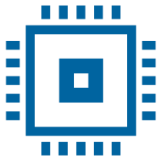
INTEGRAREA BAZELOR DE DATE CLOUD ÎN APLICAȚII WEB

Badea Cosmin - Sebastian

Cadru didactic: Șef. Lucr. Univ. Dr. Ing. Corneliu Zaharia



Universitatea Transilvania din Brașov

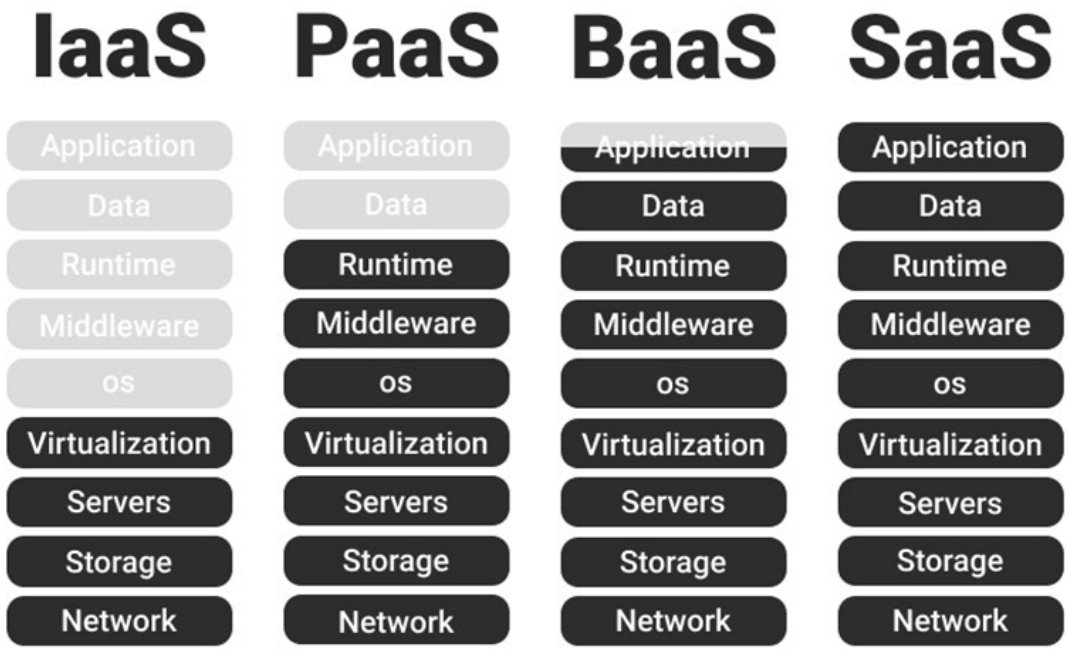
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

1. INTRODUCERE ÎN CLOUD SERVICES

Firebase

Google Cloud

- Când discutăm despre servicii Cloud, trebuie să menționăm că fiecare are un model arhitectural în spate, pe care, în funcție de cerințele aplicației, utilizatorii îl folosesc. Aceste modele diferă prin nivelul de responsabilitate pe care îl preia furnizorul de servicii și cel rămas utilizatorului:



- Infrastracture as a Service (IaaS):
  - Utilizatorul controlează infrastructura virtualizată (ex.: servere, stocare, rețea)
  - Exemplu: AWS EC2, Google Compute Engine
- Platform as a Service (PaaS):
  - Oferă o platformă pentru dezvoltarea aplicațiilor, fără grija infrastructurii
  - Exemplu: Google App Engine, Heroku, AWS Lambda
- **Backend as a Service (BaaS):**
  - **Simplifică dezvoltarea aplicațiilor oferind funcționalități preconfigurate (ex.: baze de date, autentificare)**
  - **Exemplu: Firebase**
- Software as a Service (SaaS):
  - Aplicații complete livrate utilizatorului prin intermediul internetului.
  - Exemplu: Gmail, Google Docs

- Această prezentare se axează pe **Firebase**, un serviciu oferit de **Google Cloud Platform**, care funcționează pe modelul **Backend-as-a-Service**. Firebase le oferă dezvoltatorilor acces rapid la o serie de funcționalități preconfigurate, printre care:
  - Baze de date
  - Autentificare
  - Analize
  - Etc.
- Aceste funcționalități simplifică dezvoltarea aplicațiilor web și mobile, permițând utilizatorilor să se concentreze pe experiența finală a aplicației.
- Motivele pentru care integrăm Firebase cu aplicațiile noastre React sunt:
  - Integrare ușoară
  - Sincronizare în timp real
  - Scalabilitate

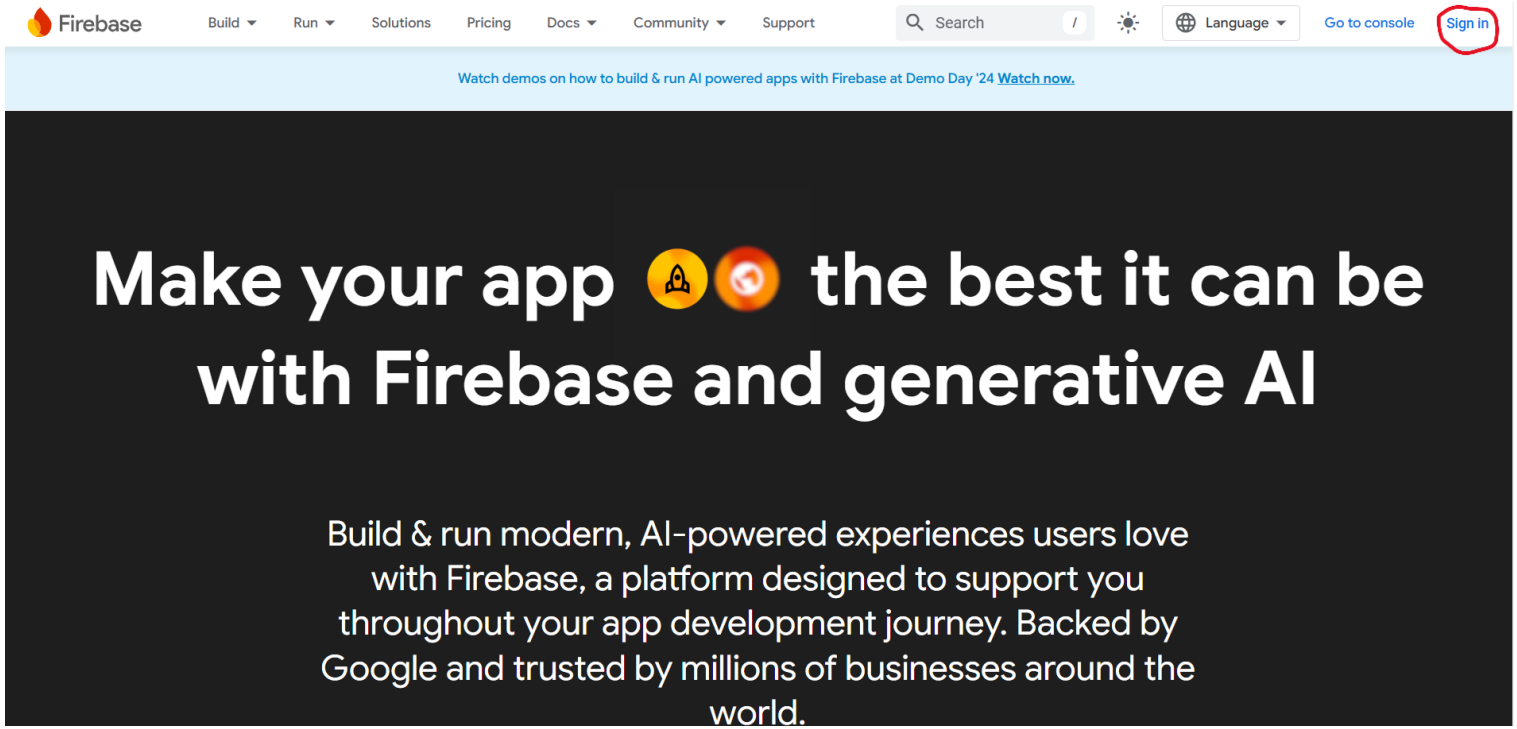
2. CONFIGURAREA FIREBASE

Firebase

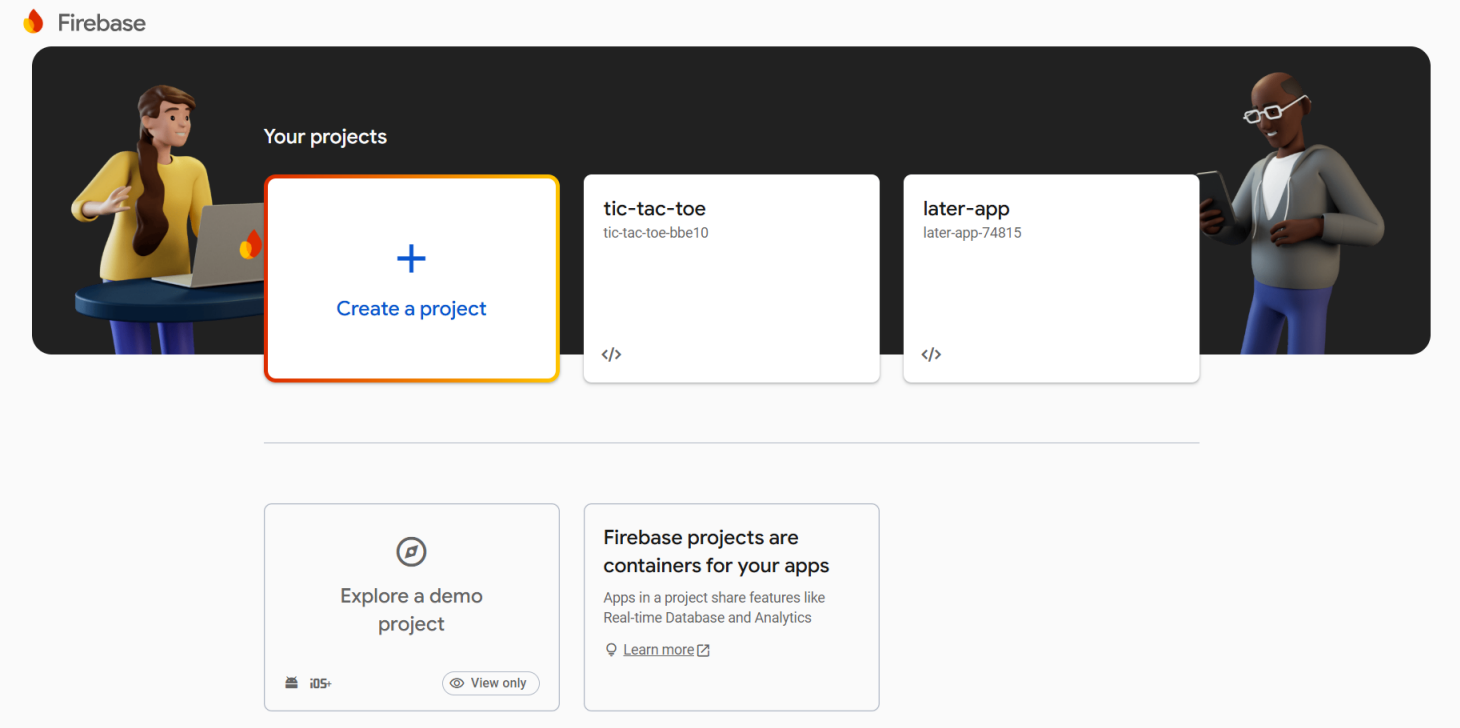
Google Cloud

- Pentru a crea o aplicație în care să utilizăm Firebase va trebui să trecem prin doi pași, și anume:
  - Să creăm baza de date Firebase (😅)
  - Să creăm aplicația web (😅)

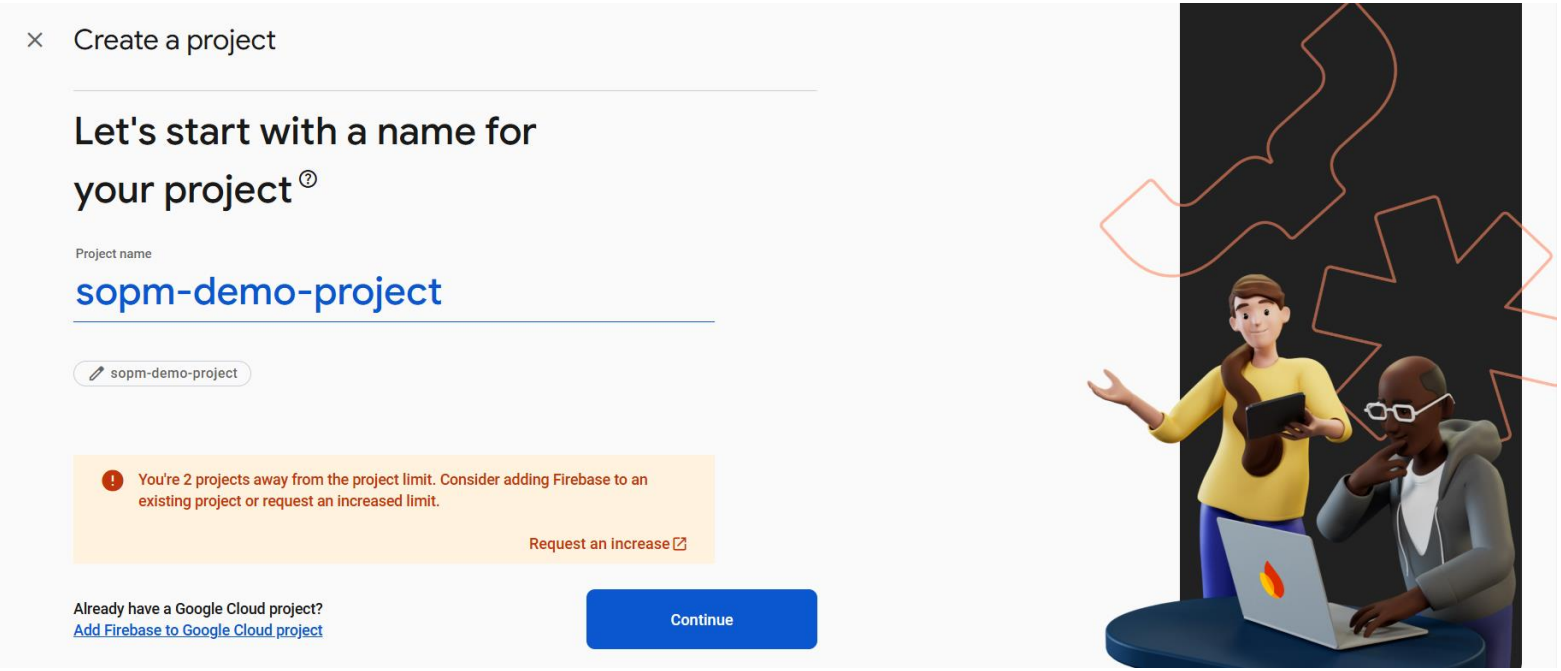
- **Să creăm baza de date**
  - Se accesează link-ul: firebase.google.com



- Apăsăm pe **Sign in** (la fel ca în poza inserată), ne conectăm cu contul de Google, iar ulterior vom fi redirecționați la pagina de la care am plecat spre conectare. Iar acolo, in loc de Sign in, ne va apărea poza de profil pe care o avem pe Google, semn că am reușit să ne conectăm. Pasul următor este să apăsăm pe **Go to console** (lângă poză)

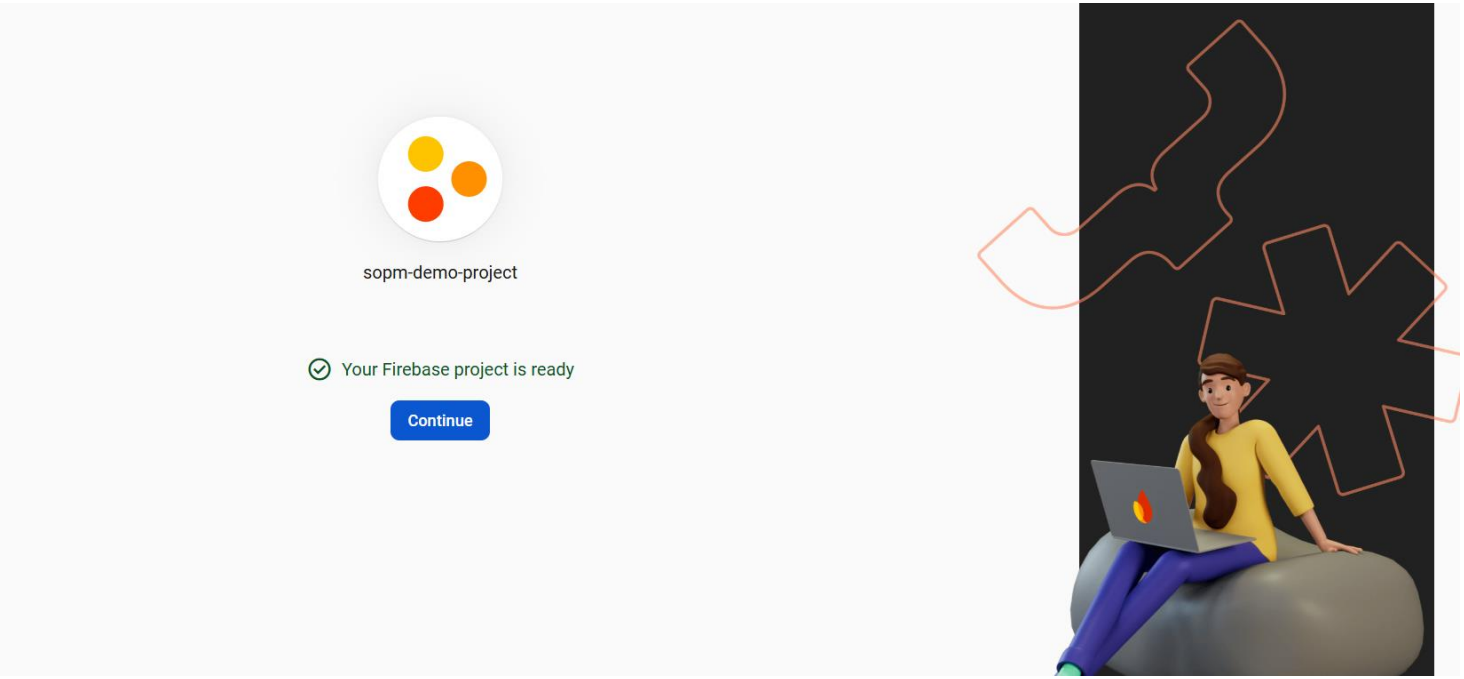


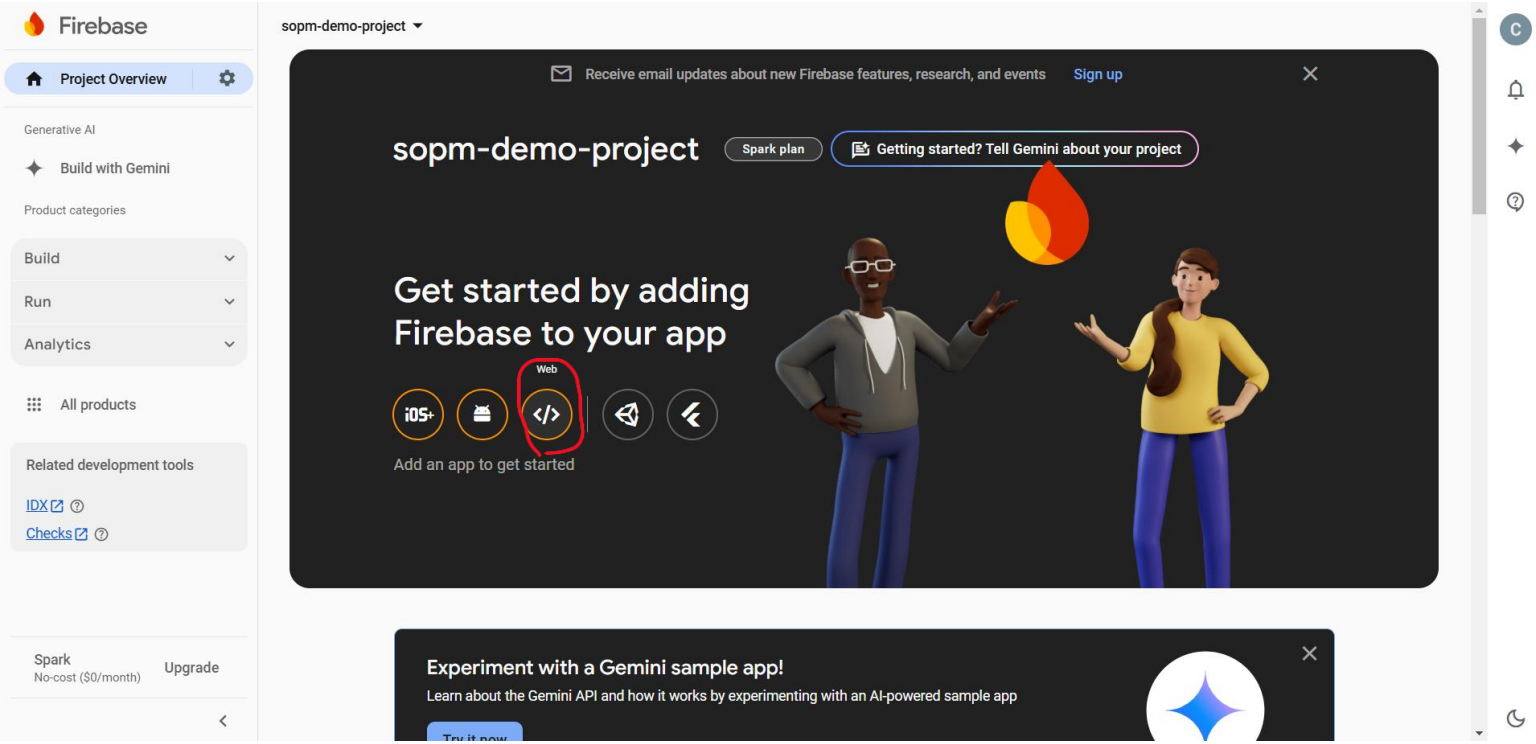
- Ne este deschisă consola (trebuie să vă arate interfața prezentată în poza anterioară, doar că fără niciun proiect lângă opțiunea de **Create a project**, dacă nu ați creat niciun proiect în trecut).



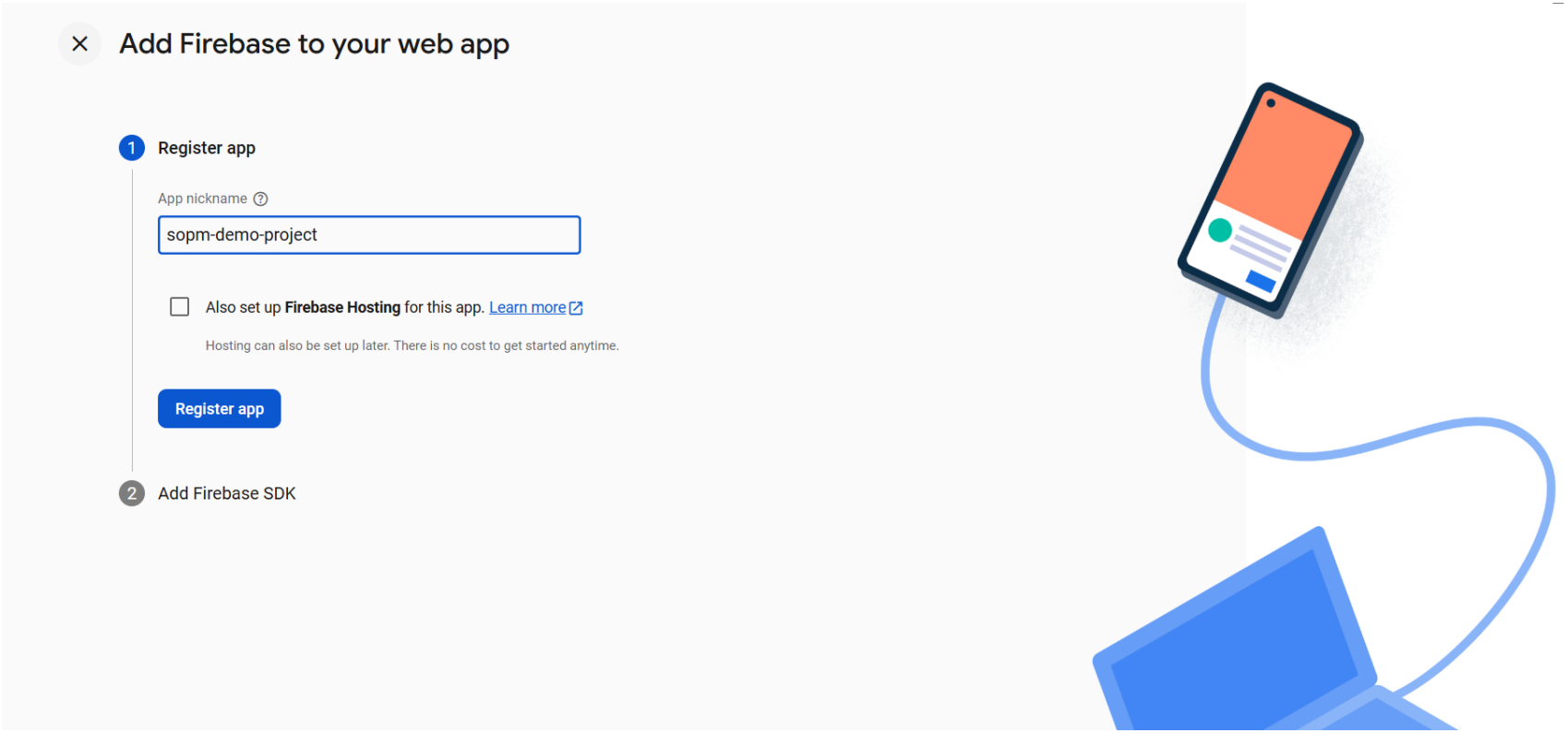
- Apăsăm pe **Create a project** și îi dăm un nume proiectului, nu contează ce nume, nu trebuie să fie unic. Apăsăm **Continue** după ce am ales numele.

- Imediat am dat Continue, vom fi redirecționați către o pagină care ne întreabă dacă dorim să activăm și Google Analytics pentru proiectul nostru. Pentru această demonstrație le vom dezactivate pentru a evita să configurăm și Google Analytics. Dam **Continue** după ce le-am dezactivat, iar ulterior ni se va crea proiectul. Când este gata, suntem anunțați și putem da **Continue**, ca în poza de mai jos.





- După ce am dat Continue, am fost redirecționați pe această pagină. Practic, de aici ne vom lua datele (**API + credențiale)** necesare pentru sincronizarea Firebase cu aplicația noastră web.
- Selectăm că dorim să adăugăm Firebase la o aplicație Web (se apasă pe butonul încercuit cu roșu in poza de sus).



- După ce am selectat opțiunea **Web**, ni se va cere un nume pentru aplicația noastră. Putem să îl dăm pe același pe care l-am dat proiectului, nu este foarte relevant (pentru testing phase-ul unui proiect).
- Debifăm Firebase Hosting, deoarece o putem face mai târziu dacă este necesar, momentan nu ne ajută cu nimic. După ce am făcut asta, apăsăm **Register App**.

- Înainte să dăm **Continue to console**, va trebui să preluăm datele de configurare pe care Firebase ni le pune la dispoziție (snippet-ul de cod oferit în acest pas, ce poate fi regăsit și în poza de mai jos.
- **IMPORTANT!** Păstrați-vă API-urile private, nu le partajați.

2 Add Firebase SDK

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 (Learn more):

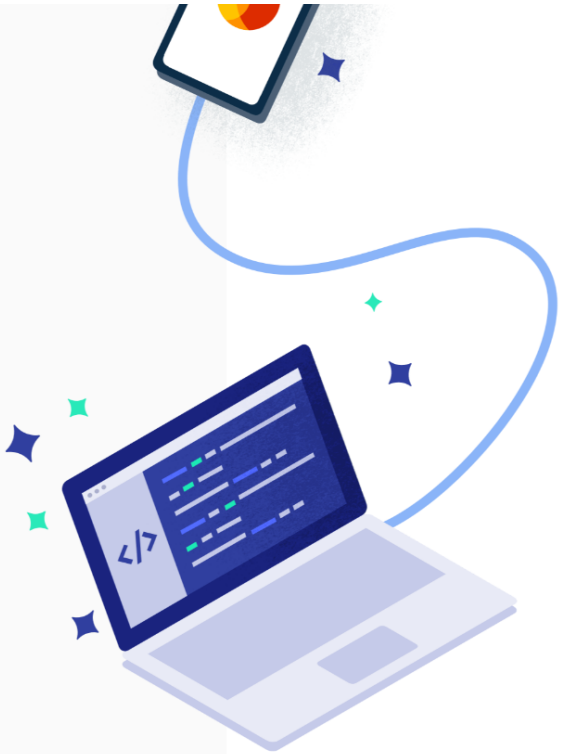
```
$ npm install firebase
```

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

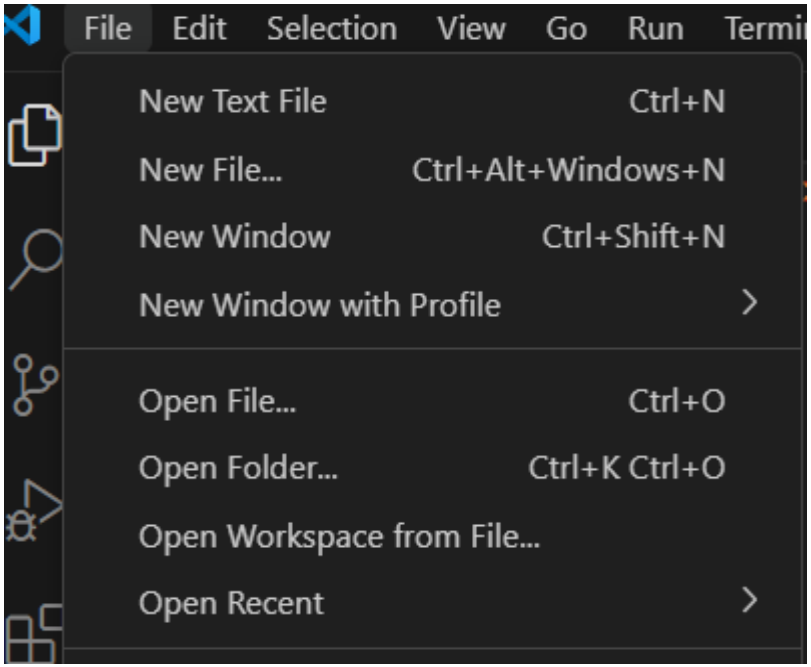
```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIz[redacted]",
  authDomain: "sopm-demo-project.firebaseapp.com",
  projectId: "sopm-demo-project",
  storageBucket: "sopm-demo-project.firebasestorage.app",
  messagingSenderId: "376242510484",
  appId: "1:376242510484:web:dfa79b94b538e1784bb281"
};

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

Note: This option uses the modular JavaScript SDK, which provides reduced SDK size.

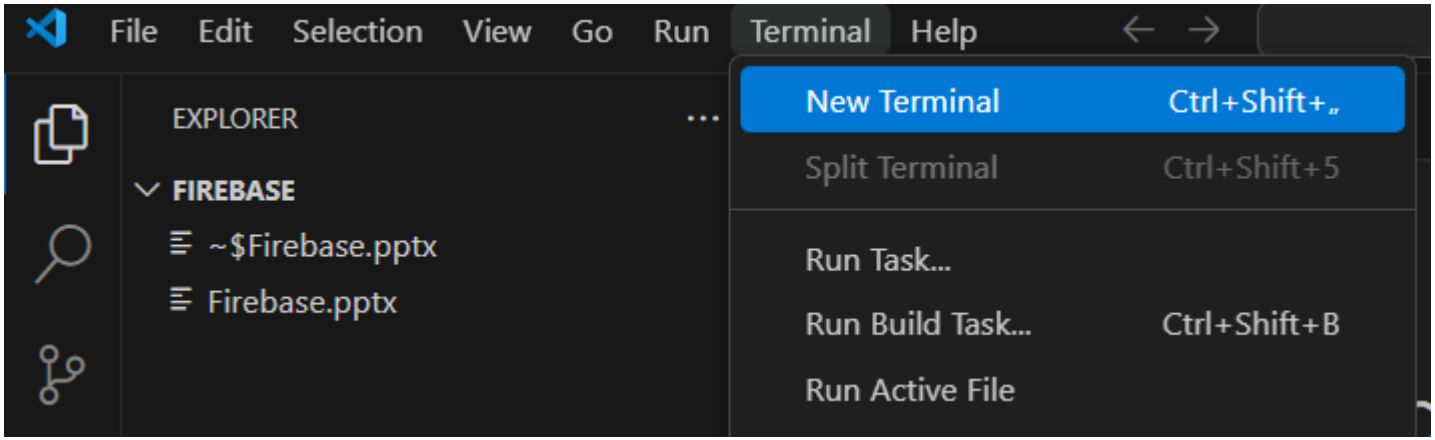
Learn more about Firebase for web: Get Started, Web SDK API Reference, Samples



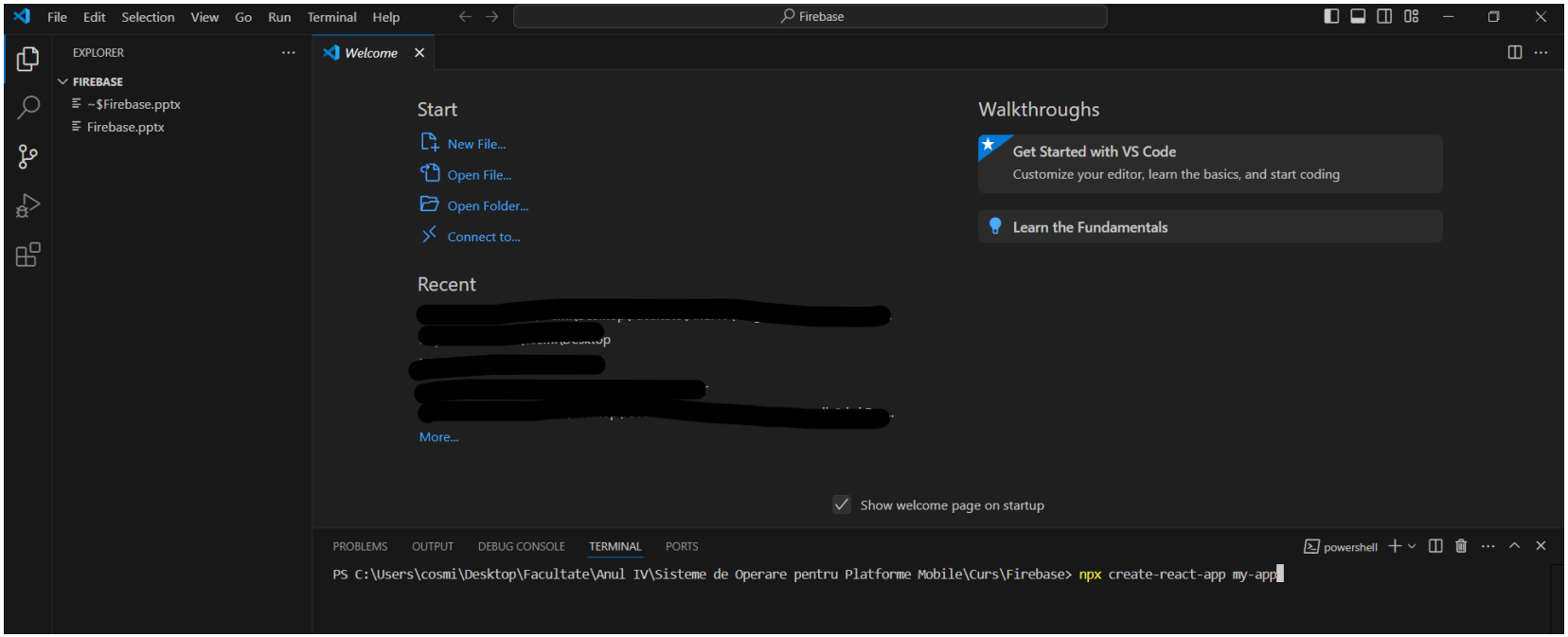
- Când am ajuns în punctul prezentat în imaginea de pe slide-ul anterior, va trebui să creăm și aplicația web propriu-zisă. Pentru asta, vom urma pașii ca și până acum.
- Deschidem un editor de cod (**Visual Studio Code – recomandat și folosit în această reprezentare**)
- Din **NavBar** vom selecta **File – Open Folder**, la fel ca în poza alăturată.
- Facem asta pentru a merge în ce director vrem noi și pentru a naviga mai ușor, să nu fim nevoiți să schimbăm directoarele din terminal.
- Selectăm folder-ul în care vreți să creați proiectul, nu contează care



- Când am selectat folder-ul dorit, ne vom întoarce în **NavBar**, de unde vom deschide un nou terminal, pentru a iniția proiectul.



- În partea inferioară a ecranului ni s-a deschis un terminal, aflat în directorul selectat de noi în pasul anterior.
- Aici va trebui să folosim următoarea comandă pentru a crea un nou proiect react: **npx create-react-app my-app**
- **ATENȚIE!** Trebuie să aveți instalat Node.js pentru asta.



- După ce am dat enter comenzii de mai sus, proiectul nostru se va încărca. Lucru ce va dura cam câteva minute, în funcție de specificațiile tehnice ale dispozitivului pe care îl folosim.

```
We suggest that you begin by typing:

  cd my-app
  npm start
```

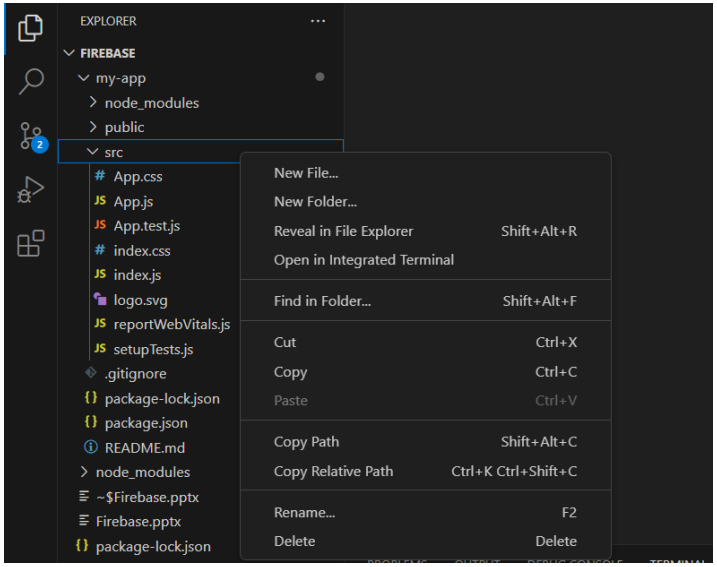
- Așa ne dăm seama că procesul este gata. După cum scrie și acolo, vom da cd my-app, dar **nu** vom da și **npm start.**

- După ce am dat cd my-app (ne-am mutat în directorul în care se află aplicația creată), vom instala dependințele necesare pentru a folosi Firebase. Asta vom face folosind comanda **npm install firebase**.

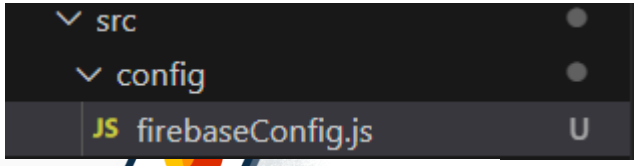
```
PS C:\Users\cosmi\Desktop\Facultate\Anul IV\Sisteme de Operare pentru Platforme Mobile\Curs\Firebase> cd my-app
PS C:\Users\cosmi\Desktop\Facultate\Anul IV\Sisteme de Operare pentru Platforme Mobile\Curs\Firebase\my-app> npm install firebase
```

- După ce am dat cd my-app (ne-am mutat în directorul în care se află aplicația creată), vom instala dependințele necesare pentru a folosi Firebase. Asta vom face folosind comanda **npm install firebase**.

- Acum, după ce am instalat tot ce era necesar, va trebui să adăugăm configurația oferită de Firebase.
- Vom merge în **my-app -> src -> New folder**. Creăm folder-ul cu numele config (se poate face și fără, dar așa este best pratice.



- În folder-ul conving, vom crea un nou fișier (**New File**), cu numele **firebaseConfig.js**



2 Add Firebase SDK

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 (Learn more):

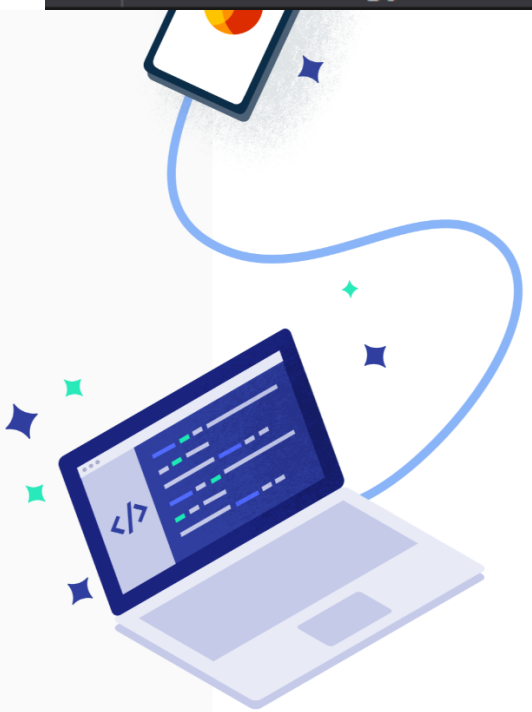
```
$ npm install firebase
```

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: "AIzaSyDuDR6Ijyfj3a-in5cG4o6jzzkntGX0OaI",
  authDomain: "sopm-demo-project.firebaseapp.com",
  projectId: "sopm-demo-project",
  storageBucket: "sopm-demo-project.firebasestorage.app",
  messagingSenderId: "376242510484",
  appId: "1:376242510484:web:dfa79b94b538e1784bb281"
};

// 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

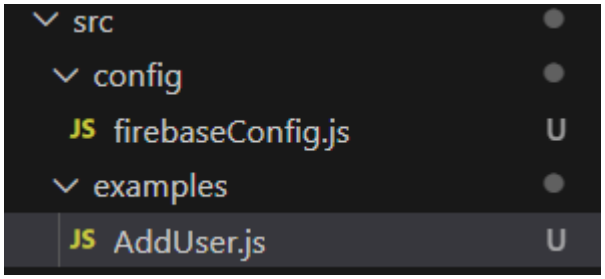


- Revenim în browser, în punctul în care am rămas ultima oară. Vom da copy la tot snippet-ul și îi vom da paste in fișierul creat, **firebaseConfig.js**

- Deoarece noi vom folosi baza de date statică din Firebase, o vom importa, (și anume Firestore) cât și exporta (sub formă de variabilă), ca să poată să fie folosită în aplicație. Astfel, vom adăuga următoarele linii de cod în firebaseConfing.js (sunt cele marcate cu roșu)

```
// Import the functions you need from the SDKs you
need
import { initializeApp } from "firebase/app";
import { getFirestore } from "firebase/firestore";
// 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: "AIzaSyDuDR6Ijyfj3a-in5cG4o6jzzkntGX0OaI",
  authDomain: "sopm-demo-project.firebaseapp.com",
  projectId: "sopm-demo-project",
  storageBucket: "sopm-demo-
project.firebasestorage.app",
  messagingSenderId: "376242510484",
  appId: "1:376242510484:web:dfa79b94b538e1784bb281"
};

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

- În acest moment, Firebase-ul, implicit Firestore-ul, sunt configurate cu aplicația. Acum, pentru a testa și demonstra, vom face câteva exemple. Astfel, la fel ca până acum vom crea un nou folder și un nou fișier: **click dreapta pe src -> New Folder -> examples -> New File -> AddUser.js** (din nou, se poate face fără acest folder, dar good practice, este să păstrăm proiectul cât mai organizat și bine structurat.)
- Aici vom insera următorul cod: (se regăsește pe slide-ul următor)



```
import React, { useState } from 'react';
import { db } from '../config/firebaseConfig';
import { collection, addDoc } from
'firebase/firestore';

function AddUser() {
  const [nume, setNume] = useState('');
  const [varsta, setVarsta] = useState('');

  const adaugaUtilizator = async (e) => {
    e.preventDefault();
    try {
      // Adăugăm document nou în colecția
'utilizatori'
      const docRef = await addDoc(collection(db,
'utilizatori'), {
        nume: nume,
        varsta: parseInt(varsta),
        dataAdaugare: new Date()
      });
      console.log("Utilizator adăugat cu ID: ",
docRef.id);
      setNume('');
      setVarsta('');
    } catch (error) {
      console.error("Eroare la adăugarea
utilizatorului: ", error);
    }
  };
```

```
return (
    <form
onSubmit={adaugaUtilizator}>
      <input
        value={nume}
        onChange={(e) =>
setNume(e.target.value)}
        placeholder="Nume"
      />
      <input
        type="number"
        value={varsta}
        onChange={(e) =>
setVarsta(e.target.value)}
        placeholder="Vârstă"
      />
      <button type="submit">Adaugă
Utilizator</button>
    </form>
  );
}

export default AddUser;
```

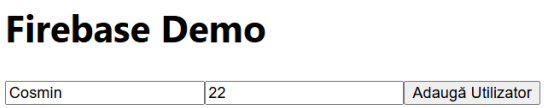
- Ultimele două slide-uri conțin codul pentru **AddUser.js**. Cel din urmă vine în continuarea primului cod, e distanță de o singură linie, în același fișier.
- Urmează să testăm codul, să vedem dacă avem rezultatele pe care le dorim.
- Pentru a folosi componenta creată, va trebui să o importăm în **App.js**. Pentru asta, vom merge în src/App.js, vom șterge tot ce se află acolo și vom da paste următorului cod:

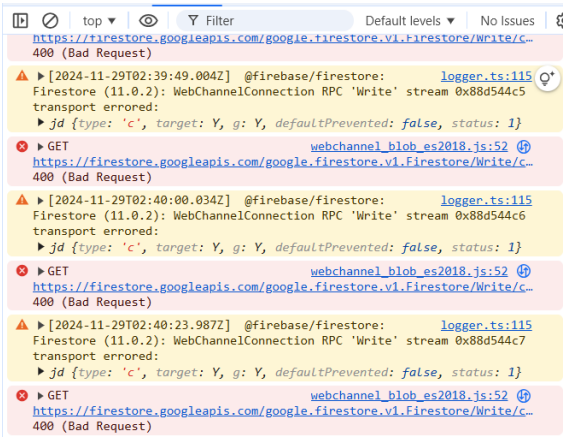
```
import AddUser from
'./examples/AddUser';

function App() {
  return (
    <div className="App">
      <h1>Firebase Demo</h1>
      <AddUser />
    </div>
  );
}

export default App;
```

- Mergem în terminal, folosim comanda npm start și primim următorul mesaj și suntem redirecționați către pagina creată:

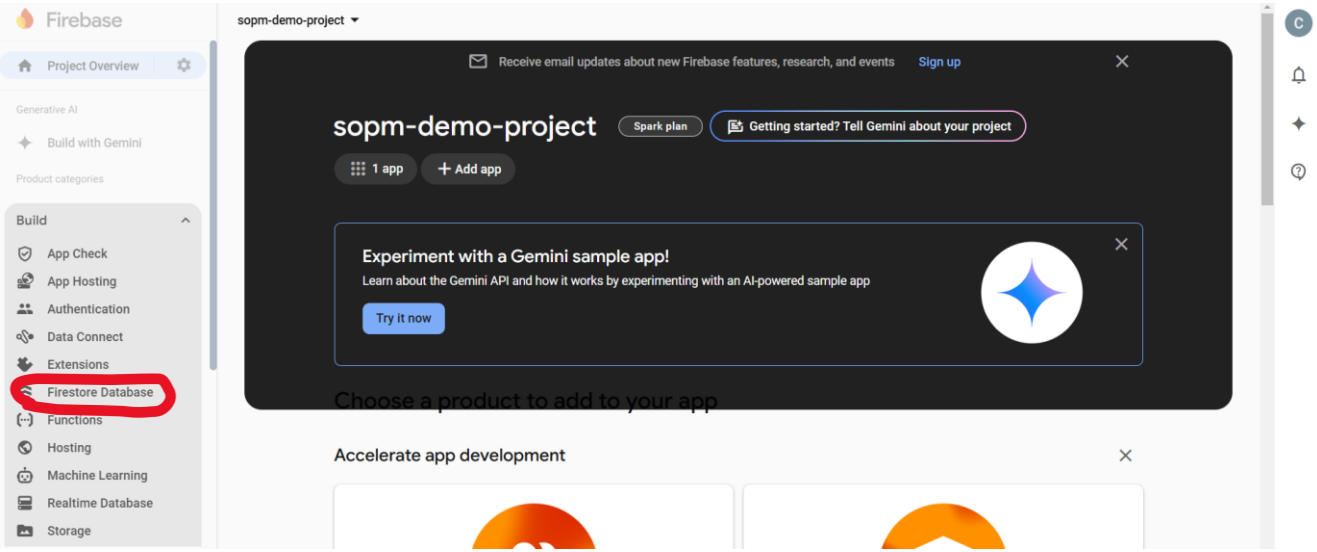
```
Compiled successfully!

You can now view my-app in the browser.

  Local:            http://localhost:3000
  On Your Network:  [redacted]

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

- Aspectual, pagina arată cum ne doream, totuși, trebuie să vedem dacă funcționează în parametri optimi, și anume să trimită datele introduse către baza de date. Completăm câmpurile, iar ulterior, pentru a verifica ce se întamplă cu datele, folosim F12 pentru a deschide consola și a primi informații ce nu sunt afișate în pagină.

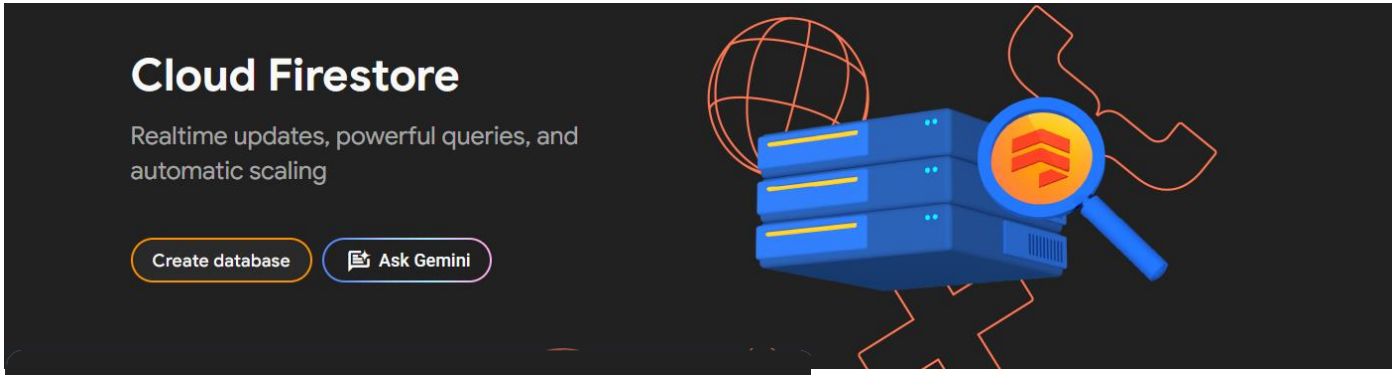


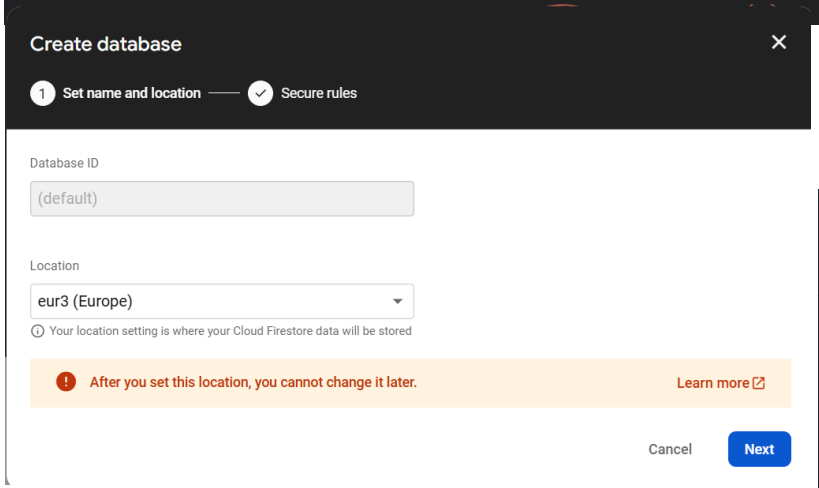


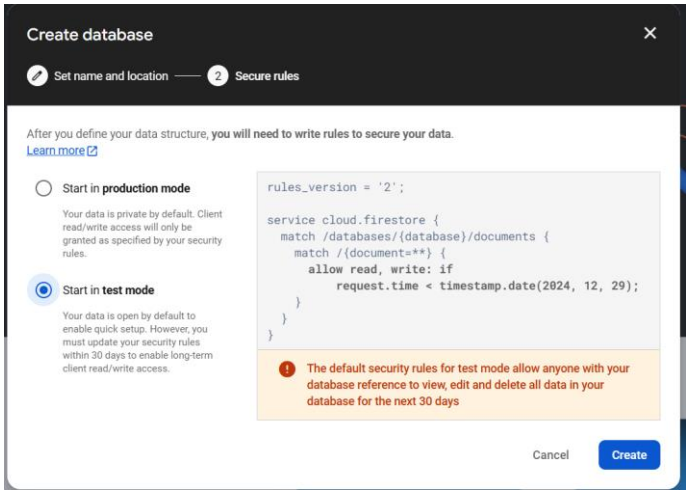
- Vedem că avem o eroare, ei bine motivul acesteia este pentru că nu am configurat regulile bazei de date, și anume: ce operații CRUD se pot face pe ea, cine le poate face, în ce condiții se pot face.
- Acest aspect e unul foarte important, acesta și fiind motivul pentru care a fost lăsată această eroare aici.
- Pentru a remedia, ne vom întoarce în browser, în ultimul pas pe care l-am făcut (după ce am apăsat Continue to console). În stânga vom avea drop-down menu-ul **Build**, pe care îl vom apăsa, iar de acolo vom selecta **Firestore Database**.



- **Create database** – selectăm regiunea Europa – **Next** – **Start in test mode - Create**

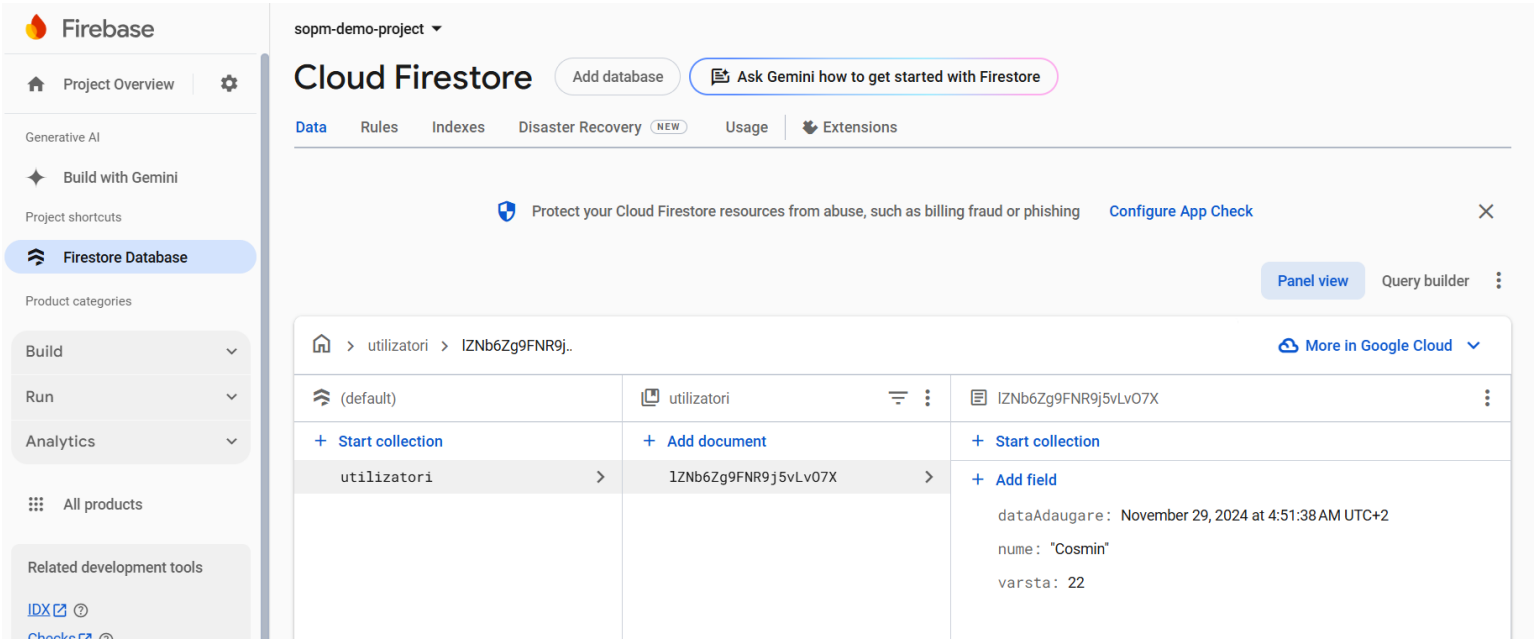






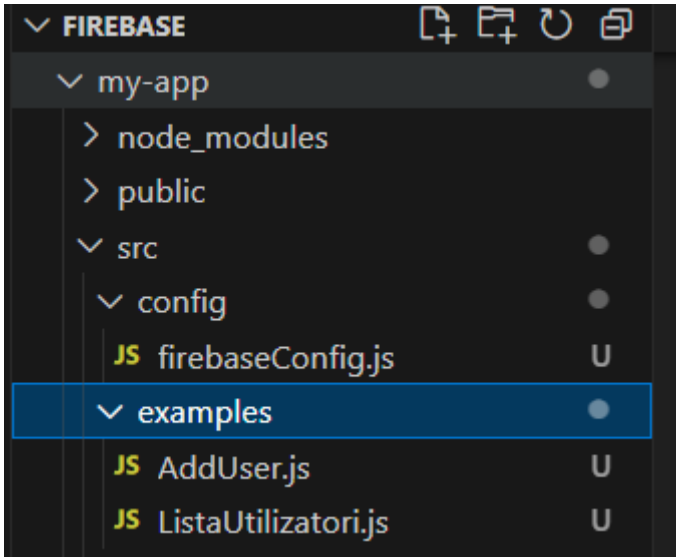
- Regulile din test mode pot fi folosite pentru demonstrația noastră, astfel că după ce am trecut prin acești pași, ne vom întoarce în aplicația noastră web, vom da un refresh și vom încerca să trimite datele, din nou, către baza de date.

- De această dată, după ce am introdus date în câmpurile cu Nume și Vârstă, primim un mesaj de confirmare, setat din cod, cum că totul a mers cum a trebuit. Cum putem verifica asta? Tot din baza de date.
- Ne întoracem în locul în care am fost redirecționați după ce am creat baza de date și am setat regulile, iar acolo, dupa un refresh, ar trebui să ne apară datele, în secțiunea **Data**.



- Acesta este modul în care datele s-au stocat în baza de date, sub colecția de **utilizatori** avem un utilizator adăugat sub acela ID, generat de funcția **docRef.id**, iar sub forma unei alte colecții, avem datele introduse de noi. Aceasta este o instrucție de tip **POST**.
- Totuși, utilizatorii acestei aplicații nu au cum să acceseze baza de date, așa că trebuie să existe o modalitate și pentru ei să vadă datele. Asta se poate face printr-o instrucțiune **GET**.

- Pentru a implementa o componentă care să preia și să afișeze datele, va trebui, în primul rând să o creăm. Astfel că ne vom întoarce în folder-ul deja existent, example, iar acolo vom crea fișierul **ListaUtilizatori.js**
- Codul pentru această componentă se află pe slide-ul următor.



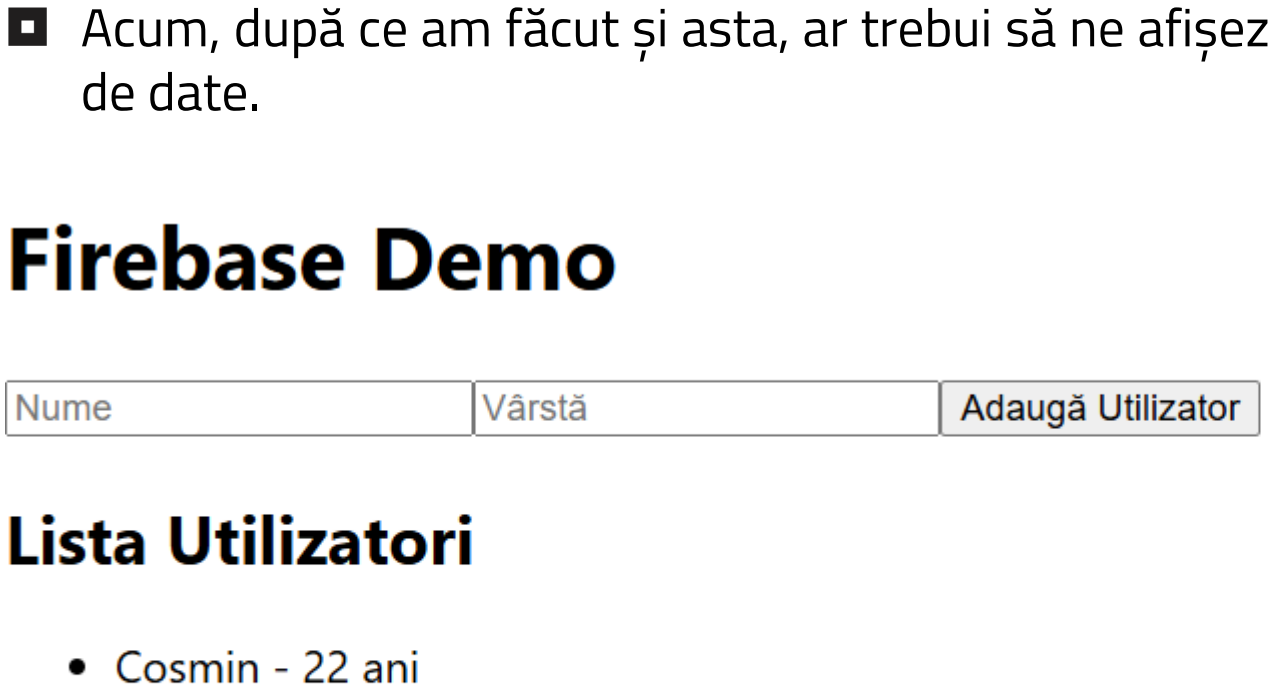
```
import React, { useEffect, useState } from 'react';
import { db } from '../config/firebaseConfig';
import { collection, getDocs } from 'firebase/firestore';

function ListaUtilizatori() {
  const [utilizatori, setUtilizatori] = useState([]);

  useEffect(() => {
    const preiaUtilizatori = async () => {
      try {
        const querySnapshot = await getDocs(collection(db,
'utilizatori'));
        const listaUtilizatori = [];
        querySnapshot.forEach((doc) => {
          listaUtilizatori.push({
            id: doc.id,
            ...doc.data()
          });
        });
        setUtilizatori(listaUtilizatori);
      } catch (error) {
        console.error("Eroare la preluarea utilizatorilor: ",
error);
      }
    };
```

```
 preiaUtilizatori();
  }, []);

  return (
    <div>
      <h2>Lista Utilizatori</h2>
      <ul>
        {utilizatori.map((utilizator) => (
          <li key={utilizator.id}>
            {utilizator.nume} -
{utilizator.varsta} ani
          </li>
        ))}
      </ul>
    </div>
  );
}

export default ListaUtilizatori;
```

- La fel ca în snippet-ul anterior, codul trebuie pus în același fișier, implicit în **ListaUtilizatori.js**, despărțit doar printr-o linie goală.
- După ce am completat codul, va trebui să revenim în fișierul **App.js** (src/App.js), care practic este pagina principală și noi ne afișăm componentele prin intermediul ei, și să instanției noua componentă creată **(ListaUtilizatori.js)**. Facem asta la fel cum am făcut cu **AddUser.js**

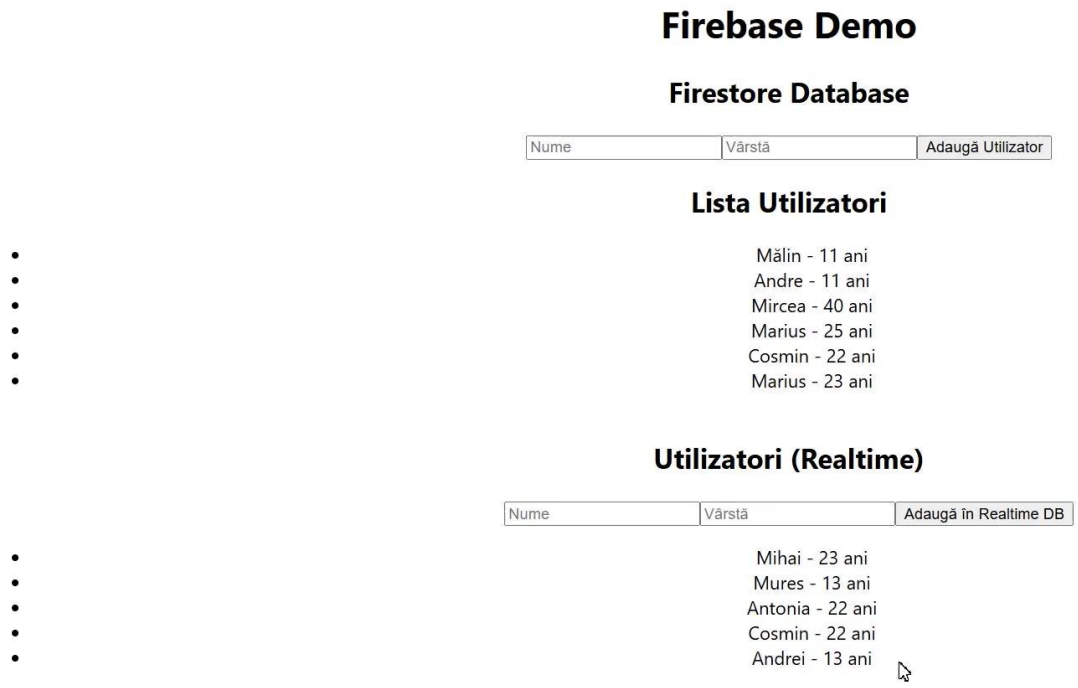
```
import AddUser from './examples/AddUser';
import ListaUtilizatori from './examples/ListaUtilizatori';

function App() {
  return (
    <div className="App">
      <h1>Firebase Demo</h1>
      <AddUser />
      <ListaUtilizatori />
    </div>
  );
}

export default App;
```

- Noile linii de cod, ce trebuie adăugate, au fost marcate cu roșu, pentru o depistare mai ușoare.
- Acum, după ce am făcut și asta, ar trebui să ne afișeze datele din baza de date.



- Ne afișează exact după modul în care programat să facem asta, iar de aici putem trage concluziile că am făcut o treabă bună. Pentru a fi siguri că totul este în regulă, putem adăuga și mai multe date, să vedem ce se întâmplă.

- Totuși, ce putem observa din asta, datele nu se actualizează în timp real, fiind nevoie de un refresh pentru a vedea datele proaspăt adăugate.
- Pentru ca datele să fie afișate în timp real, imediat ce ajung în baza de date, va trebui să schimbă serviciul folosit, și să folosim **Firebase Realtime Database**. Astfel vom avea parte de următoarele:



3. CONCLUZII

Firebase

Google Cloud

- În momentul de față, există provideri de servicii Cloud sub orice formă, pentru orice infrastructură dorită. În cazul în care nu aveți nevoie de un serviciu foarte specific, pe care să îl controlați și gestionați în proporție de 100%, nu are rost să vă chinuiți și să încercați să faceți ceva de la 0.
- Totodată, în următoarele slide-uri se vor parcurge câteva puncte avantaje și câteva dezavantaje pentru două dintrele cele mai mari providere Cloud, și anume **AWS** și **Google Cloud Platform**.

- Pentru **AWS**:
- Cea mai largă gamă de servicii oferite pe piață:
  - AWS oferă servicii și pentru cele mai specifice assignment-uri, iar calitatea acestora sunt foarte calitative.
  - Acesta poate acoperi toate nevoile d.p.d.v. al infrastructurii, indiferent de mărimea companiei.
- Rețea globală extinsă:
  - AWS are cea mai mare rețea de centre de date distribuite la nivel mondial (24 de regiuni și peste 100 de zone de disponibilitate).
  - Permite utilizatorilor să implementeze aplicații aproape de utilizatorii finali pentru performanțe maxime
- Scalabilitate uriașă:
  - Cu AWS se poate începe cu resurse minime și se poate scala automat pe măsură ce nevoile afacerii cresc.
  - Exemple: Auto-scaling pentru EC2 sau scalabilitatea automată a bazelor de date RDS.

- Pentru **GCP**:
- Interfețele serviciilor se remarcă prin ușurința de utilizare
  - GCP este cunoscută pentru interfața sa prietenoasă și ușor de utilizat, care permite gestionarea rapidă și eficientă a resurselor, reducând timpul necesar pentru a învăța platforma.
- Integrarea cu ecosistemul Google:
  - GCP se integrează perfect cu alte servicii Google, cum ar fi Gmail, Google Drive și Google Analytics.
  - Este ușor de utilizat pentru persoanele și organizațiile deja familiare cu produsele Google.
- Inteligență Artificială și Machine Learning avansate:
  - GCP oferă instrumente AI și ML de top, cum ar fi AutoML și Vertex AI, care permit dezvoltarea de modele personalizate fără a necesita cunoștințe avansate de machine learning.
  - Este ideal pentru aplicații ce implică procesarea imaginilor, NLP (procesare limbaj natural) sau analize predictive.

- Punându-le cap la cap, ambii provideri sunt variante de top, de departe cele mai puternice existente pe piață.
- Ce ar trebui să aleagă o persoană care vrea să integreze servicii de Cloud Computing în proiectele lor?
  - Nu există un răspuns corect sau greșit. Fiecare provider vine cu propriile avantaje. Dar, principiile din spate sunt aceleași, bazele networking-ului, concepte de securitate și arhitecturi de microservicii.
  - Un utilizator le poate încerca pe ambele, fiecare dintre cele două providere au programe gratuite de încercare și experimentare ale serviciilor pe care le pun la îndemână.

Google Cloud

VĂ MULȚUMESC PENTRU ATENȚIE!

- Materiale bibliografice și surse de aprofundare:
  - firebase.google.com/docs
  - firebase.google.com/docs/firestore?hl=en
  - firebase.google.com/docs/database
  - Sequeira, Anthony. *AWS Certified Cloud Practitioner: Domain 1: Cloud Concepts*