



Fakultet informatike u Puli

# Vue.js - Autentifikacija

U ovoj vježbi prikazivat ćemo jedan od načina na koji se može ostvariti autentifikacija u aplikaciji. Pritom ćemo koristiti Firebase bazu i JS pakete.

## Koraci

1. Kod sa prethodnih VUE-03 vježbi možemo pruezti s GitHuba. Repozitorij: <https://github.com/fipu-nastava/fipugram> (branch `step3`). Preuzimanje s Git-a, instaliranje paketa i pokretanje aplikacije pojašnjeno je u prethodnim vježbama.
2. Za korištenje Firebase-a u JavaScript/Vue aplikacije potrebno je uključiti Firebase module. Tu nam pomaže `npm` - upravitelj paketima.

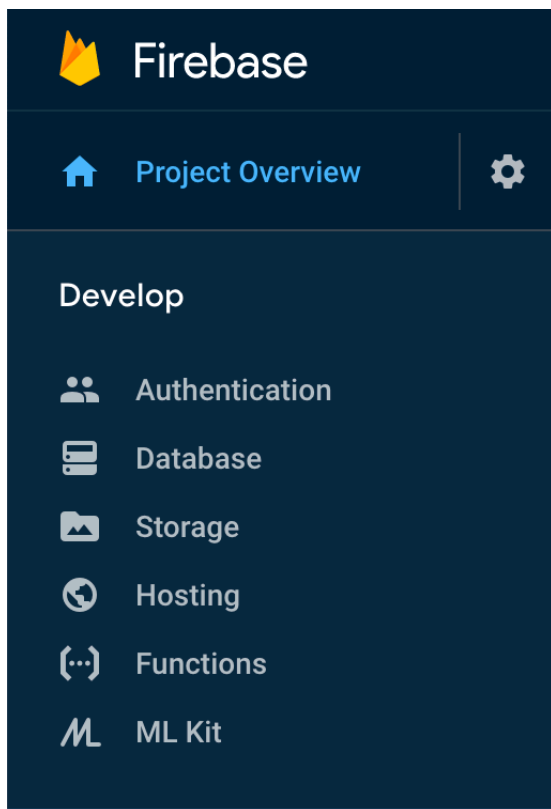
```
1 # u direktoriju projekta
2
3 npm install --save firebase
```

3. U projektu ćemo napraviti novi JS modul `firebase.js` sa sljedećim sadržajem:

```
1 import firebase from 'firebase/app';
2 import 'firebase/auth';
3 import 'firebase/firestore';
4
5 // zamijeniti s vašom konfiguracijom!
6 var firebaseConfig = {
7   apiKey: 'ZAMIJENI ME!!',
8   authDomain: 'fipugram.firebaseio.com',
9   projectId: 'fipugram',
10  storageBucket: 'fipugram.appspot.com',
11  messagingSenderId: 'ZAMIJENI ME!!!',
12  appId: 'ZAMIJENI ME!!!',
13 };
14 // Initialize Firebase
15 firebase.initializeApp(firebaseConfig);
```

Time smo uključili potrebne `firebase` JavaScripte objekte u svoju aplikaciju. U ovom koraku otvorit ćemo novi Firebase račun (ukoliko ga već nemamo). Potrebno je izvršiti registraciju na <https://firebase.google.com> i ulogirati se na administracijsku konzolu: <https://console.firebase.google.com/u/0/> i stvoriti novi Firebase projekt (nije nužno uključiti Firebase analytics).

4. Otvori svoj Firebase projekt i sa izbornika odaberi **Authentication**. Zatim odaberi **Set up sign-in method** i uključi (**Enable**) Email/Password način logina (za svoj projekt možeš podesiti i različite oblike logina poput Google ili Facebook logina).



Slika 1. Izbornik Firebase konzole.

5. Još je potrebno u Vue projektu postaviti da se spaja na naš Firebase projekt. Do sada smo samo uključili Firebase biblioteke i omogućili autentifikaciju preko emaila i šifre. Podatke za spajanje na Firebase projekt možemo naći u Firebase konzoli u postavkama projekta (klik na zupčanik na izborniku) u sekciji **Firebase SDK snippet** i podsekciji **CDN** (ili **Config**).

### Firestore SDK snippet

☐ Automatic  ☒ CDN  ☐ Config 

Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

```
<!-- The core Firebase JS SDK is always required and must be listed
<script src="https://www.gstatic.com/firebasejs/7.5.2/firebase-app.js"

<!-- TODO: Add SDKs for Firebase products that you want to use
      https://firebase.google.com/docs/web/setup#available-libraries

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIzaSyDDxH7r2SbULMH4PftZQ4EJSUq0rEh9ZYQ",
    authDomain: "instaclone-ntankovic.firebaseio.com",
    databaseURL: "https://instaclone-ntankovic.firebaseio.com",
    projectId: "instaclone-ntankovic",
    storageBucket: "instaclone-ntankovic.appspot.com",
    messagingSenderId: "98968488013",
    appId: "1:98968488013:web:d2235bb46d4c788d7a3404"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
</script>
```



**Slika 2.** Primjer Firestore konfiguracije.

Navedenu konfiguraciju kopiramo i smjestimo u naš `firebase.js` modul.

6. Time je naša aplikacija povezana sa Firestoreom i možemo napraviti mogućnost registracije pomoću komponente `Signup.vue`. Prvo ćemo definirati Vue varijable za Email i dvije lozinke u

`Signup.vue` :

```
1  <script>
2  export default {
3    data() {
4      return {
5        email: "",
6        password: "",
7        passwordRepeat: ""
8      }
9    }
10 }
11 </script>
```

Zatim ćemo povezati navedene varijable sa `v-model` na odgovarajuća `input` polja u `<template>` sekciji:

```
1   <form>
2       <div class="form-group">
3           <label for="emailField">Email address</label>
4           <input v-model="email" type="email" class="form-control"
id="emailField" aria-describedby="emailHelp" placeholder="Enter email">
5           <small id="emailHelp" class="form-text text-muted">We'll never
share ...</small>
6       </div>
7       <div class="form-group">
8           <label for="passwordField">Password</label>
9           <input v-model="password" type="password" class="form-control"
id="passwordField" placeholder="Password">
10      </div>
11      <div class="form-group">
12          <label for="confirmPasswordField">Confirm Password</label>
13          <input v-model="passwordRepeat" type="password" class="form-
control" id="confirmPasswordField" placeholder="Confirm password">
14      </div>
15          <button type="submit" class="btn btn-primary mt-5">Sign
up</button>
16      </form>
```

Potrebna nam je i nova metoda koja će učitati utipkane podatke i pozvati Firebase metodu za registraciju. Dodati ćemo je u `methods` sekciju:

```
1   export default {
2       name: "signup",
3       data() {
4           return {
5               email: "",
6               password: "",
7               passwordAgain: ""
8           }
9       },
10      methods: {
11          signup() {
12              firebase.auth().createUserWithEmailAndPassword(this.email,
this.password).catch(function(error) {
13                  console.log(error);
14              });
15          }
16      }
17  }
```

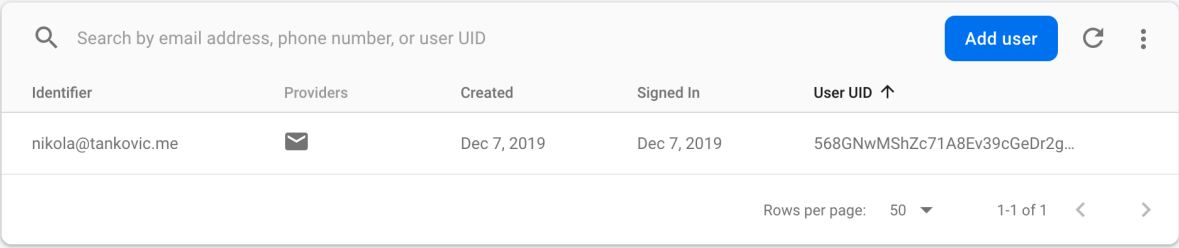
Namjestit ćemo da se navedena metoda poziva prilikom *submita forme* ( `<template>` dio u `Signup.vue` ):

```

1    ...
2    <form @submit.prevent="signup">
3    ...

```

Ukoliko je sve u redu trebali bi se moći registrirati i vidjeti novoga korisnika u Firebase konzoli (sekcija Authentication):



Identifier	Providers	Created	Signed In	User UID ↑
nikola@tankovic.me		Dec 7, 2019	Dec 7, 2019	568GNwMShZc71A8Ev39cGeDr2g...

Rows per page: 50 1-1 of 1

- Na sličan način podesiti `Login.vue` komponentu. Potrebno je umjesto Firebase metode za *signup* koristiti metodu za *sign-in*. Možeš ju pronaći u Firebase dokumentaciji :) P.S. Razmisli jesu li potrebni baš identične varijable u `data` sekciji.
- Registracija i Login sada rade. Međutim aplikacija je isto nezavisno o tome jeli trenutni korisnik ulogiran ili ne. U ovom ćemo koraku:
  - razlikovati kada je korisnik ulogiran i prema tome prikazati ili sakriti određene dijelove (npr. Login button),
  - nakon *logina* ili *logouta* preusmjeriti korisnika na `/home` ili `/login`.

Da bi pratili stanje autentifikacije korisnika moramo dodati tzv. *listener* koji reagira na promjenu stanja autentifikacije (<https://firebase.google.com/docs/reference/js/firebase.auth.Auth.html>).

U glavnu komponentu `App.vue` dodajemo sljedeći isječak:

```

1    firebase.auth().onAuthStateChanged((user) => {
2        if (user) {
3            // User is signed in.
4            console.log('*** User', user.email);
5            store.currentUser = user.email;
6        } else {
7            // User is not signed in.
8            console.log('*** No user');
9            store.currentUser = null;
10
11            if (router.name !== 'login') {
12                router.push({ name: 'login' });
13            }
14        }
15    });
16

```

**! Razmisli:** zašto smo praćenje autentifikacije dodali u `App.vue`, a ne recimo u `Login.vue` ?

U `store.js` dodajemo varijablu `currentUser` pomoću koje ćemo razlikovati postoji li trenutni korisnik. Inicijalizirat ćemo varijablu na `null` čime naznačujemo da u početku ne postoji ulogirani korisnik.

```
1 // src/store.js
2
3 export default {
4   searchTerm: '',
5   currentUser: null, // nova varijabla
6 };
```

Iskoristit ćemo `currentUser` varijablu za prikaz/sakrivanje login/logout buttona. Još je potrebno napraviti i `logout` akciju. Već imamo `logout` metodu u koju treba pozvati `Firebase` metodu za logout. Pronađi ju u dokumentaciji i zamijeni kod u postojećoj metodi. Zašto nije potrebno postaviti `store.currentUser = null` ?

```
1 <!-- App.vue -->
2
3 <ul class="navbar-nav ml-auto">
4   <li v-if="!store.currentUser" class="nav-item">
5     <router-link to="/login" class="nav-link">Login</router-link>
6   </li>
7   <li v-if="!store.currentUser" class="nav-item">
8     <router-link to="/signup" class="nav-link">Sign up</router-link>
9   </li>
10  <li v-if="store.currentUser" class="nav-item">
11    <a href="#" @click.prevent="logout()" class="nav-link">Logout</a>
12  </li>
13 </ul>
```

9. U zadnjem koraku još ćemo malo podesiti redirekciju i neovlašteni pristup nekim stranicama. Jedan od boljih načina na koji to možemo riješiti jest da uz svaku rutu u ruteru posebno naznačimo jeli potreban login za tu rutu. U tu svrhu ćemo koristiti `meta` podatke svake rute:

```
1
2 const routes = [
3   {
4     path: '/',
5     name: 'home',
6     component: Home,
7     meta: { // dodajemo meta-podatke na home rutu jer je potreban
      login za nju
8     needsAuth: true, // možemo nazvati ovaj atribut kako želimo
9   },
10 },
11 // ...
12 ];
```

Zatim radimo prilagodbu u `App.vue` :

```

1  firebase.auth().onAuthStateChanged((user) => {
2      const currentRoute = router.currentRoute;
3
4      if (user) {
5          // User is signed in.
6          console.log('*** User', user.email);
7          store.currentUser = user.email;
8
9          // take me home
10         if (!currentRoute.meta.needsAuth) {
11             router.push({ name: 'home' });
12         }
13     } else {
14         // User is not signed in.
15         console.log('*** No user');
16         store.currentUser = null;
17
18         // kick me out
19         if (currentRoute.meta.needsAuth) {
20             router.push({ name: 'login' });
21         }
22     }
23 });

```

I kao dodatni vid sigurnosti dodat ćemo provjeru u trenutku kada Vue mijenja rute, to možemo učiniti sa `beforeEach` metodom u ruteru:

```

1  import store from '@/store';
2
3  // ...
4
5  // dodajemo na dnu
6  router.beforeEach((to, from, next) => {
7      console.log('Bio sam na', from.name, 'idem na', to.name, 'a korisnik
je', store.currentUser);
8
9      const authenticated = store.currentUser !== null;
10
11     if (!authenticated && to.meta.needsAuth) {
12         next('login');
13     } else {
14         next();
15     }
16 });
17
18 export default router;

```

10. **Samostalni zadatak.** Provjeriti jesu li dvije lozinke iste pri registraciji. Napraviti novu stranicu postavki korisnika. Na toj stranici omogućiti korisniku promjenu lozinke. Zadatak predati na predviđeno mjesto pod šifrom VUE-04.

