

Techniki Projektowania Frontendowego	LAB 04
Autentykacja z zewnętrznym providerem	

Dzisiaj skupimy się na autentykacji przez zewnętrznego providera, takiego jak np. Google. Ta metoda umożliwia użytkownikom logowanie się do aplikacji za pomocą swoich istniejących kont w serwisach takich jak Google, Facebook czy GitHub, zamiast tworzyć nowe konta. To zwiększa wygodę dla użytkowników i eliminuje potrzebę zapamiętywania kolejnych haseł. Dodatkowo, wykorzystując autentykację przez zewnętrznego providera, możemy skorzystać z silnych mechanizmów zabezpieczeń oferowanych przez te platformy, co przyczynia się do zwiększenia bezpieczeństwa aplikacji. Ta metoda także ułatwia zarządzanie tożsamościami użytkowników, a dzięki temu, że dane dotyczące logowania i autoryzacji są obsługiwane przez dostawcę, możemy skoncentrować się na budowaniu głównej funkcjonalności aplikacji.

Użytkownik przede wszystkim zostaje zwolniony z:

1. długiego procesu rejestracji
2. wymyślania nowego, skomplikowanego hasła
3. nie musi pamiętać o odświeżaniu/ zmiany hasła co jakiś czas na dedykowanej platformie
4. logowanie zajmuje mniej czasu - to tylko kilka kliknięć bez wpisywania maila, haseł itd

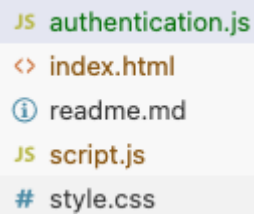
Twórz formularze tak, by były prostsze! Logowanie przez providera pozwoli nam uprościć cały proces autentykacji użytkownika do systemu.

Let's start! 🛠️

1. Dodaj przyciski do logowania i wylogowania

```
<div class="container mt-4 mb-4">
  <button id="signInButton" class="btn btn-primary">
    Create Account / Sign In
  </button>
  <button id="signOutButton" class="btn btn-secondary">
    Sign Out
  </button>
</div>
```

2. Utwórz plik z skryptem i podłącz go do HTML'a



Utwórz nowy plik javascript np. authentication.js

i podłącz go w sekcji <head> pliku index.html:

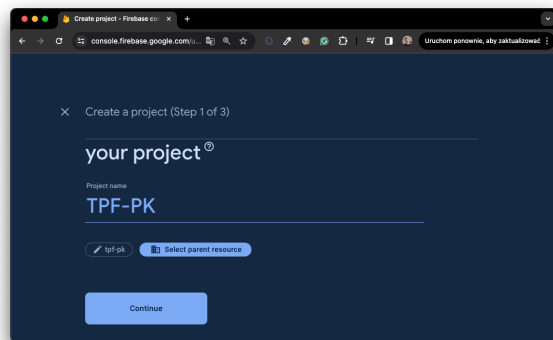
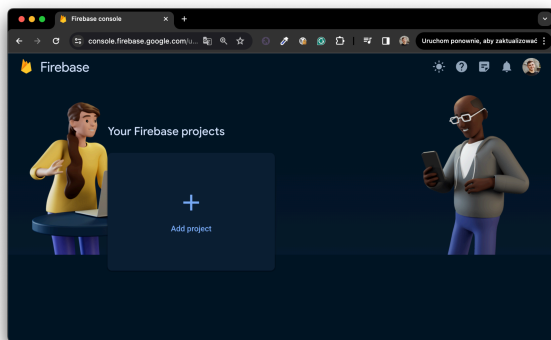
```
<script type="module" src="authentication.js" defer></script>
```

3. Utwórz konto firebase

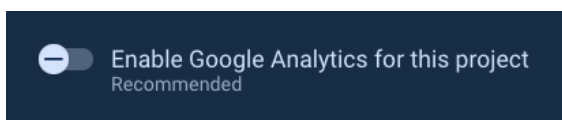
Stwórz konto na bezpłatnej platformie firebase - jest to platforma backend as a service, która umożliwia nam podpięcie pewnych usług bez konieczności tworzenia własnego servera. W tym celu wejdź na:

<https://console.firebase.google.com/>

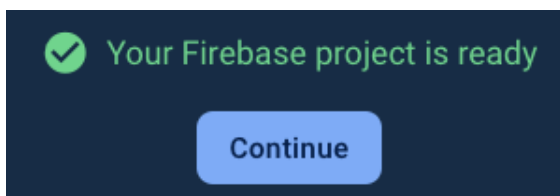
Zaloguj się z konta Google i utwórz nowy projekt.



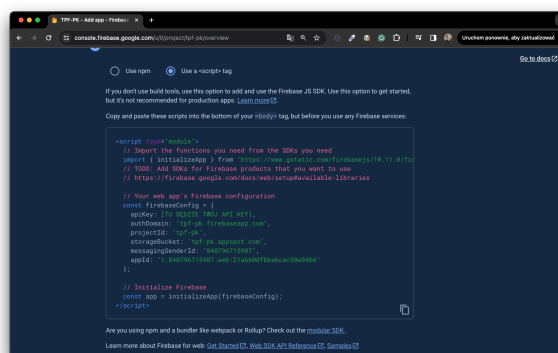
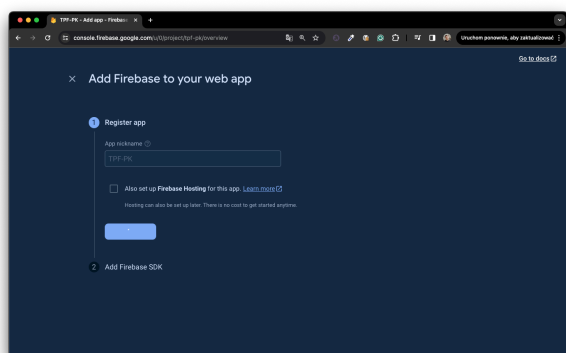
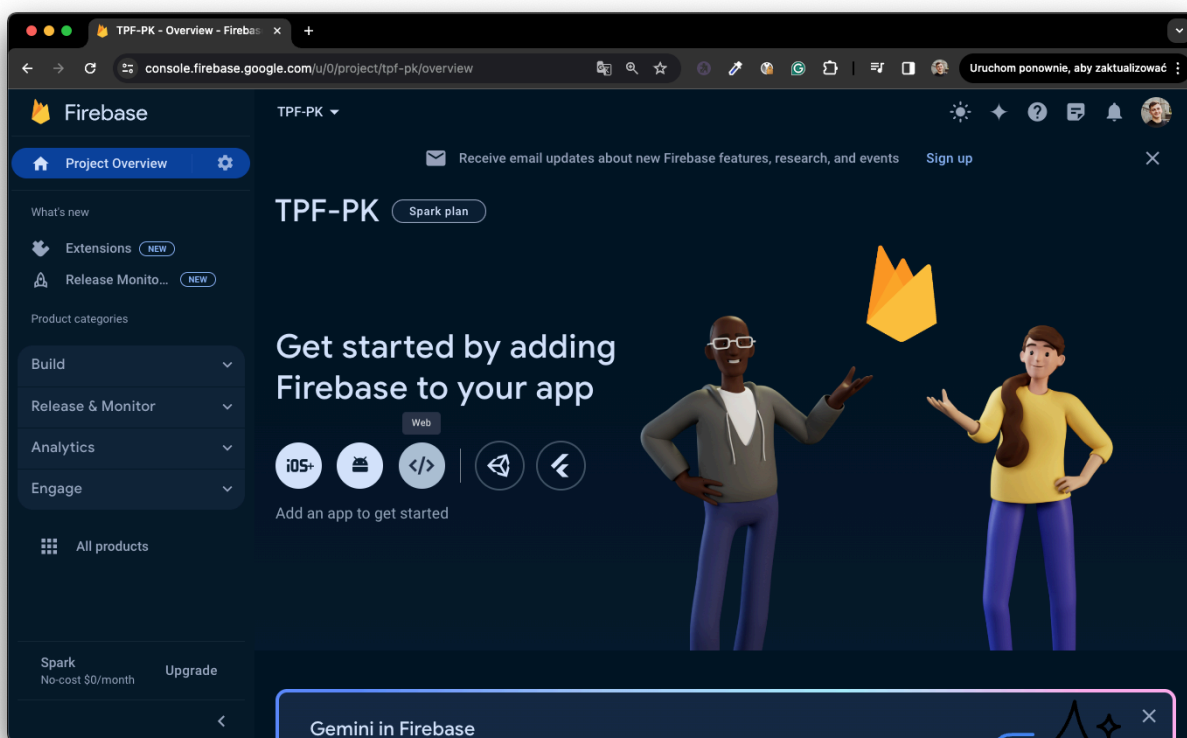
Jeśli chcesz podpiąć usługę śledzenia odwiedzin i wejść na Twoją aplikację możesz podpiąć dodatkowo usługę Google Analytics - ale w naszym przypadku nie jest to konieczne.



Zaczekaj, aż Twój projekt zostanie stworzony.



Z poziomu dashboardu wybierz ikonkę podpięcia Firebase pod aplikację internetową .



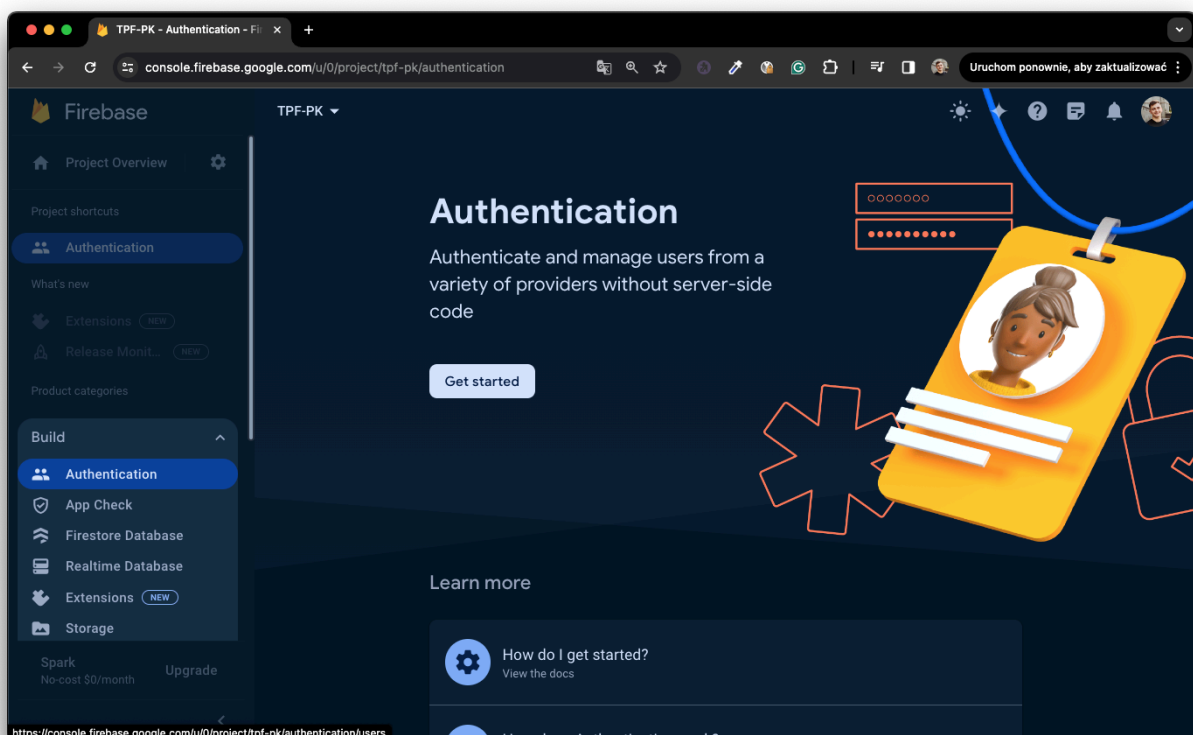
Wklej podaną zawartość tagu script do projektu z laboratorium dotyczącego formularzy. Dodaj ją do pliku **authentication.js**. Dzięki temu będziemy mogli połączyć się z usługą Firebase.

```

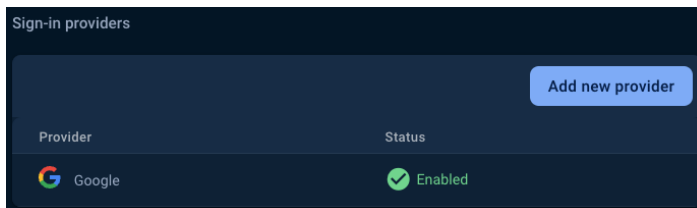
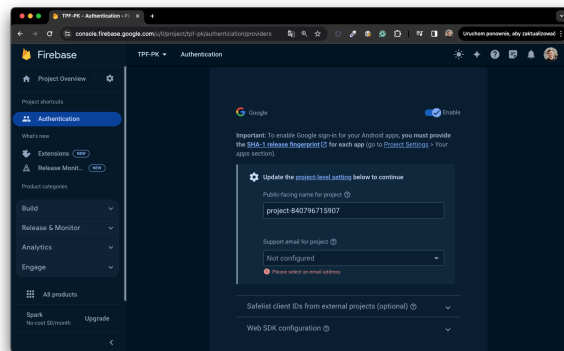
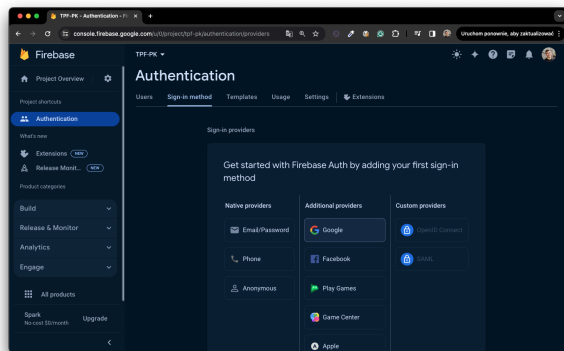
JS authentication.js U X
JS authentication.js > ...
1 // Import the functions you need from the SDKs y
2 import { initializeApp } from "https://www.gstat
3 // TODO: Add SDKs for Firebase products that you
4 // https://firebase.google.com/docs/web/setup#av
5
6 // Your web app's Firebase configuration
7 const firebaseConfig = {
8   apiKey: "[TWOJ API KEY]",
9   authDomain: "tpf-pk.firebaseio.com",
10  projectId: "tpf-pk",
11  storageBucket: "tpf-pk.appspot.com",
12  messagingSenderId: "840796715907",
13  appId: "1:840796715907:web:21a6b0dfbba6cac58
14 };
15
16 // Initialize Firebase
17 const app = initializeApp(firebaseConfig);
18

```

Następnie w **Firebase** przełącz się na zakładkę **Build** → **Authentication** i włącz Autentykację przyciskiem “**Get started**”.



Wybierz Google jako providera, przez którego będziemy się logować i podepnij swój email Google (będzie to adres e-mail prezentowany użytkownikom podczas uwierzytelniania w Google. Można go zmienić na adres e-mail zalogowanego użytkownika lub adres e-mail grupy Google zarządzanej przez użytkownika).



Wracamy do naszego pliku `authentication.js`. Po linii:

```
import { initializeApp } from
"https://www.gstatic.com/firebasejs/10.11.0/firebase-app.js";
```

Dodaj import biblioteki, która posłuży nam do autentykacji.

```
import { getAuth, GoogleAuthProvider, signInWithPopup, signOut, onAuthStateChanged } from
"https://www.gstatic.com/firebasejs/10.11.0/firebase-auth.js";
```

getAuth - funkcja ta zwraca instancję obiektu autentykacji Firebase, która jest używana do autoryzacji użytkowników.

GoogleAuthProvider - jest to klasa dostarczająca obiekt dostawcy uwierzytelniania dla uwierzytelniania za pomocą konta Google. Możesz go użyć, aby skonfigurować uwierzytelnianie za pomocą konta Google w swojej aplikacji Firebase.

signInWithPopup(provider) - ta funkcja umożliwia logowanie użytkownika za pomocą okna modalnego (popup). Po wywołaniu tej funkcji użytkownik może wybrać konto Google do zalogowania się. provider jest obiektem dostawcy, którym można dostarczyć GoogleAuthProvider.

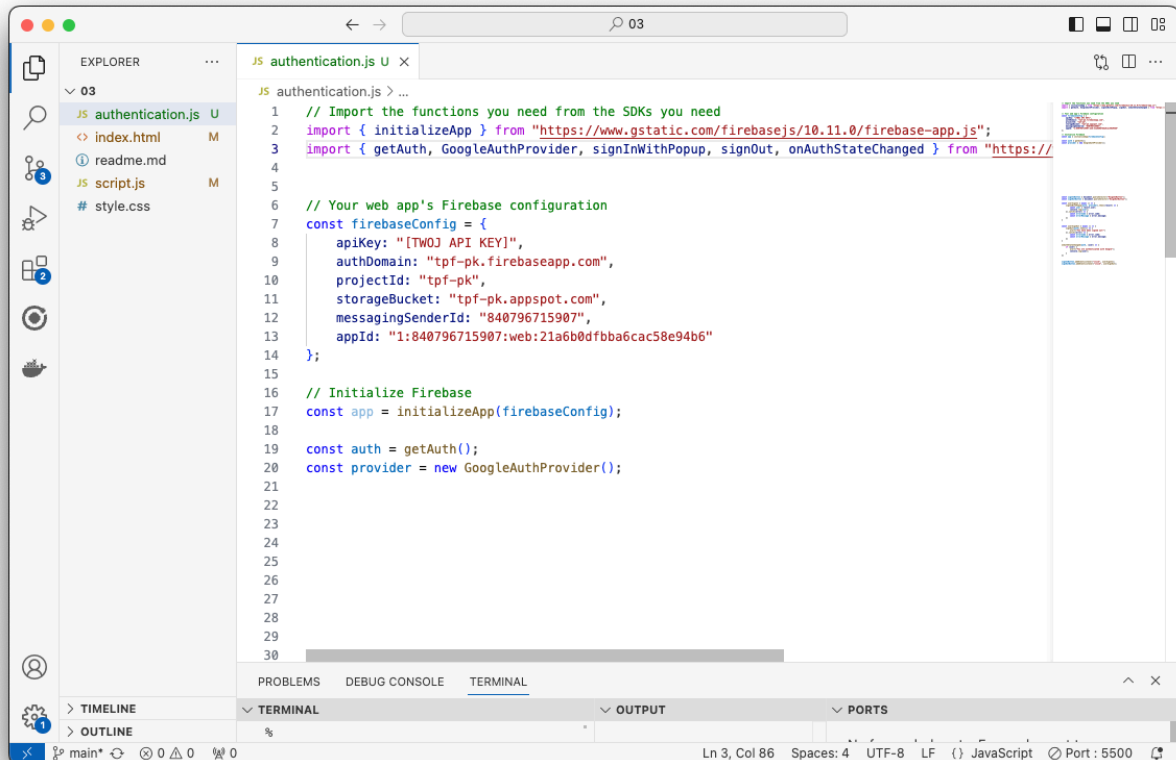
signOut() - funkcja ta służy do wylogowywania użytkownika z aplikacji.

onAuthStateChanged(nextOrObserver, error, completed) - jest to funkcja, która rejestruje obserwatora, który jest wywoływany przy zmianie stanu autentykacji użytkownika. nextOrObserver jest funkcją lub obserwatorem, który jest wywoływany, gdy stan autentykacji się zmienia. error jest funkcją, która jest wywoływana w przypadku błędu. completed jest funkcją, która jest wywoływana, gdy rejestracja obserwatora jest zakończona.

Na koniec wklejonego wcześniej skryptu dopisz inicjalizację **getAuth()** oraz **GoogleAuthProvider()**.

```
const auth = getAuth();  
const provider = new GoogleAuthProvider();
```

Całość powinna wyglądać jak poniżej:



Dodaj w **authentication.js** pobranie przycisku do logowania i innych:

```
const signInButton = document.querySelector("#signInButton");  
const signOutButton = document.querySelector("#signOutButton");
```

Następnie utwórz kilka funkcji:

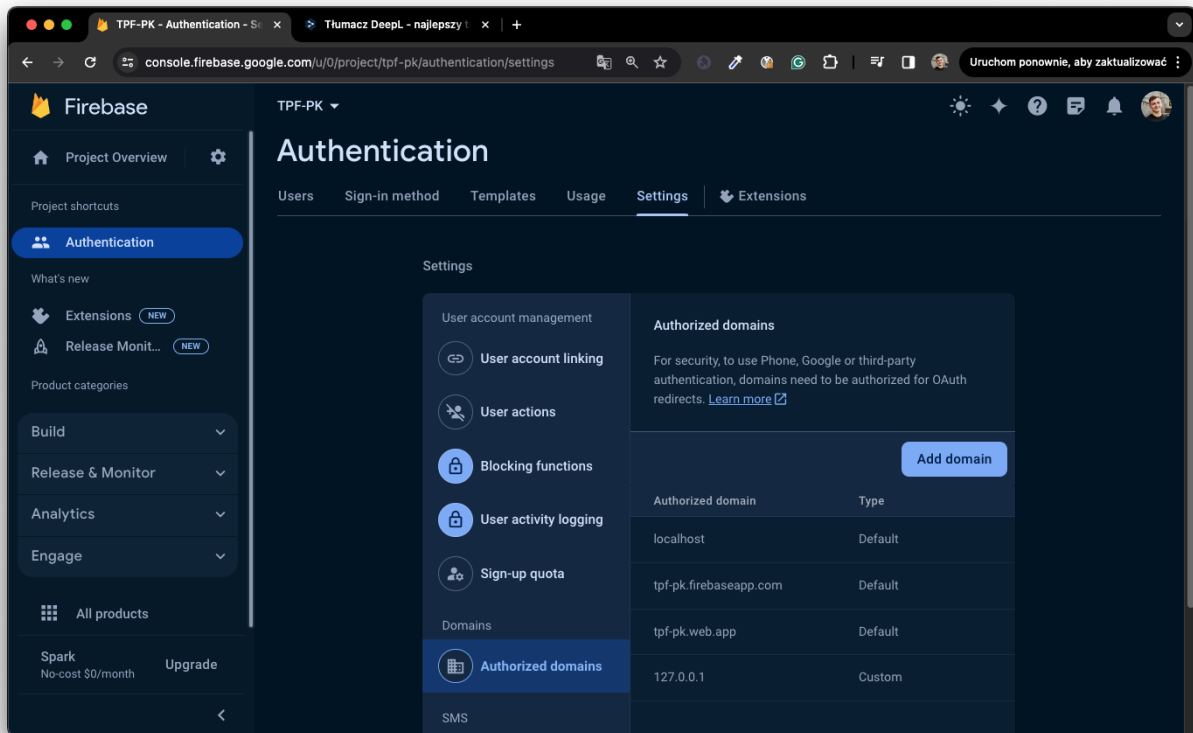
<pre>const userSignIn = async () => { signInWithPopup(auth, provider).then((result) => { const user = result.user; console.log(user); }).catch((error) => { const errorCode = error.code; const errorMessage = error.message;</pre>	Funkcja, która zostanie wywołana po kliknięciu przycisku logowania.
--	---

<pre> }) } </pre>	
<pre> const userSignOut = async () => { signOut(auth).then(() => { alert("You have been signed out!") }).catch((error) => { const errorCode = error.code; const errorMessage = error.message; }) } </pre>	<p>Funkcja, która zostanie wywołana po kliknięciu przycisku wylogowania.</p>
<pre> onAuthStateChanged(auth, (user) => { if (user) { alert("You are authenticated with Google"); console.log(user); } }) </pre>	<p>Funkcja wyświetli alert, po poprawnym zalogowaniu użytkownika.</p>
<pre> signInButton.addEventListener("click", userSignIn); signOutButton.addEventListener("click", userSignOut); </pre>	<p>Podpięcie funkcji po zdarzenie kliknięcia na przyciski.</p>

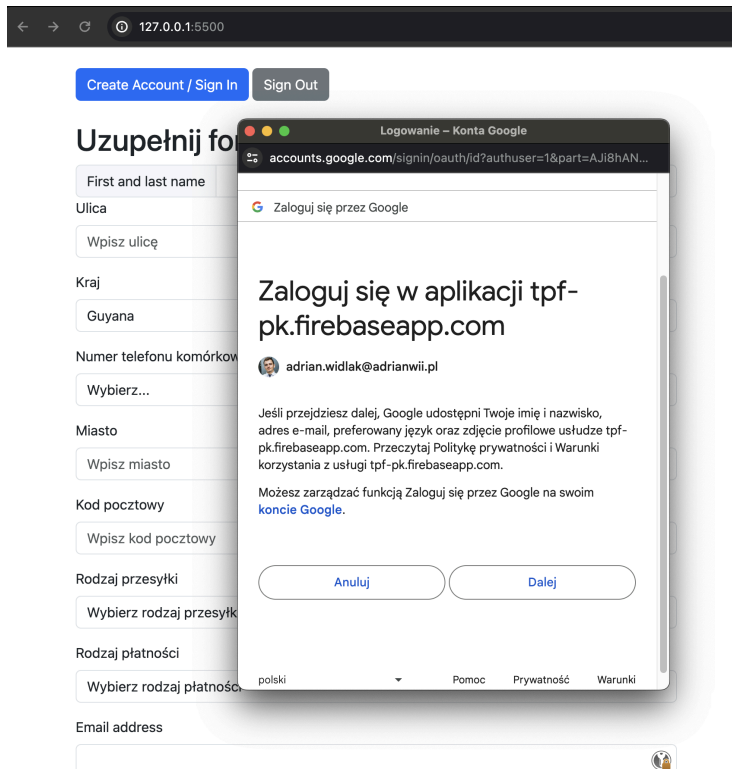
Aby uruchomić aplikację potrzebujemy servera. Jeżeli korzystasz z Visual Studio Code możesz zainstalować wtyczkę LiveServer. Wówczas w prawym dolnym rogu pojawi się narzędzie



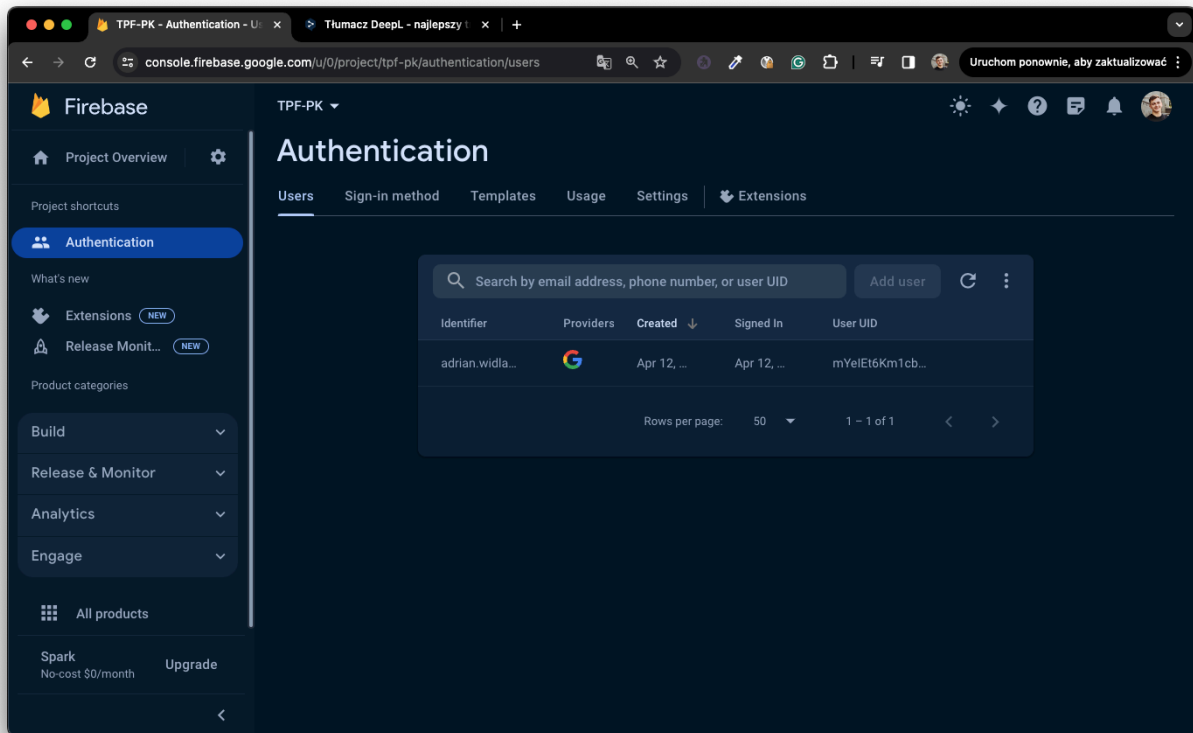
Zanim przetestujesz logowanie, dodaj adres lokalny 127.0.0.1 do zautoryzowanych domen w Firebase:



Następnie spróbuj zalogować się za pomocą zewnętrznego providera i zobacz wynik w konsoli:



Zajrzyj do konta Firebase, zakładki Authentication → Users i zobacz, czy pojawiłaś/eś się na liście zalogowanych użytkowników:



ZADANIE:

1. Na podstawie uzyskanego z Google obiektu użytkownika wstrzyknij imię, nazwisko i email w pola formularza z poprzedniego ćwiczenia:

```

UserImpl {providerId: 'firebase', proactiveRefresh: Proac
reloadListener: null, uid: 'mYeIEt6Km1cb1pgdLGZwQImH4fH3'
accessToken: "eyJhbGciOiJIUzU1NiIsImtpZCI6ImYyOThjZDA3N
auth: AuthImpl {app: FirebaseAppImpl, heartbeatService
displayName: "Adrian Widlak"
email: "adrian.widlak@adrianwii.pl"
emailVerified: true
isAnonymous: false
metadata: UserMetadata {createdAt: '1712947913202', las
phoneNumber: null
photoURL: "https://lh3.googleusercontent.com/a/ACg8ocJ3
proactiveRefresh: ProactiveRefresh {user: UserImpl, isR
providerData: [{...}]
providerId: "firebase"
reloadListener: null
reloadUserInfo: {localId: 'mYeIEt6Km1cb1pgdLGZwQImH4fH3
stsTokenManager: StsTokenManager {refreshToken: 'AMf-vB
tenantId: null
uid: "mYeIEt6Km1cb1pgdLGZwQImH4fH3"
refreshToken: (...)}
[[Prototype]]: Object

```

← → ↻
127.0.0.1:5500

Create Account / Sign In
Sign Out

Uzupełnij formularz

First and last name
Adrian
Widlak

Email address
adrian.widlak@adrianwii.pl

Ulica
Wpisz ulicę

Jeżeli u kogoś nie przechwytuje się nazwisko to można przechwycić tylko imię i mail.