
firebase  
  .auth()  
  .signInWithEmailAndPassword(this
```

Projeto Final do Módulo

Site em Vue.js (usando obrigatoriamente o cliente e não o cdn) com as seguintes funcionalidades:

- Registo, Login, barra de navegação e página de erro
- Página do Admin (deverá ser um formulário com inputs, o tipo e número de inputs deverá corresponder a informação que o aluno quer registar no seu site. Exemplo: site p/ supermercado: a página de admin apresenta um formulário para introduzir o nome do produto, preço, descrição e imagem) -> **a introdução dos dados na BD será feita em aula**
- Página para ler dados inseridos pelo Admin (este conteúdo pode ser inserido na página Home)
- Página para consumir dados de uma API externa -> **será feito em aula**
- Footer

Projeto Final do Módulo

Entrega obrigatória e individual para o dia:

21 de Março, até às 23h59. Via [email](#) (zip do projeto) e link para o github
(sem relatório)

Para o email: ana.caraban@staff.uma.pt

qualificar

TAL

