Rendszerterv

Medical Toolbox

Msc Önlab 2

Készítette: Pásztori Péter Benjámin

Tartalom

[Feladatkiírás 4](#_Toc184163637)

[Felhasználói jogkörök: 4](#_Toc184163638)

[Alkalmazás és Architektúra 6](#_Toc184163639)

[Főbb Kódelemek 6](#_Toc184163640)

[1. Hitelesítés és jogosultságok 6](#_Toc184163641)

[2. Adatkezelés 7](#_Toc184163642)

[3. Tárolás 7](#_Toc184163643)

[4. Middleware használata 8](#_Toc184163644)

[Frontend: Next.js 8](#_Toc184163645)

[A frontend felelős a felhasználói interakciók és az alkalmazás dinamikus megjelenítéséért. A Next.js keretrendszer a React-ökorendszer erejét használja a szerveroldali rendereléshez és a kliensoldali funkcionalitáshoz. 8](#_Toc184163646)

[ Stílus: Tailwind CSS biztosítja az egységes és reszponzív megjelenést. 8](#_Toc184163647)

[ Kommunikáció: A backendgel való kapcsolat GraphQL API-n keresztül valósul meg. 8](#_Toc184163648)

[Frontend Dokumentáció 8](#_Toc184163649)

[Bevezetés 8](#_Toc184163650)

[Fontosabb fájlok részletesebb leírással: 8](#_Toc184163651)

[auth/resource.ts 8](#_Toc184163652)

[storage/resource.ts 9](#_Toc184163653)

[data/resource.ts 9](#_Toc184163654)

[backend.ts 10](#_Toc184163655)

[További fájlok: 10](#_Toc184163656)

[middleware.ts 10](#_Toc184163657)

[utils/amplify-utils.ts 10](#_Toc184163658)

[context/AdminContext.tsx 10](#_Toc184163659)

[components/AuthenticatorClient.tsx 11](#_Toc184163660)

[components/ClientSetupContextProvider.tsx 11](#_Toc184163661)

[components/Image.tsx 11](#_Toc184163662)

[components/NavBar.tsx 11](#_Toc184163663)

[components/ProductCreate.tsx 12](#_Toc184163664)

[components/ProductEdit.tsx 12](#_Toc184163665)

[components/ProductItem.tsx 12](#_Toc184163666)

[components/ProductItemControls.tsx 12](#_Toc184163667)

[app/layout.tsx 13](#_Toc184163668)

[app/page.tsx 13](#_Toc184163669)

[app/signin/page.tsx 13](#_Toc184163670)

[app/admin/product-edit/[id]/page.tsx 13](#_Toc184163671)

[app/admin/product-create/page.tsx 14](#_Toc184163672)

[actions/revalidate.ts 14](#_Toc184163673)

[components-ui generálása: 14](#_Toc184163674)

[Funkció 14](#_Toc184163675)

[Működés 14](#_Toc184163676)

[Backend: Amplify Gen2 15](#_Toc184163677)

[Technikai Részletek 15](#_Toc184163678)

[Főbb Backend Funkciók: 15](#_Toc184163679)

[Adatmodell 15](#_Toc184163680)

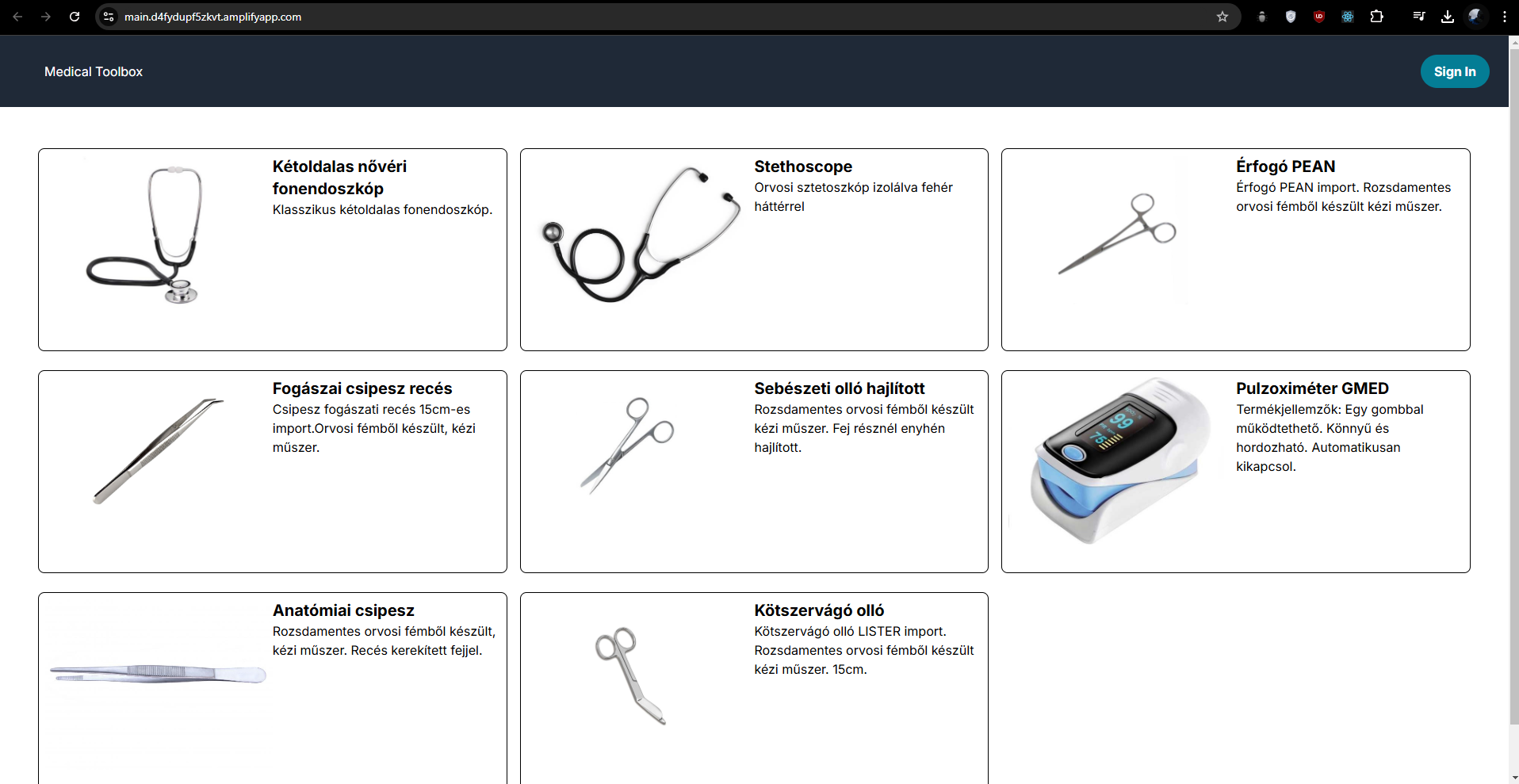
[DynamoDB Táblák 15](#_Toc184163681)

# Feladatkiírás

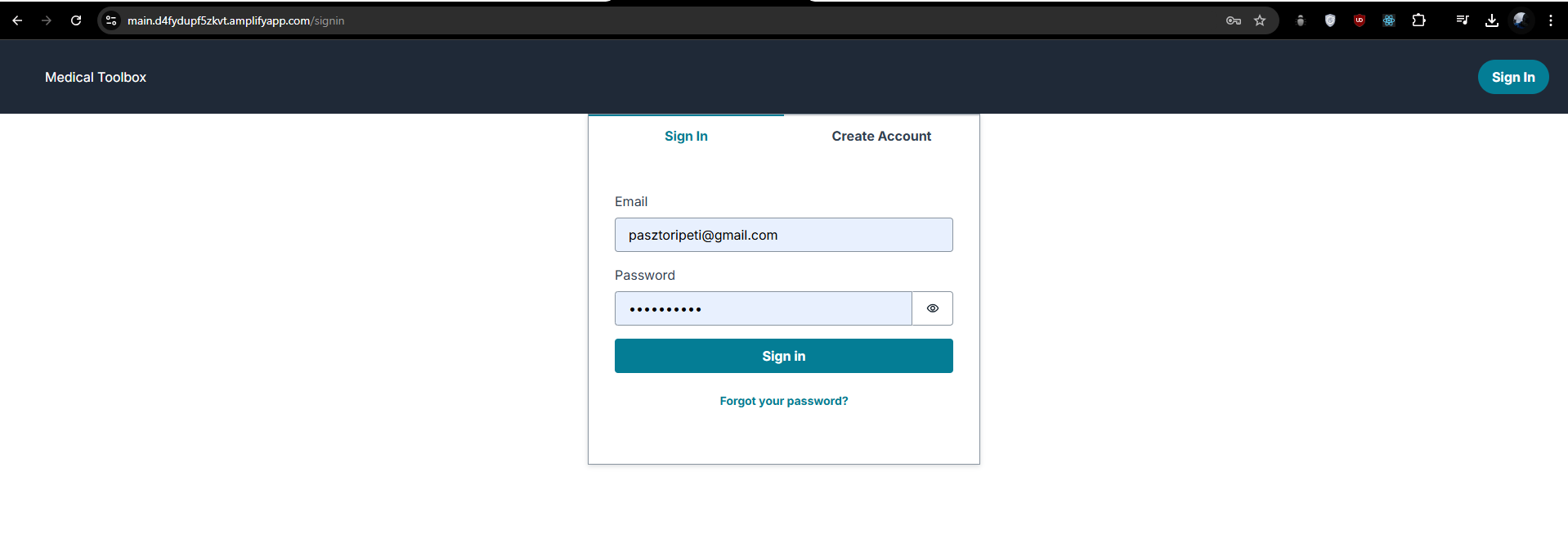
Az Medical Toolbox egy modern, AWS-alapú e-kereskedelmi webalkalmazás, amely lehetővé teszi a termékek megtekintését, valamint adminisztrációs műveletek végrehajtását, például termékek kezelését.

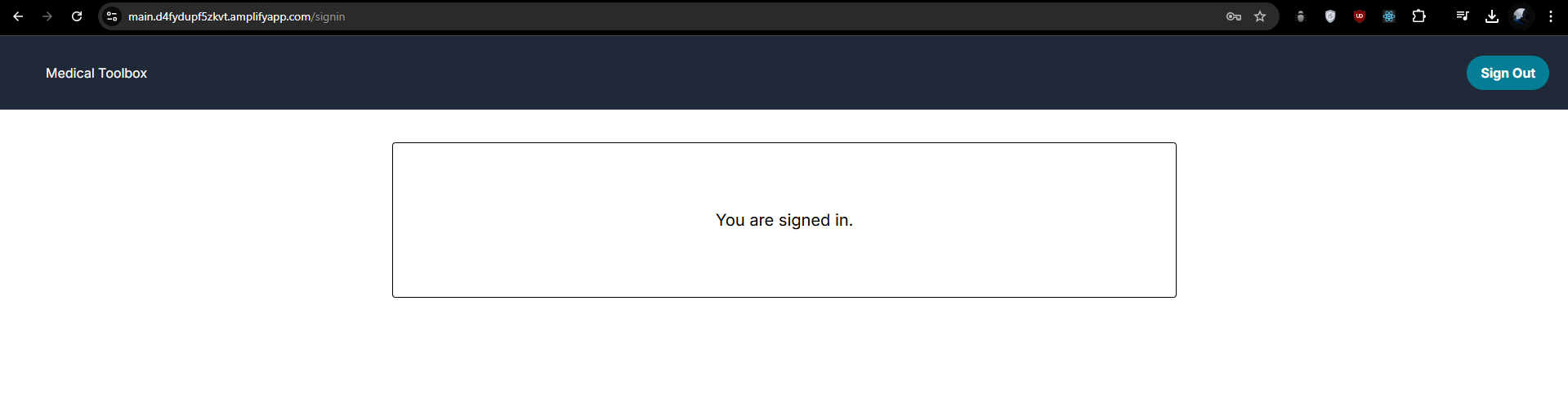
# Felhasználói jogkörök:

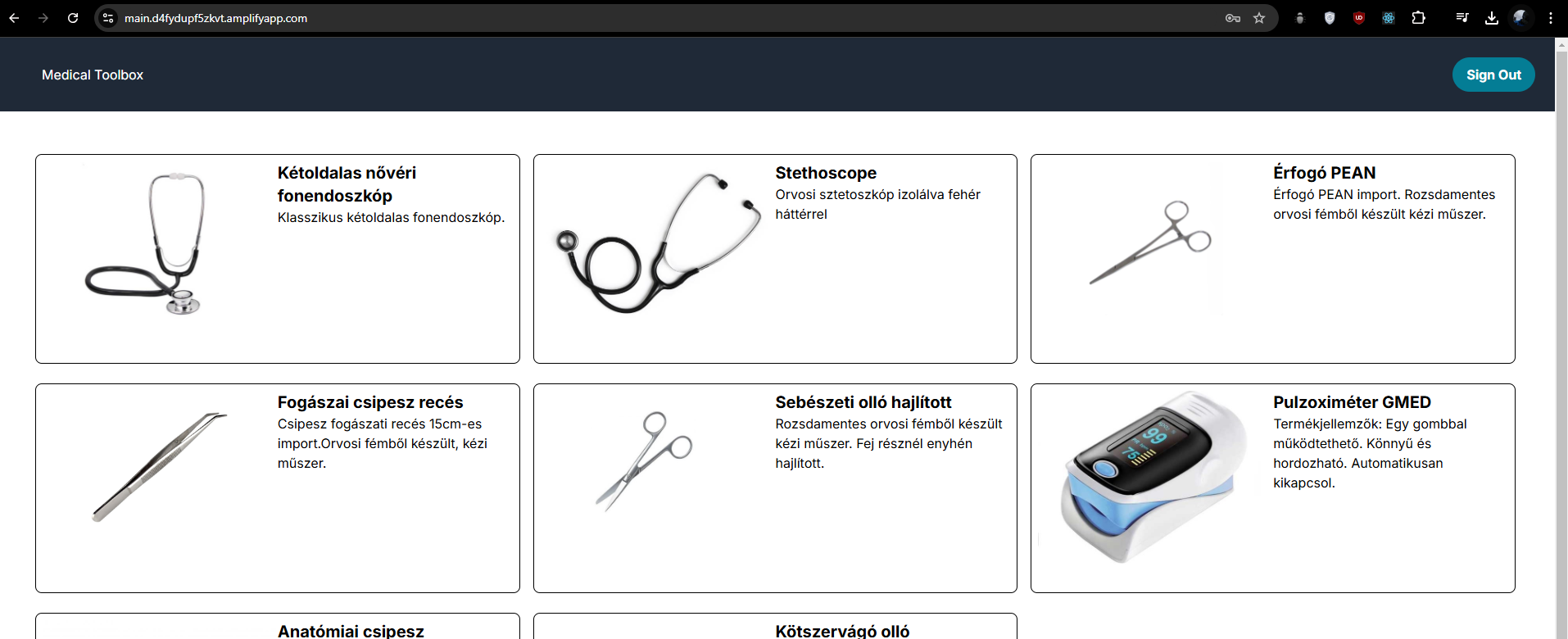
* Vendég felhasználók: Termékek böngészése.



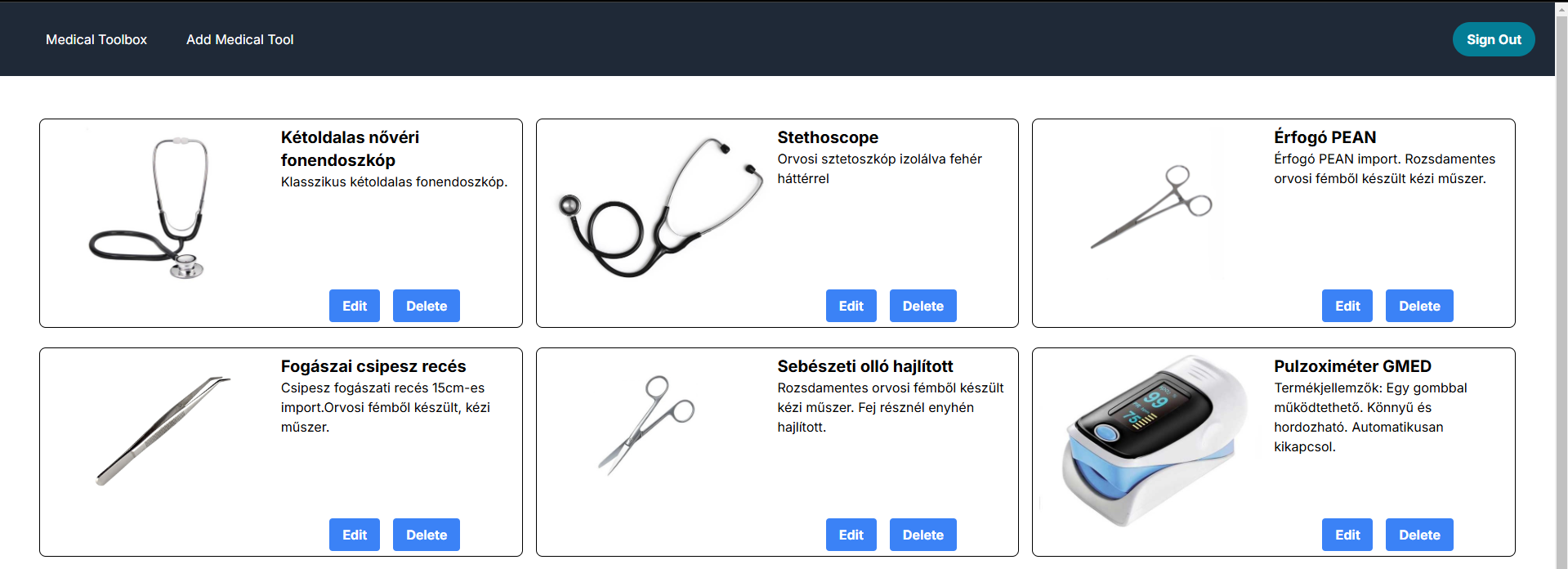
* Regisztrált felhasználók: Termékek böngészése.

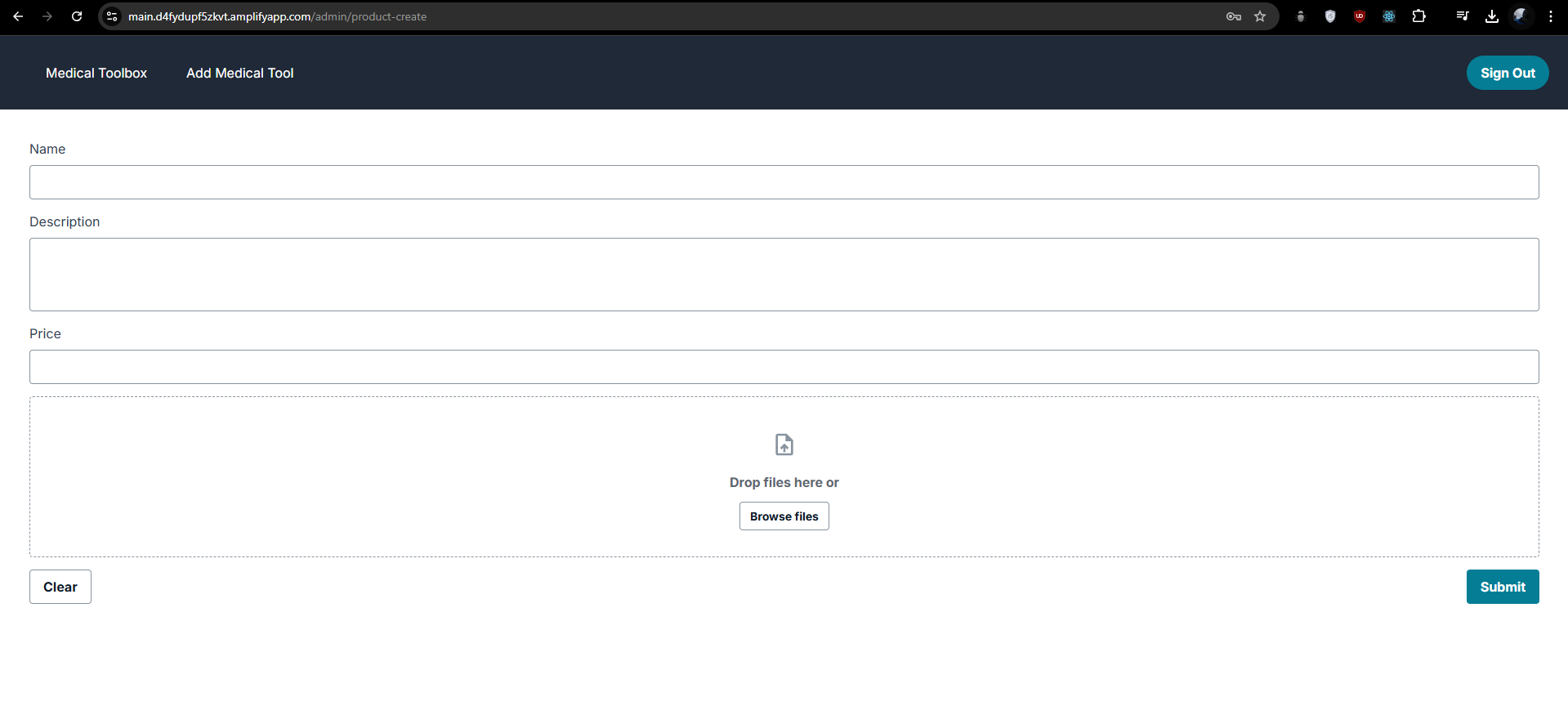


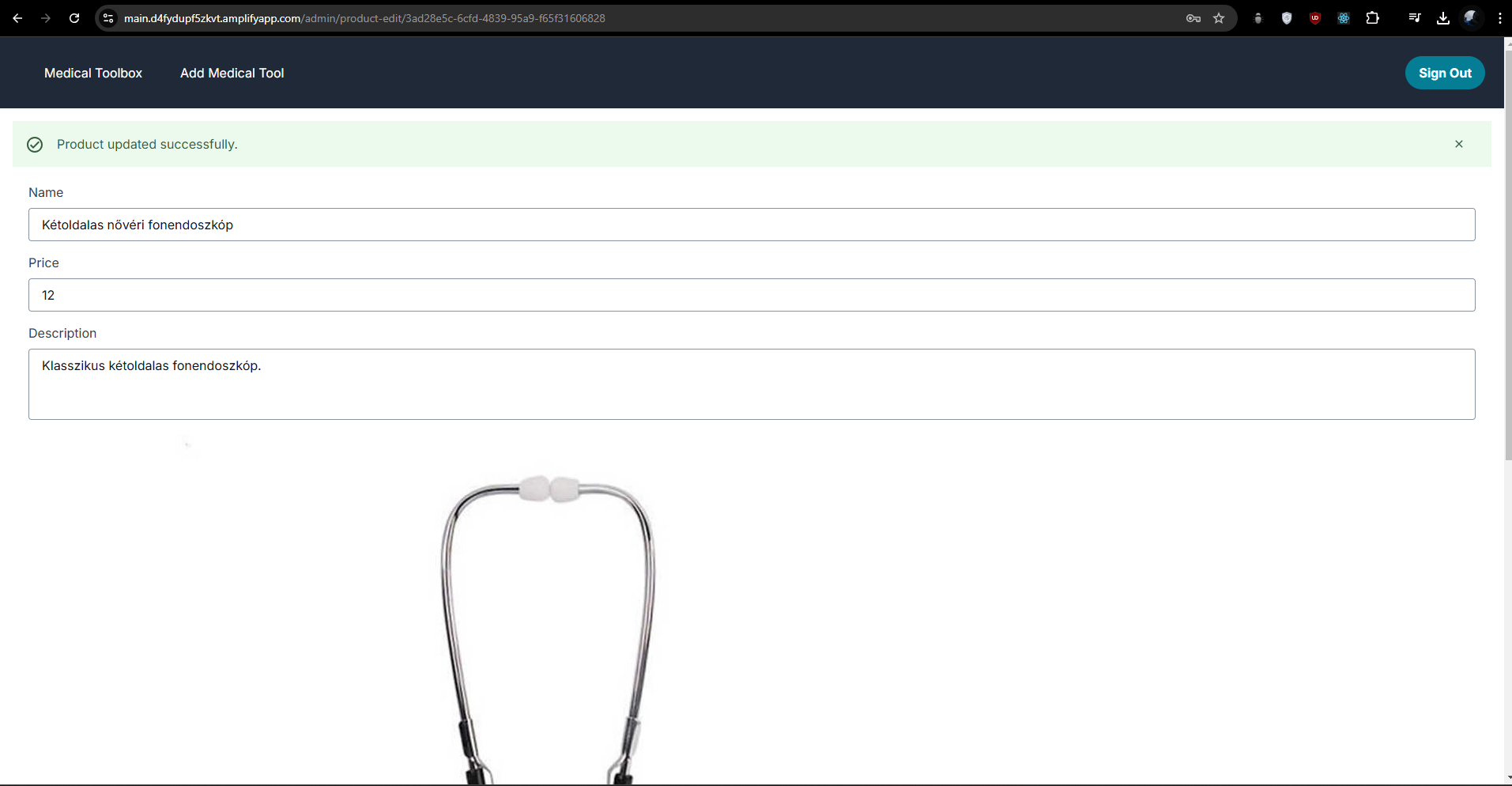




* Adminisztrátorok: Termékek hozzáadása, frissítése, törlése, Termékekhez kapcsolódó képek feltöltése az Amazon S3-ba.







# Alkalmazás és Architektúra

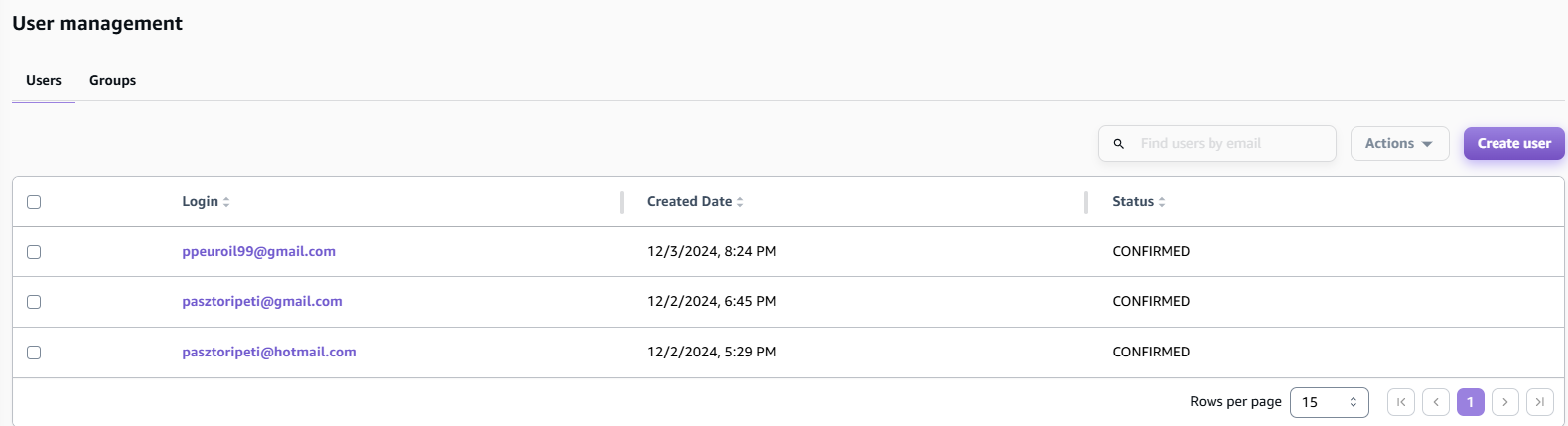
Az alkalmazás modern webfejlesztési megközelítést alkalmaz az alábbi főbb technológiák integrálásával:

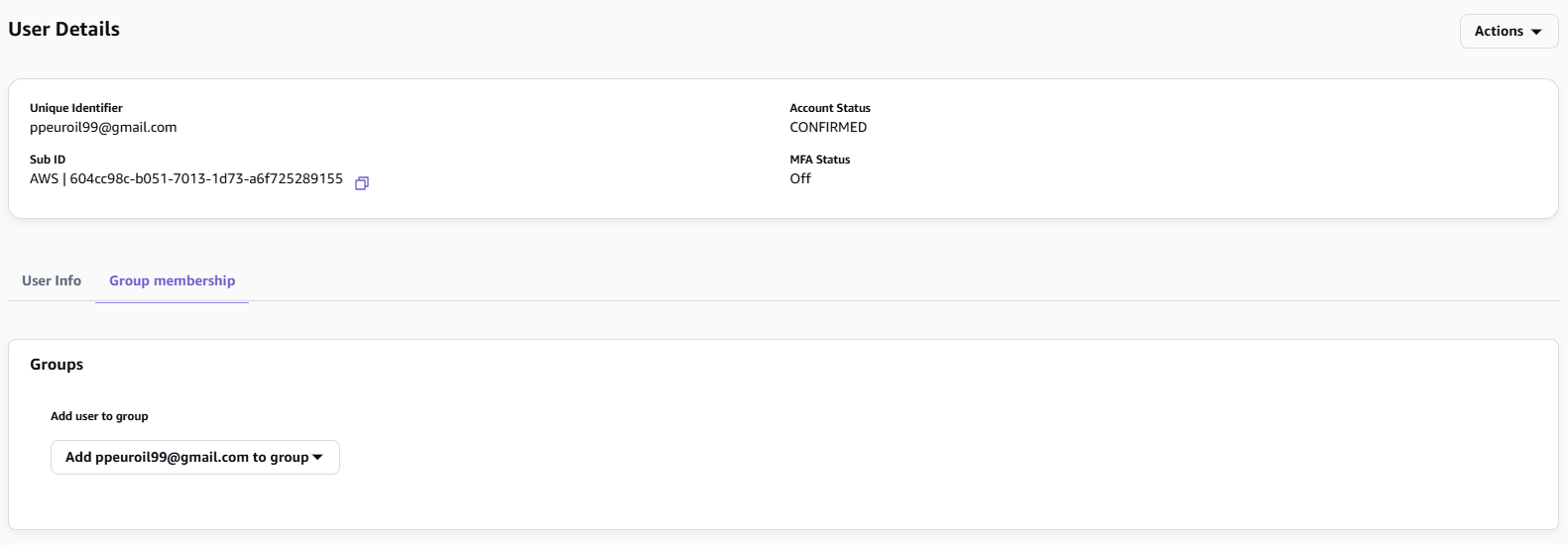
* Frontend: Next.js a dinamikus, gyors és reszponzív felhasználói felülethez.
* Backend és szolgáltatások: AWS Amplify platform a felhőalapú funkciók egyszerű kezeléséhez.
* Adattárolás és adatkezelés: Amazon DynamoDB és S3 a strukturált adattároláshoz és a médiafájlok kezeléséhez.
* Autentikáció: Amazon Cognito a felhasználók biztonságos hitelesítéséhez.

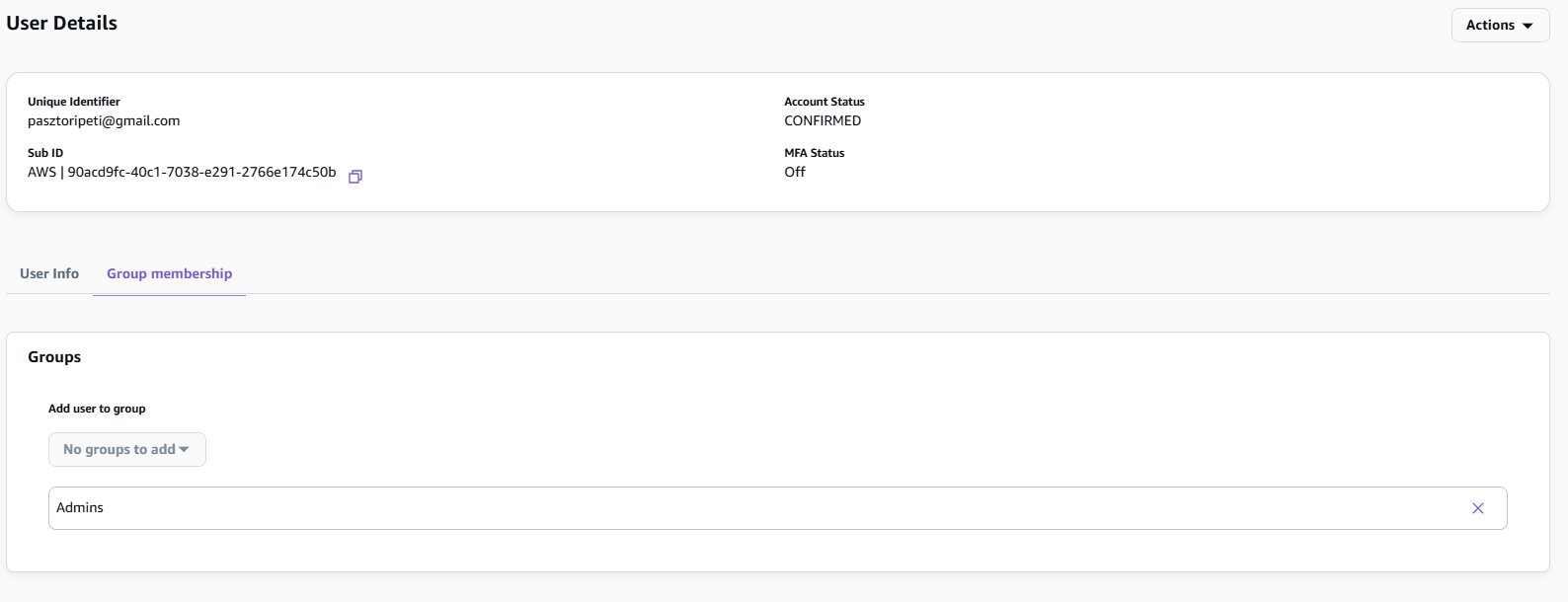
# Főbb Kódelemek

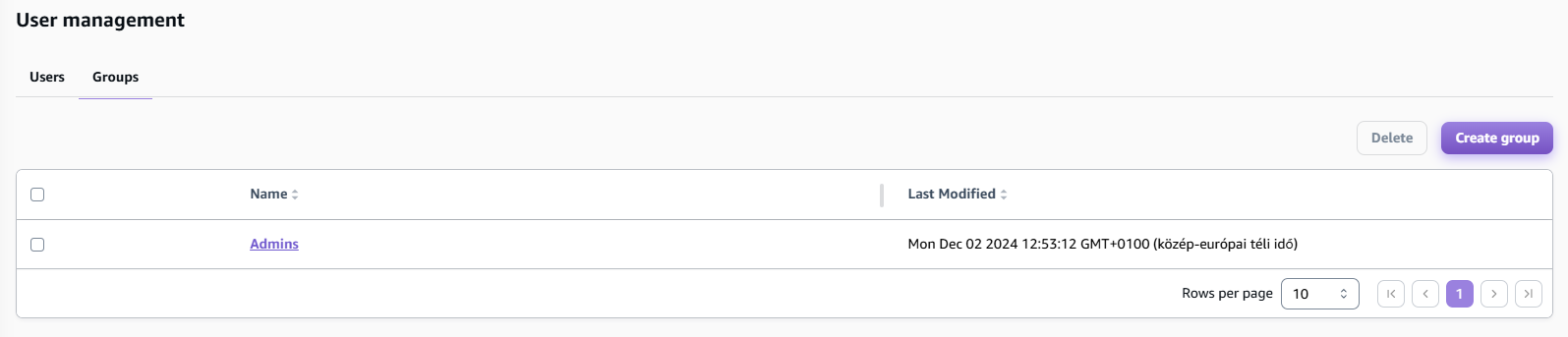
## 1. Hitelesítés és jogosultságok

* Amazon Cognito használata:
  + "Admins" csoport a rendszergazdai jogosultságokhoz.
  + E-mail alapú hitelesítés.



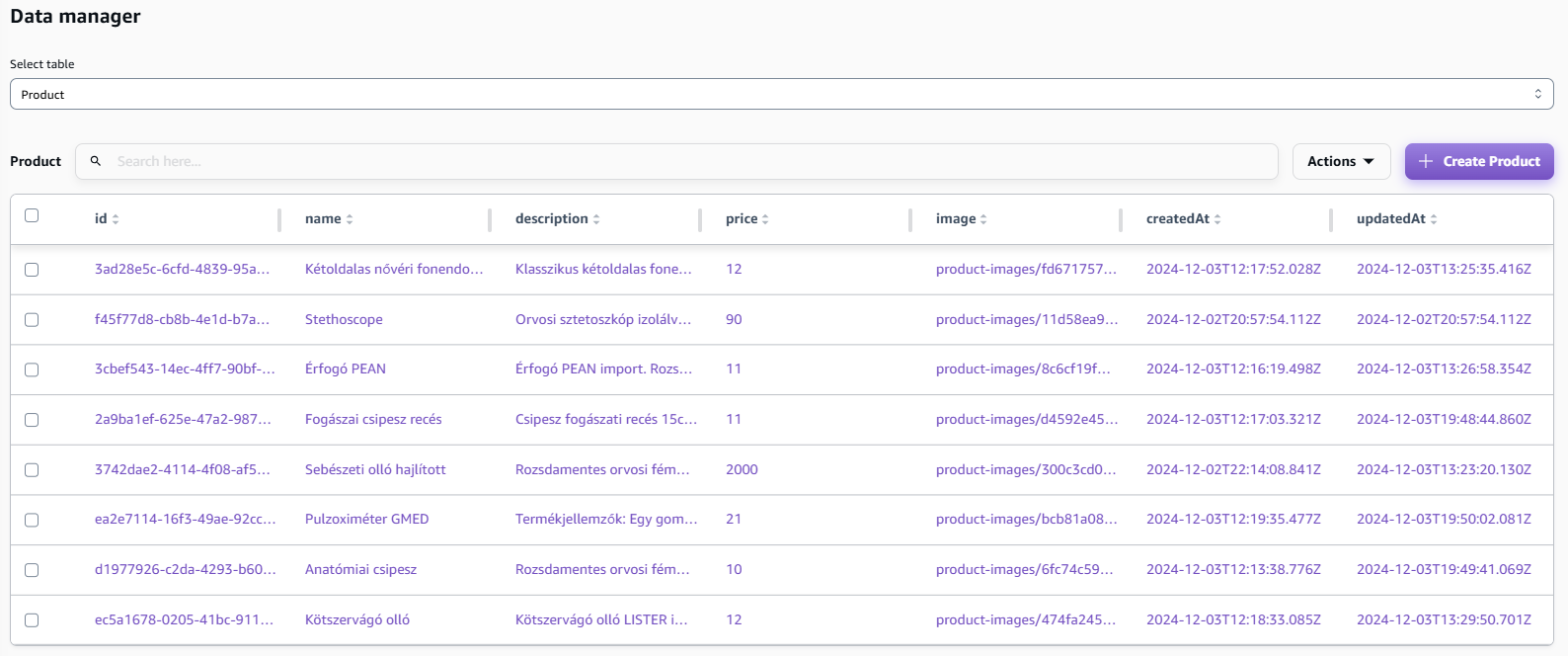






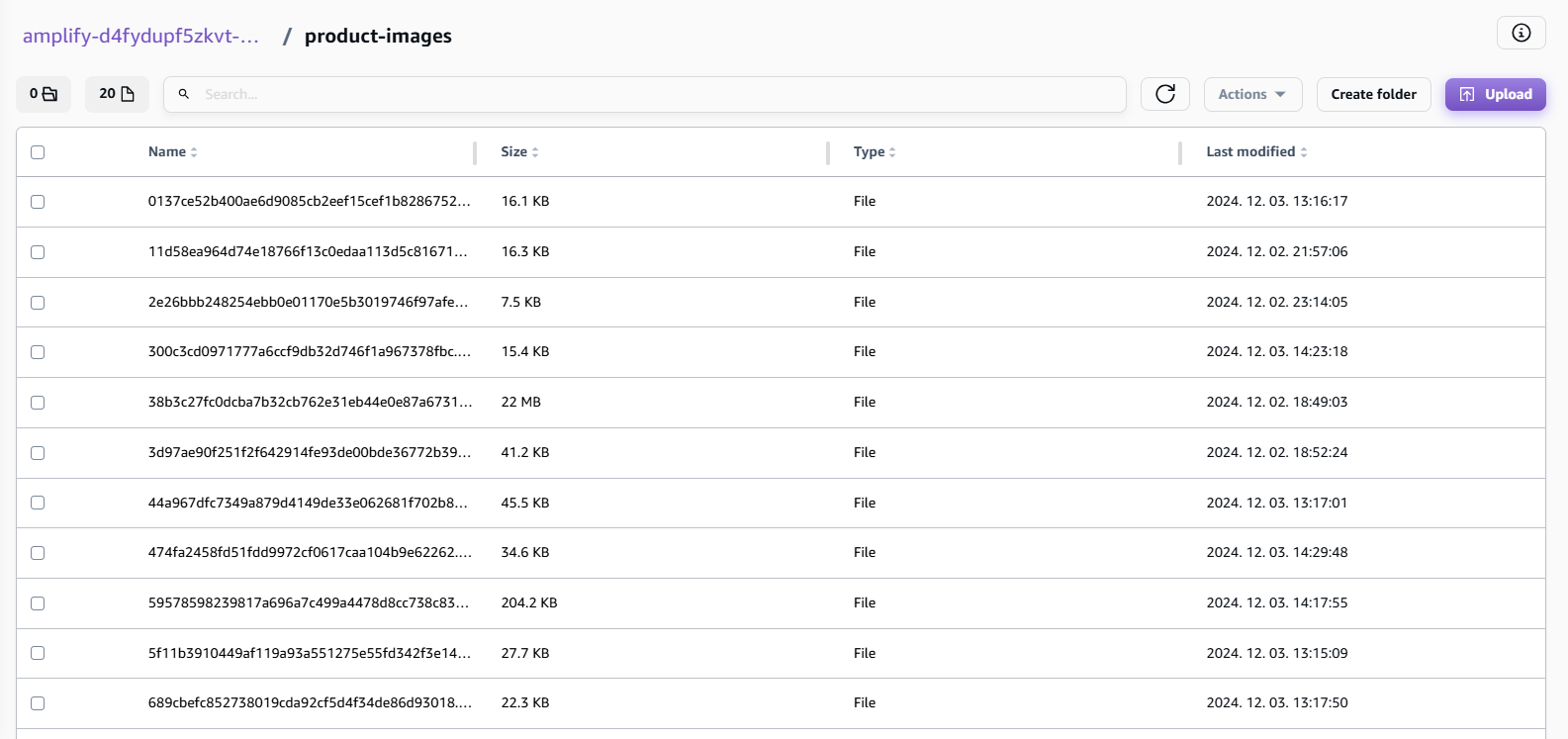
## 2. Adatkezelés

* DynamoDB: A termékek és felhasználói adatok tárolására.
* GraphQL sémák: Az adatmodellek és hozzáférések meghatározása.



## 3. Tárolás

* Amazon S3:
  + Képek feltöltése és kezelése.
  + A React StorageManager komponens használata.



## 4. Middleware használata

* Adminisztrációs oldalak hozzáférésének korlátozása.

# Frontend: Next.js

# A frontend felelős a felhasználói interakciók és az alkalmazás dinamikus megjelenítéséért. A Next.js keretrendszer a React-ökorendszer erejét használja a szerveroldali rendereléshez és a kliensoldali funkcionalitáshoz.

# Stílus: Tailwind CSS biztosítja az egységes és reszponzív megjelenést.

# Kommunikáció: A backendgel való kapcsolat GraphQL API-n keresztül valósul meg.

# Frontend Dokumentáció

## Bevezetés

Az oldalon lehetőség nyílik a medical tool létrehozására, szerkesztésére, megtekintésére és törlésére. Személyes adatok megadásával létre lehet hozni egy profilt mely az előbbiekben említetteket lehet elvégezni, ha Admin jogot kap a felhasználó.

Az alábbiakban részletezve van, hogy a különböző komponensek miként használom a frontenden.

## Fontosabb fájlok részletesebb leírással:

## auth/resource.ts

* **Funkció:** Az Amazon Cognito-alapú hitelesítést definiálja és konfigurálja az AWS Amplify használatával.
* **Főbb jellemzők**:
  + E-mail-alapú bejelentkezést támogat.
  + Csoportos jogosultságkezelést tesz lehetővé az "Admins" csoport létrehozásával.
* **Függvények és metódusok**:
  + defineAuth: Az auth konfiguráció definiálására szolgál.
    - Paraméterek:
      * loginWith: Bejelentkezési módok meghatározása (itt: e-mail).
      * groups: Felhasználói csoportok listája (itt: "Admins").
    - Funkció: Létrehozza és konfigurálja az auth erőforrást az AWS Amplify számára.

## storage/resource.ts

* **Funkció:** Az Amazon S3-tárolók konfigurációját határozza meg az AWS Amplify környezetében.
* **Főbb jellemzők:**
  + Tároló neve: **onyxStoreNextGen2Bucket**.
  + Hozzáférés-szabályozás a különböző felhasználói csoportokhoz:
    - Vendégek (guest): Csak olvasási jogosultság.
    - Hitelesített felhasználók (authenticated): Csak olvasási jogosultság.
    - Adminisztrátorok (Admins): Olvasási, írási és törlési jogosultság.
* **Függvények és metódusok:**
  + defineStorage: A tárolási erőforrás konfigurálására szolgál.
    - **Paraméterek:**
      * name: A tároló neve az Amazon S3-ban.
      * access: A hozzáférési szabályok meghatározása egy objektumstruktúrán keresztül.
        + **Almappák és szabályok:**

"product-images/\*": Az elérési út, ahol a tárolt fájlokhoz jogosultságokat rendelnek.

allow.guest.to(["read"]): Vendégek számára olvasási hozzáférés biztosítása.

allow.authenticated.to(["read"]): Hitelesített felhasználók számára olvasási hozzáférés biztosítása.

allow.groups(["Admins"]).to(["read", "write", "delete"]): Adminisztrátorok számára teljes hozzáférés engedélyezése (olvasás, írás, törlés)

## data/resource.ts

* **Funkció:** Az Amazon DynamoDB adatmodelljeinek és hozzáférési szabályainak definiálása az AWS Amplify használatával.
* **Főbb jellemzők:**
  + Adatmodell létrehozása a Product entitás számára.
  + Felhasználói hozzáférések meghatározása:
    - Vendégek (guest): Csak olvasási jogosultság.
    - Hitelesített felhasználók (authenticated): Csak olvasási jogosultság.
    - Adminisztrátorok (Admins): Teljes hozzáférés.
* **Függvények és metódusok:**
  + a.schema(): Az adatmodell sémájának meghatározására szolgál.
    - **Product modell:**
      * **Attribútumok:**
        + name: Kötelező szövegmező a termék nevéhez.
        + description: Kötelező szövegmező a termék leírásához.
        + price: Kötelező egész szám a termék árához.
        + image: Nem kötelező szövegmező a termékkép URL-jéhez.
      * **Hozzáférési szabályok:**
        + allow.guest().to(["read"]): Vendégek számára olvasási jogosultság.
        + allow.authenticated().to(["read"]): Hitelesített felhasználók számára olvasási jogosultság.
        + allow.group("Admins"): Adminisztrátorok számára teljes hozzáférés.
  + defineData(): Az adatforrás definiálása.
    - **Paraméterek:**
      * schema: A fent definiált adatmodell.
      * authorizationModes: Az alapértelmezett engedélyezési módot "IAM"-ként határozza meg.

## backend.ts

* **Funkció:** Az AWS Amplify backend erőforrásainak összefogása és inicializálása.
* **Főbb jellemzők:**
  + Importálja a hitelesítést (auth), adatkezelést (data), és tárolást (storage).
  + Ezeket az erőforrásokat a defineBackend függvényben regisztrálja.
* **Metódusok:**
  + defineBackend: Az Amplify backend inicializálására szolgál, az auth, data, és storage erőforrásokat összefűzve.

Egyszerűen összefoglalja és egyesíti a korábban definiált erőforrásokat az alkalmazás backendjének konfigurációjához.

## További fájlok:

## middleware.ts

* **Funkció**: Ellenőrzi, hogy a felhasználó admin jogosultsággal rendelkezik-e az /admin útvonalak eléréséhez.
* **Működés**:
  + Az admin jogosultságot a checkIsAdmin függvény ellenőrzi.
  + Ha a felhasználó admin, továbblép.
  + Ha nem, átirányítja a bejelentkezési oldalra (/signin).
* Útvonal-konfiguráció: Csak az /admin és annak alútvonalaira vonatkozik.

## utils/amplify-utils.ts

* **Funkció:** AWS Amplify-alapú hitelesítési és szerveroldali adatkezelési segédfüggvényeket biztosít.
* **Metódusok:**
  + runWithAmplifyServerContext: Amplify szerveroldali kontextusban futtatja a műveleteket.
  + checkIsAuthenticated: Ellenőrzi, hogy a felhasználó be van-e jelentkezve.
  + checkIsAdmin: Ellenőrzi, hogy a felhasználó admin jogosultsággal rendelkezik-e.
  + cookieBasedClient: Szerveroldali kliens, amely cookie-kat használ az adatok eléréséhez az IAM hitelesítési móddal.

## context/AdminContext.tsx

* **Funkció:** Adminisztrátor jogosultságkezelést biztosít React kontextus használatával.
* **Működés:**
  + Ellenőrzi, hogy a felhasználó be van-e jelentkezve és admin jogosultsággal rendelkezik-e.
  + Hitelesítési események (signedIn, signedOut) figyelése az AWS Amplify Hub segítségével.
  + Admin státusz ellenőrzése a Cognito Admins csoport alapján.
* **Metódusok:**
  + AdminContextProvider: Kontextust biztosít az admin státusz eléréséhez.
    - Állapotkezelést (isSignedIn, isAdmin) végez.
    - Automatikusan frissíti a jogosultságokat hitelesítési események során.
  + useAdminContext: Hozzáférést ad az admin státuszhoz a komponenseken belül.

## components/AuthenticatorClient.tsx

* **Funkció:** AWS Amplify hitelesítési felület biztosítása.
* **Működés:** Az Authenticator komponenssel egyszerű hitelesítési folyamatot tesz elérhetővé, például bejelentkezést vagy regisztrációt.
* **Metódusok:**
  + **Nincs egyedi metódus:** A komponens egyszerű megjelenítésre fókuszál.

## components/ClientSetupContextProvider.tsx

* **Funkció:** AWS Amplify konfigurációs környezet biztosítása React alkalmazáshoz.
* **Működés:**
  + Amplify beállításainak inicializálása (Amplify.configure).
  + Hitelesítési környezet létrehozása az Authenticator.Provider segítségével.
  + Adminisztrátori jogosultságok kezelése az AdminContextProvider-rel.
* **Metódusok:**
  + Amplify.configure(outputs, { ssr: true })**:** Amplify inicializálása.
  + Authenticator.Provider**:** Globális hitelesítési szolgáltatást nyújt.
  + AdminContextProvider**:** Adminisztrátori kontextus biztosítása.

## components/Image.tsx

* **Funkció:** S3-ban tárolt képek megjelenítése.
* **Működés:**
  + Az ImageComponent segítségével az Amazon S3 tárhelyen található képeket jeleníti meg a StorageImage komponenssel.
* **Metódusok:**
  + <StorageImage>**:** A kép eléréséhez a path és a altText paramétereket használja.

## components/NavBar.tsx

* **Funkció:** Navigációs sáv biztosítása.
* **Működés:**
  + Dinamikusan változó linkek jelennek meg a felhasználó admin státusza alapján.
  + Kezeli a kijelentkezési és bejelentkezési folyamatokat.
* **Metódusok:**
  + useAuthenticator**:** Ellenőrzi, hogy a felhasználó hitelesítve van-e.
  + signOutSignIn**:** Kijelentkezés vagy bejelentkezés oldali átirányítás.
  + routes.filter**:** Linkek szűrése admin státusz szerint.

## components/ProductCreate.tsx

* **Funkció:** Új termék létrehozása.
* **Működés:**
  + Egy űrlapon keresztül kezeli a termék létrehozását.
  + Hiba vagy siker esetén állapotot frissít és értesítéseket jelenít meg.
* **Metódusok:**
  + handleSuccess**:** Állapotfrissítés sikeres műveletnél.
  + handleError**:** Hibakezelés és hibaüzenet megjelenítése.
  + handleSubmit**:** Az űrlap beküldési folyamatának kezelése.

## components/ProductEdit.tsx

* **Funkció:** Termékek szerkesztése.
* **Működés:**
  + Egy szerkesztő űrlapon keresztül frissíti a termék adatait.
  + Hiba vagy siker esetén értesítéseket jelenít meg.
* **Metódusok:**
  + handleSuccess**:** Állapotfrissítés sikeres frissítés esetén.
  + handleError**:** Hibakezelés és hibaüzenet megjelenítése.
  + handleSubmit**:** A szerkesztési művelet kezelése.

## components/ProductItem.tsx

* **Funkció:** Termék információinak megjelenítése kártya formájában.
* **Működés:**
  + A termék nevét, leírását és képét jeleníti meg.
  + Adminok számára szerkesztési és törlési lehetőségeket biztosít.
* **Metódusok:**
  + ProductItemControls**:** Admin vezérlőelemek (szerkesztés, törlés).
  + ImageComponent**:** Termékkép betöltése.

## components/ProductItemControls.tsx

* **Funkció:** Admin vezérlők biztosítása a termékekhez (szerkesztés, törlés).
* **Működés:**
  + Törlés esetén az adatbázisból eltávolítja a terméket.
  + Átnavigál a termék szerkesztési oldalára.
* **Metódusok:**
  + handleEdit**:** Szerkesztési oldalra navigál.
  + handleDelete**:** Terméktörlés és gyorsítótár ürítése.

## app/layout.tsx

* **Funkció:** Az alkalmazás alapvető elrendezésének meghatározása.
* **Működés:**
  + Beállítja a HTML és test (body) alapvető elrendezését.
  + Tartalmazza a NavBar navigációs komponenst és a ClientSetupContextProvider-t, amely az alkalmazás állapotát biztosítja.
* **Metódusok:**
  + RootLayout: Az alkalmazás alapelrendezését hozza létre.
    - Globális stílusokat alkalmaz (globals.css).
    - Betölti a navigációs sávot és a gyermekkomponenseket.

## app/page.tsx

* **Funkció:** A kezdőoldal tartalmának és termékeinek megjelenítése.
* **Működés:**
  + Ellenőrzi, hogy a felhasználó be van-e jelentkezve (checkIsAuthenticated).
  + Listázza az adatbázisból lekért termékeket a cookieBasedClient segítségével.
  + Termékkártyákat jelenít meg a ProductItem komponenssel.
* **Metódusok:**
  + Home:
    - Ellenőrzi a hitelesítési állapotot.
    - Lekéri a termékeket az Amazon DynamoDB-ből.
    - Reszponzív elrendezésben jeleníti meg a termékeket.

## app/signin/page.tsx

* **Funkció:** Bejelentkezési oldal megjelenítése.
* **Működés:**
  + Ellenőrzi a felhasználó hitelesítési állapotát (checkIsAuthenticated).
  + Ha be van jelentkezve, egy üzenetet jelenít meg.
  + Ha nincs bejelentkezve, a AuthClient komponenst jeleníti meg a hitelesítéshez.
* **Metódusok:**
  + SignIn:
    - Lekéri a felhasználó bejelentkezési állapotát.
    - Feltételesen megjeleníti a bejelentkezett állapothoz tartozó üzenetet vagy a hitelesítési komponenst.

## app/admin/product-edit/[id]/page.tsx

* **Funkció:** Termékszerkesztési oldal megjelenítése.
* **Működés:**
  + A kapott id paraméter alapján betölti a termékszerkesztő komponenst (ProductEdit).
  + Azonosító (params.id) alapján egyedi terméket szerkeszt.
* **Metódusok:**
  + ProductEditPage:
    - Paraméterként kapott id alapján megjeleníti a ProductEdit komponenst.

## app/admin/product-create/page.tsx

* **Funkció:** Terméklétrehozási oldal megjelenítése.
* **Működés:**
  + A ProductCreate komponenst jeleníti meg, amely felelős egy új termék létrehozásáért.
* **Metódusok:**
  + ProductCreatePage: Egyszerűen betölti és megjeleníti a ProductCreate komponenst.

## actions/revalidate.ts

* **Funkció:** Gyorsítótár frissítése a Next.js alkalmazásban.
* **Működés:**
  + A megadott útvonal (vagy alapértelmezés szerint a gyökér /) gyorsítótárát újragenerálja a revalidatePath metódus segítségével.
  + Hiba esetén naplózza az eseményt.
* **Metódusok:**
  + clearCachesByServerAction:
    - **Paraméter:** path (opcionális) – Az újragenerálni kívánt útvonal.
    - Ha nincs megadva, a gyökérútvonal (/) gyorsítótárát frissíti.
    - Hibakezelést biztosít a try-catch blokkban.

## components-ui generálása:

**npx amplify generate forms** parancs kifejezetten az AWS Amplify projektekben űrlapok generálására szolgál. Ez lehetővé teszi az adatmodellekből automatikusan létrehozott, használatra kész formok létrehozását.

### Funkció

Automatizált űrlapkomponenseket hoz létre, amelyek az AWS Amplify adatmodelljeihez kapcsolódnak.

### Működés

1. **Adatmodellek beolvasása:**
   * Az AWS Amplify backendben definiált modelleket (amplify/backend/models) használja.
   * Ezek az adatmodellek határozzák meg, hogy milyen mezőket tartalmazzon az űrlap.
2. **Űrlapok generálása:**
   * A parancs a modellek alapján automatikusan létrehozza a formokat.
   * Példa generált komponensek:
     + **Létrehozási űrlap (Create Form):** Új adatok felvitelére.
     + **Frissítési űrlap (Update Form):** Létező adatok módosítására.
     + **Törlési űrlapok integrálása:** Törlési funkciók implementációjához.
3. **Integráció a generált komponensekkel:**
   * A generált űrlapok már tartalmazzák az AWS AppSync vagy a DynamoDB adatkapcsolatot.
   * Támogatják az Amplify DataStore-t vagy az AppSync GraphQL API-t.
4. **Elhelyezés:**
   * A generált fájlokat a src/ui-components mappába helyezi.
   * Példa generált fájlok:
     + YourModelCreateForm.jsx
     + YourModelUpdateForm.jsx
5. **Stílusok és UX:**
   * Az AWS Amplify UI-stílusokat használja, reszponzív és használatra kész megjelenést biztosít.
   * A formok alapértelmezett validációval rendelkeznek (pl. kötelező mezők).

# Backend: Amplify Gen2

## Technikai Részletek

* Backend Platform: AWS Amplify integráció az alábbi főbb szolgáltatásokkal:
  + Cognito: Felhasználói autentikáció és jogosultságkezelés.
  + AppSync: GraphQL API adatlekérésekhez és frissítésekhez.
  + Lambda Functions: Szerver nélküli számítási logika implementálásához.
  + DynamoDB: Strukturált adatok tárolása.
  + S3: Termékekhez kapcsolódó képek és fájlok tárolása.

## Főbb Backend Funkciók:

* User Management: Felhasználói regisztráció, bejelentkezés, jogkörök kezelése.
* Product Management: Termékek hozzáadása, módosítása és törlése.

## Adatmodell

### DynamoDB Táblák

* Users Table: Felhasználók adatai, beleértve a jogosultsági szinteket.
* Products Table: Termékek részletes információi, például nevek, leírások, árak.

