

Projekt dokumentáció



Szak: Számítástechnika III.

Készítették:

Mihácsa Ervin-László
Mihácsa Nándor
Vernes Dávid-László

1.Bevezető.....	3
2.Projekt célja.....	3
3.Követelmény specifikáció.....	4
3.1.Felhasználói követelmények.....	4
3.2. Rendszerkövetelmények.....	7
3.2.1. Funkcionális követelmények:.....	8
3.2.2. Nem funkcionális követelmények:.....	9
4.Tervezés.....	10
4.1.Architektúra.....	10
4.2. Szekvencia diagramok.....	11
4.2.1. Bejelentkezés.....	11
4.2.1. Termék feltöltés.....	12
4.3.Adatbázis.....	13
4.4Wireframe.....	15
5.Projekt management.....	17
5.1.Github.....	17
5.1.Kanban.....	18
6.Alkalmazás működése.....	18
6.1 Front End.....	18
6.2 Backend.....	19
6.2 Adatbázis.....	19
7.Összegzés.....	22
8.1.Hibák javítása.....	22
8.2.Továbbfejlesztési lehetőségek.....	22
9.Bibliográfia.....	23

1.Bevezető

Napjainkban rengeteg terméket vásárolnak az emberek online, egyre több üzlet és márka van jelen az online piacon. A mai világban az idő körül forog, az embereknek szükségük van egy gyors és hatékony módszerrel termékeik eladására és a vásárlásra.

Az MGO oldalán mindenki árulhatja termékeit, akár használtan vagy újan, kérdéseket tehet fel róla és vásárlás előtt ellenőrizheti ha megfelel a kívánt célra.

Fontos, hogy ne csak az adott üzlet vagy márka termékeit találjuk meg, hanem minél szélesebb körben lehessen válogatni.

2.Projekt célja

A projekt célja egy innovatív, felhasználóbarát weboldal és mobil alkalmazás létrehozása, amely a modern e-kereskedelem új dimenzióját kínálja. Ez a platform segít az embereknek abban, hogy könnyen megtalálják és megvásárolják az érdekeiknek megfelelő termékeket, legyen szó elektronikáról, ruházatról, háztartási eszközökről vagy bármilyen más termékről.

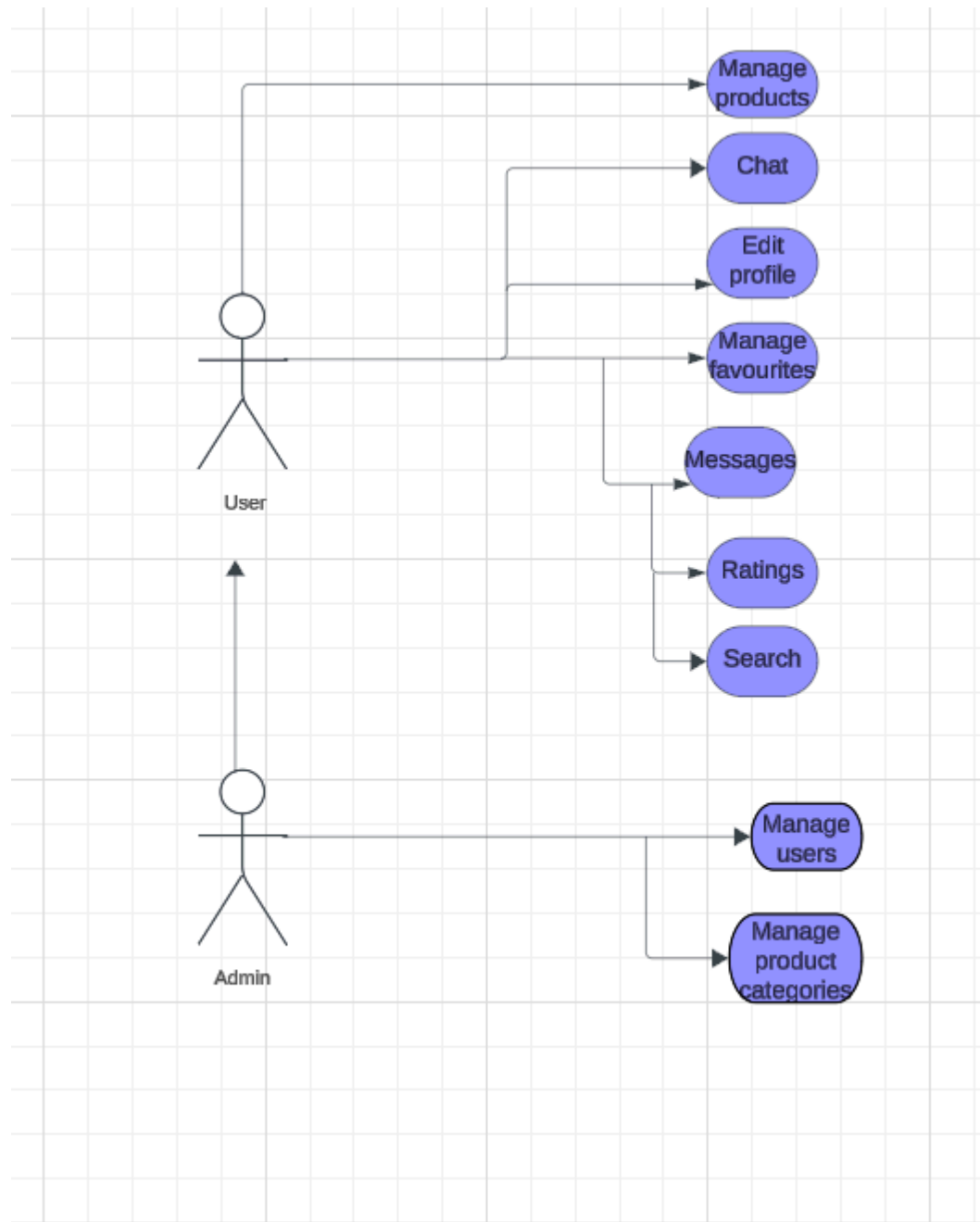
A rendszerlehetővé teszi a felhasználók számára, hogy pontosan azokat a termékeket találják meg, amelyek a leginkább megfelelnek az igényeiknek és preferenciáiknak. Emellett a platform támogatja a felhasználókat abban is, hogy megbizonyosodjanak a termékek minőségéről és megfelelőségéről, részletes leírásokkal, vásárlói véleményekkel és értékelésekkel.

A platformon nemcsak vásárolni, hanem eladni is lehet, lehetőséget biztosítva mind az új, mind a használt termékek piacára. Az eladók könnyedén hirdethetik termékeiket, megadhatják azok részletes leírását, árát és feltételeit.

A vásárlók és eladók közötti közvetlen kommunikáció kiemelt szerepet kap, lehetőséget adva a részletek egyeztetésére, kérdések megválaszolására. A platformon keresztül szervezhetők személyes találkozók is, amelyek lehetővé teszik, hogy a vásárlók személyesen ellenőrizzék a terméket, így biztosítva a teljes elégedettséget.

3.Követelmény specifikáció

3.1.Felhasználói követelmények



Ábra 1: Use case diagram

Registration - Az weboldal és applikáció csak akkor elérhető ha egy felhasználó rendelkezik már egy fiókkal, ha nem akkor a **Sign up** gombra kattintva létrehozhat

egy saját fiókot aminek tartalmaznia kell egy email címet, egy felhasználó nevet, egy jelszavat, telefonszámot és lakcímet.

Login - Az weboldalt és applikációt csak regisztrált felhasználók használhatják, a Login oldalon, megadva az email címet és jelszót beléphetnek és igénybe vehetik a szolgáltatásokat.

Home Page - Ezen az oldalon megjelennek a kategóriák, amelyekhez hozzá vannak rendelve termékek. Rákattintva egy kategóriára megjelennek azok a termékek amelyek a kategóriához tartoznak.

Profile Page - A fejlécben lévő profil ikonra kattintva megjelenik egy legördülő menü ami tartalmaz egy **My Profile** fület. Erre kattintva megjelennek a felhasználó személyes adatai és az általa árult termékek.

Edit Profile Page - A fejlécben lévő profil ikonra kattintva megjelenik egy legördülő menü ami tartalmaz egy **Edit Profile** fület, erre kattintva a felhasználó módosíthatja személyes adatait.

Admin Page - Ezt az oldalt is a legördülő menü tartalmazza, de csak az a felhasználó érheti el akinek van admin jogosultsága. Itt az adminisztrátor hozzáfér az összes felhasználóhoz és termékhez, törölheti ezeket ha panasz van a felhasználóról vagy ha úgy ítéli meg, hogy egy bizonyos termék nem felel meg az oldal elvárásainak.

Product Page by category - A főoldalon lévő kategóriára kattintva érhető el az oldal, kategóriától függően listázza a termékeket, a felhasználóknak rendezési lehetőséget biztosít.

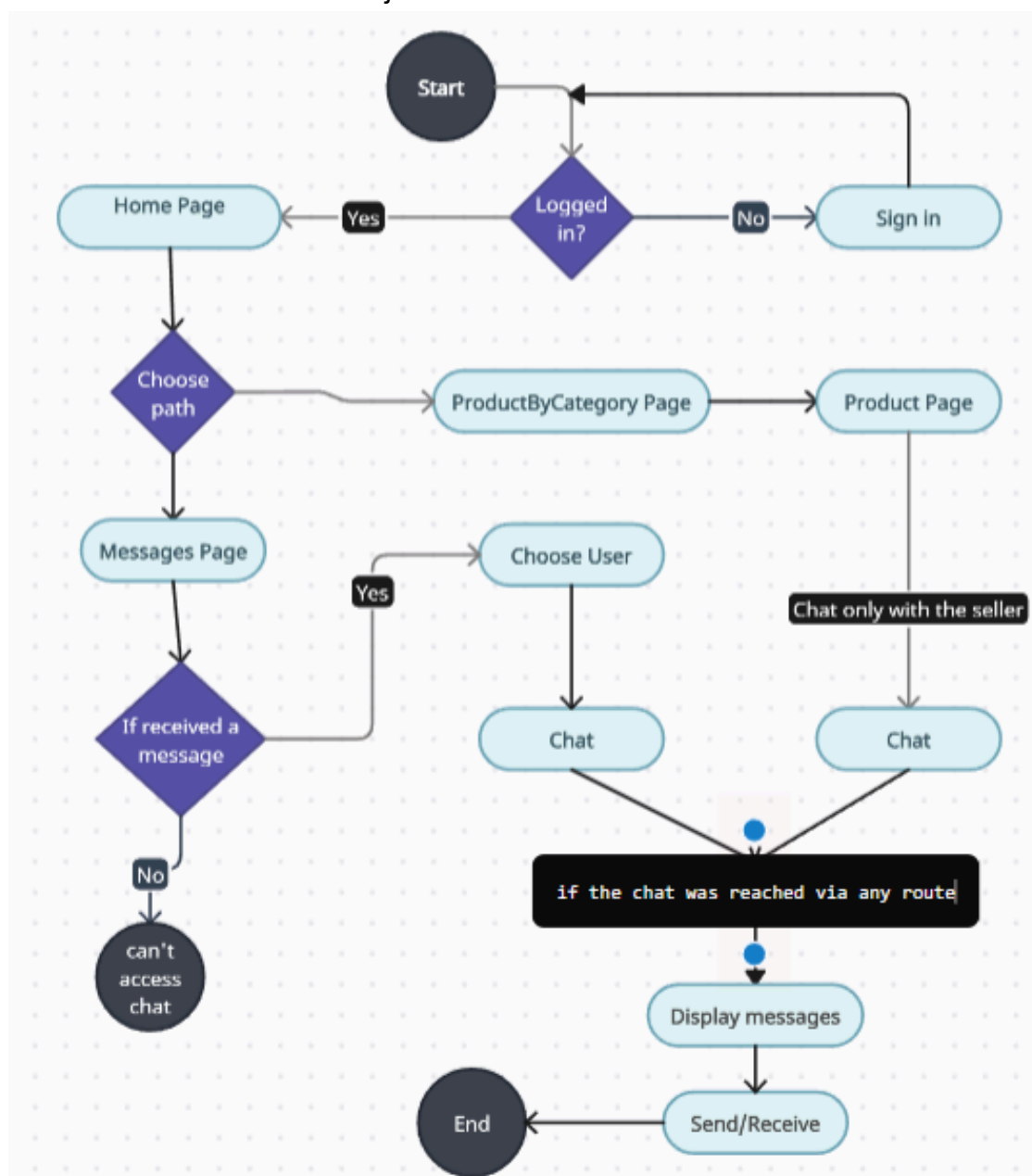
Product Detail Page - Részletes betekintést nyújt a felhasználónak a termékről, annak képeiről és leírásáról. Továbbá az eladó adatairól és értékeléséről. Ezen az oldalon a felhasználó beszélgetést kezdeményezhet az eladóval.

Favourites Page - A fejlécben található szív ikonra kattintva elérhető a kedvencek oldal ahol megjelennek azok a termékek amelyeket már korábban hozzáadtunk a kedvencekhez.

Messages Page - A fejlécben található üzenet ikonra kattintva válik elérhetővé az oldal, itt minden felhasználótól kapott utolsó üzenet jelenik meg, azokra kattintva elérhető a teljes beszélgetés.

Upload Page - Az oldal jobb felső sarkában lévő **New advertising** gomb segítségével érjük el az oldalt, itt a felhasználó képeket, leírást, árat és kategóriát adhat meg a eladásra szánt terméknek.

Chat - Első sorban a belépéshez a felhasználó be kell jelentkezzen, ha nincs fiókja, akkor létre kell hoznia egyet, ezt követően a felhasználó eléri a kezdőlapot. A kezdőlapról két különböző helyen is elérheti a chatet, először a fejlécben lévő üzenetek ikonra kattintva megjelennek az utolsó üzenetek azoktól a felhasználóktól akik üzentek neki. Ha rákattint egy üzenetre megjelenik a chat. A második elérési mód, ha a kezdőképernyőről kiválasztunk egy kategóriát, majd megjelenik az összes termék abból a kategóriából. Itt kiválasztva egy terméket megjeleníti az összes adatot a termékről és az eladóról. Ezen az oldalon a küldés gombra kattintva megjelenik a chat a felhasználó és az eladó között. Miután mindkét esetben megjelent a chat, az tartalmazni fogja az összes üzenetet a két felhasználó között. A chat ablak aljában lehet írni majd elküldeni egy üzenetet. Az alábbi ábra a Chat elérését és működését mutatja be:



Ábra 2: Chat Activity diagram

3.2. Rendszerkövetelmények

3.2.1. Funkcionális követelmények:

User:

- **Regisztráció:** A web és az applikáció is csak fiókkal elérhető, regisztráció során a felhasználó meg kell adja felhasználó nevét(mindenkinek egyedi), email címét(lehetőleg érvényest, mivel ha elfelejtette a jelszavát akkor az erre kapott üzenet segítségével állítható új), jelszó(legalább 6 karakter),lakcímét és telefonszámát.
- **Bejelentkezés:** A regisztráció során létrehozott fiók email címét és jelszavát kell megadja a felhasználó.
- **Navigálás az oldalak között.**
- **Termék hozzáadás:** A felhasználóknak lehetőségük van saját termékek eladására, megadva a termék leírását, képeit, árát, nevét és kategóriáját.
- **Kedvenc termékek:** A felhasználóknak lehetőségük van termékeket menteni a kedvencek közé, majd onnan törölni is a termék képe mellett található ikon segítségével.
- **Profil szerkesztés:** A felhasználóknak lehetőségük van profiljukat módosítani az **Edit Profil** fülben a fejlécben, termékeiket eltávolítani a **My Profile** fülben a saját adataik alatt a **Delete** gombra kattintva
- **Chat:** Biztonságos, integrált csevegési rendszer, amely lehetővé teszi a vásárlók és eladók közötti közvetlen kommunikációt a termékek részleteinek egyeztetésére és a tranzakciók elősegítésére.

Az admin oldalt a fejlécben lévő dropdown menüből lehet elérni, csak azoknak a felhasználóknak akik rendelkeznek admin jogosultsággal.

Admin:

- **Felhasználók menedzsmentje:** Az admin oldalon az adminisztrátor keresheti majd listázhatja a felhasználókat, a felhasználó adatai mellett lévő **Delete** gombbal törölhetit a felhasználókat.

- **Termékek menedzsmentje:** Az admin oldalon az adminisztrátor keresheti majd listázhatja a felhasználók termékeit, majd a termék adatai mellett lévő **Delete** gombbal törölheti a termékeket, ha azok közül valamelyik nem felel meg az oldal elvárásainak.

3.2.2. Nem funkcionális követelmények:

Web:

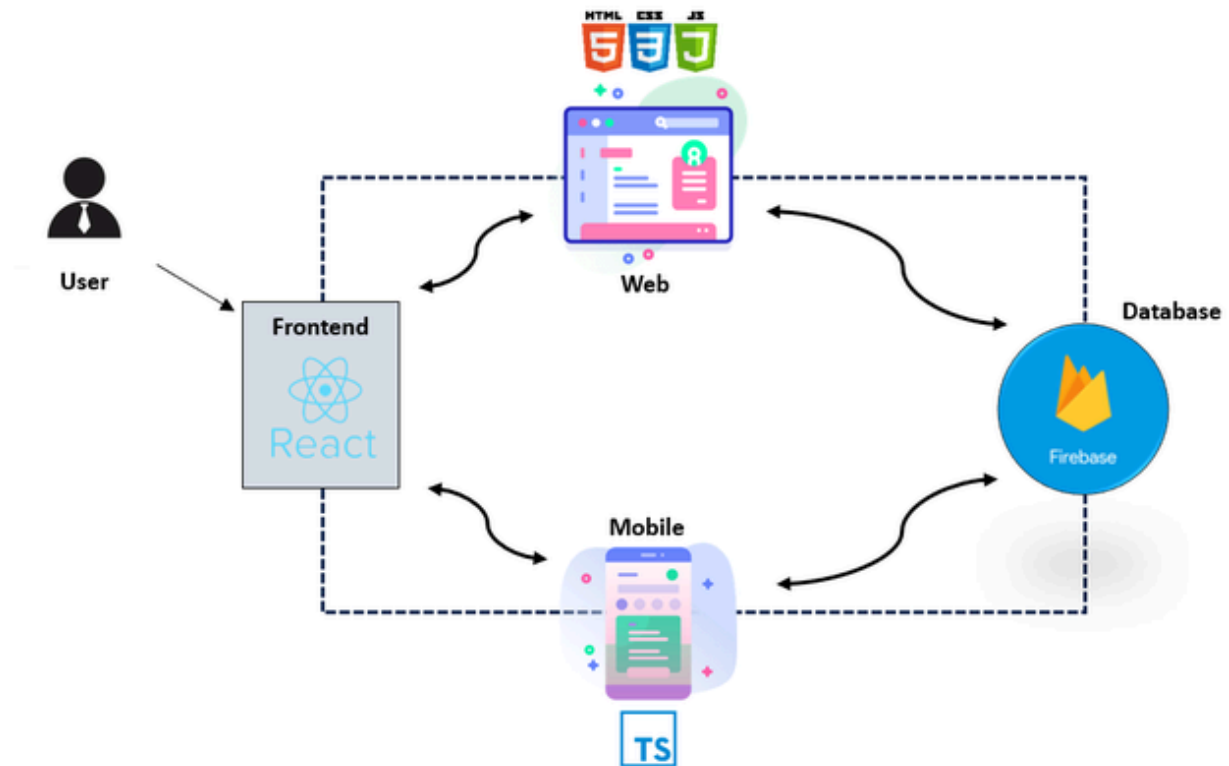
- Laptop vagy asztali gép.
- Operációs rendszer: Linux, Windows, Mac.
- Internet kapcsolat.
- Böngésző: Chrome, Opera, Brave, Mozilla, stb.
- Adatok védelme(GDPR).

Mobile:

- Mobil készülék.
- Android (5+).
- Tárhely, minimum 200 MB.
- Internet kapcsolat.
- Adatok védelme(GDPR).

4.Tervezés

4.1.Architektúra



Ábra 3: Architektúra

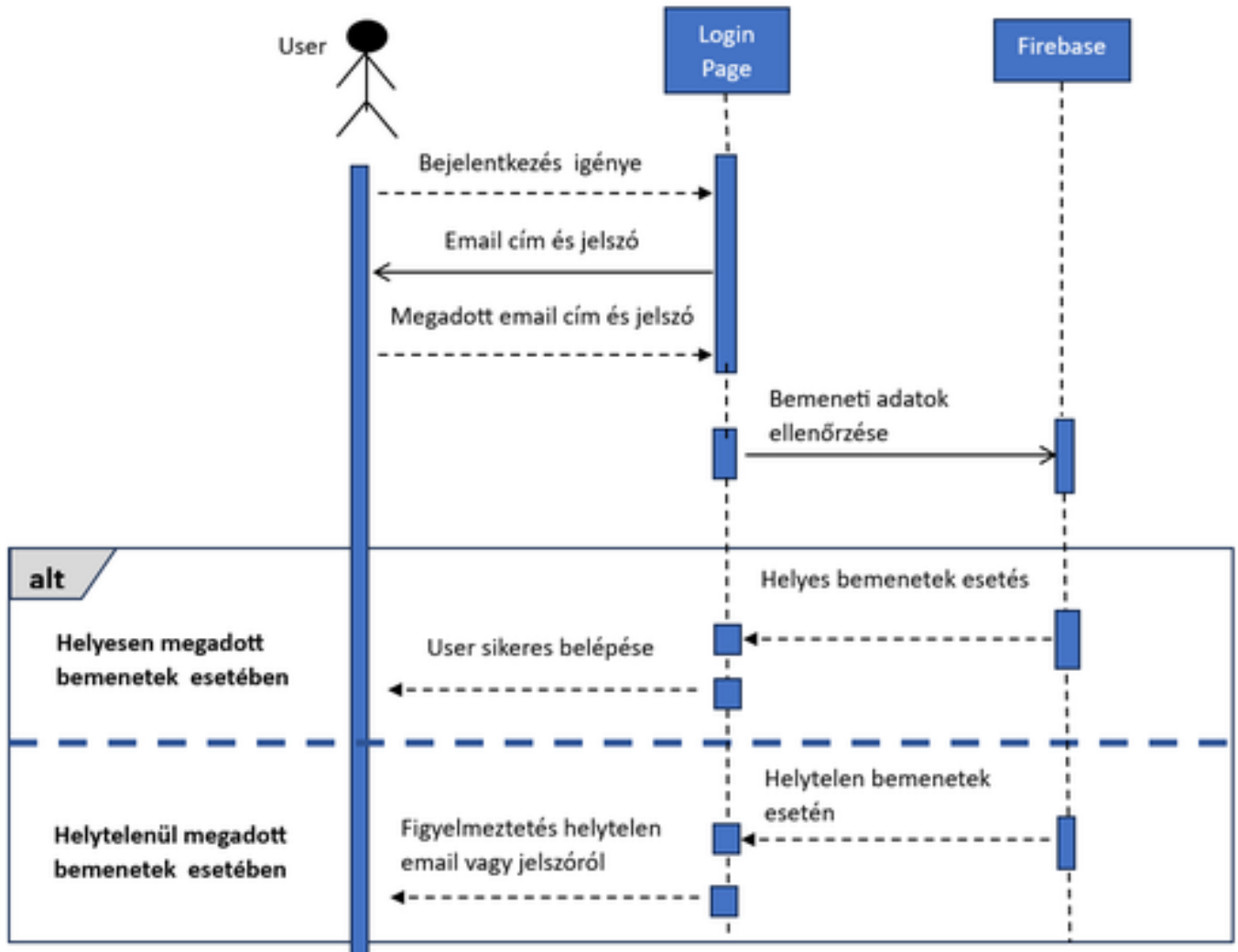
A bemutatott ábrán a projektünk architektúráját látjuk. A felhasználók közvetlenül a front-enddel érintkeznek, amely a képen látható módon React segítségével lett implementálva asztali számítógépeken, míg mobil eszközökön React Native keretrendszerben valósítottuk meg. A back-end részhez Node.js technológiát alkalmaztunk.

Mielőtt nekilátunk volna a projektnek, fontos döntéseket kellett hoznunk a használandó technológiákat illetően. A React és a React Native mellett döntöttünk, mert lehetővé teszik a gyors, hatékony fejlesztést és az egyszerű kód újra felhasználást a webes és mobil alkalmazások között.

Az adatbázisként Firebase-t használtunk, mert gyors és egyszerű beállítást kínál a felhő alapú adatbázis kezeléshez, valamint azonnal elérhető autentikációt, hosztingot és sok más szolgáltatást biztosít.

4.2. Szekvencia diagramok

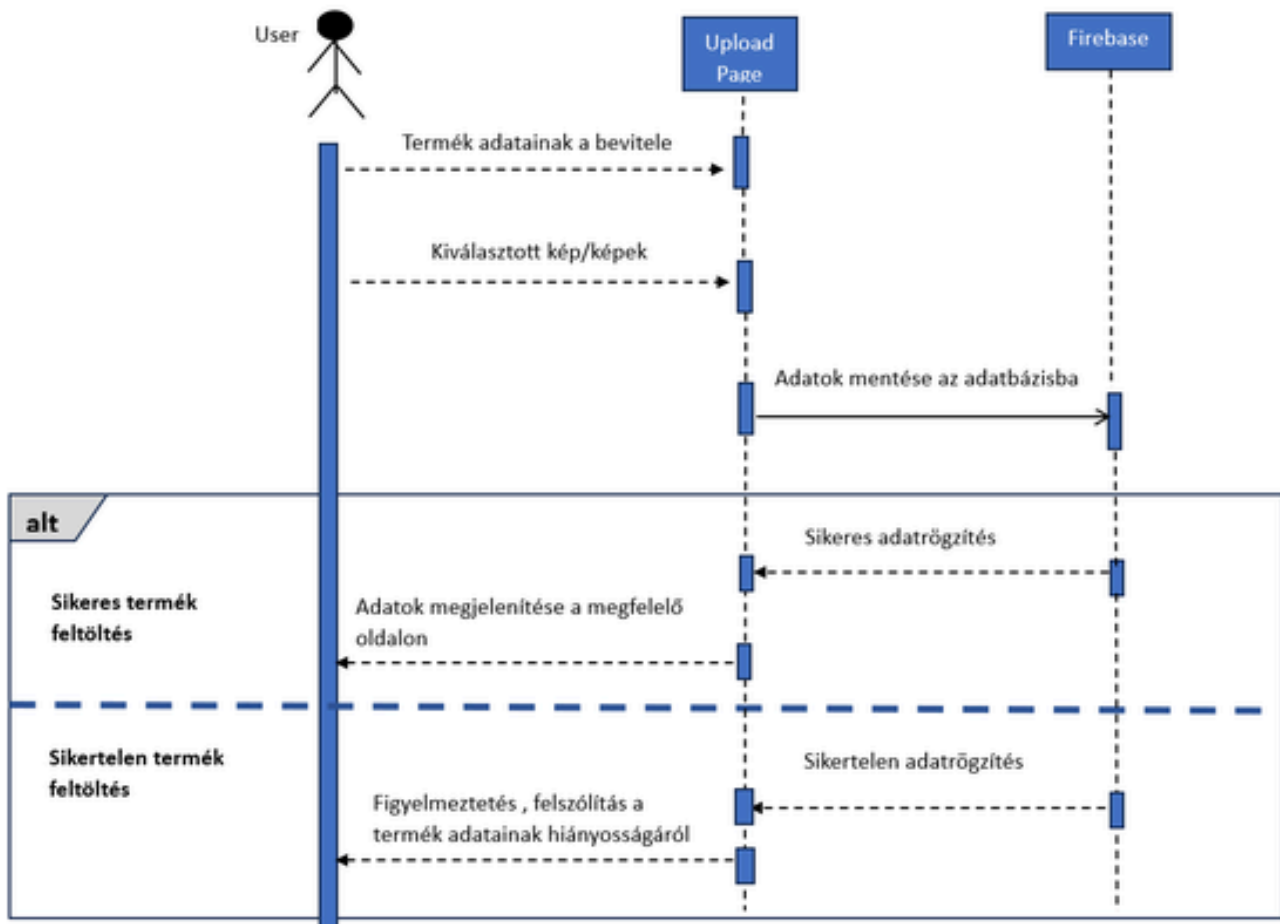
4.2.1. Bejelentkezés



Ábra 4: Bejelentkezés szekvencia diagram

A felhasználó a weboldalon legelőször a bejelentkezés oldallal találkozik. Itt ha el szeretné érni a weboldal további tartalmait, akkor be kell írja az email címét és a jelszavát, ezt követően rákattint a Login gombra. Az adatbázis ellenőrzi a bemeneti adatokat, hiteles bejelentkezés esetén továbbítja a felhasználót a kezdőképernyőre. Ha a bemenet helytelen vagy hiányos, akkor a felhasználó értesítés kap(Invalid email or password).

4.2.1. Termék feltöltés

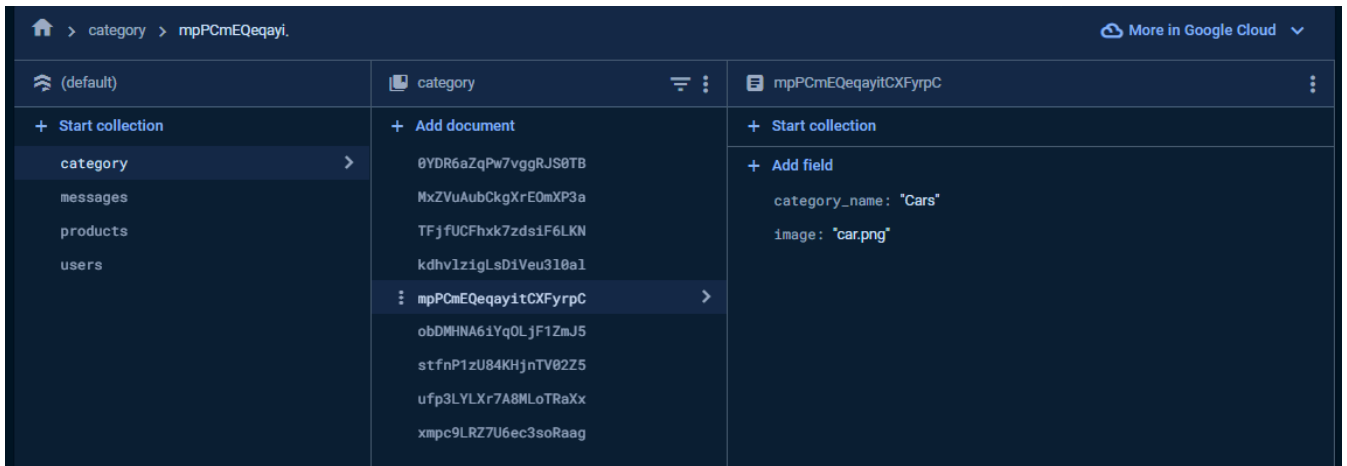


Ábra 5: Termék feltöltés szekvencia diagram

Ha a felhasználó új terméket szeretne feltölteni akkor a fejlécben lévő (New Advertising) gomb megnyomásával teheti meg. A gombot megnyomva belép egy oldalra ahol ki kell töltsse a termékéhez vonatkozó adatokat(termék név, ár, telefonszám, termékleírás). Ha ezeket kitöltötte akkor ki kell válassza a képeket a termékhez, majd kiválasztani milyen kategóriába tartozik a termék. Hiányos adatok vagy helytelen formátumú kép esetén a felhasználó értesítést kap.

4.3. Adatbázis

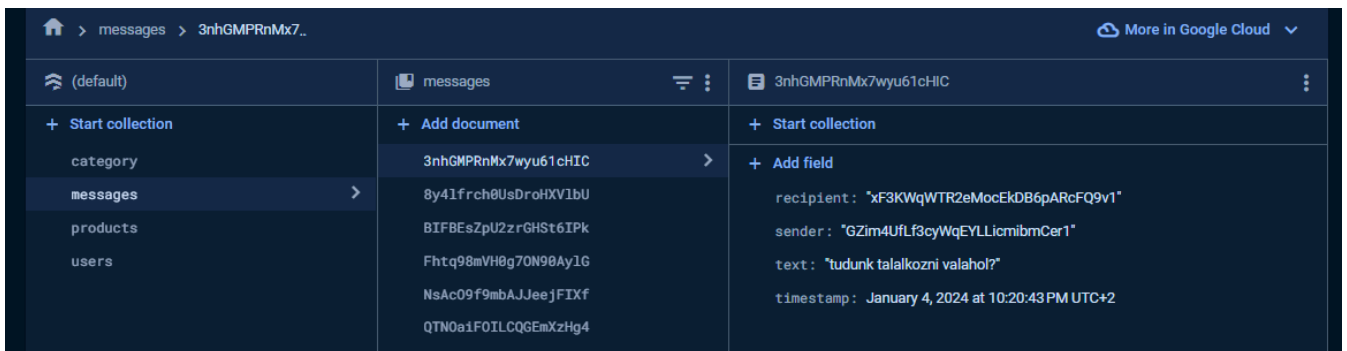
Kategóriák



Ábra 6: Category collection

A **Category** kollekcióban minden kategória el van látva egy egyedi azonosítóval, továbbá mezőket tartalmaz, mint a category_name és az image.

Üzenetek



Ábra 6: Messages collection

A **Messages** kollekció tartalmazza az összes elküldött üzenetet, egyedi azonosítóval ellátva minden üzenetet. Az üzenetek tartalmaznak egy küldő, fogadó, szöveg mezőket és egy időbélyeget.

Felhasználók

(default)	users	jmA281aSglaot4queXMoMTk3SeC2
+ Start collection	+ Add document	+ Start collection
category	1uLKMuK1S2f2Ta4eP6kU2jIP5tJ2	+ Add field
messages	GZ1m4UfLf3cyWqEYLLicm1bmCer1	address: "Ostasilor"
products	U07PYQ18hdhGoDg0YnHt9fd3owu2	averageRating: 4
users	jmA281aSglaot4queXMoMTk3SeC2	city: "Vasarhely"
	xF3KwqWTR2eMocEkDB6pARcFQ9v1	email: "mihacsanandor@gmail.com"
		phoneNumber: "0739291032"
		rating
		0 5
		1 5
		2 5
		3 1
		userName: "Nandika"

Ábra 7: Users collection

A Users kollekció a regisztrált felhasználókat tartalmazza egyedi azonosítóval ellátva. Regisztráció során a mezők közé bekerült az address, city, email, phone number, userName. Értékelés során, ha a felhasználók érték az eladott termékeket, akkor a mezők bővültek egy átlag értékeléssel és egy rating tömbbel ami tárolja az összes kapott értékelést.

Termékek

products > j0SsT6YFVQy3B.	products	j0SsT6YFVQy3Bm9Lw9cc
(default)	+ Add document	+ Start collection
+ Start collection	Rzf4KbXXhTSHHF1nCqcu	+ Add field
category	j0SsT6YFVQy3Bm9Lw9cc	category: "mpPCmEQeqayitXFyrcP"
messages	uyb3KJB8UGodv94kvXQa	description: "Recently brought from Netherlands - Engine 1.4 tdi - 80 hp - Air conditioning - functional - Electric windows -Original alloy wheels - Fog projectors -Board computer -Electric mirrors -Automatic pilot - Adjustable steering wheel"
products		images
users		0 "skoda2.webp"
		1 "skoda1.webp"
		price: "2 750 €"
		product_name: "Skoda Fabia 1.4 TDI"
		product_name_lowercase: "skoda fabia 1.4 tdi"
		user: "U07PYQ18hdhGoDg0YnHt9fd3owu2"

Ábra 7: Products collection

A Products kollekció tárolja a feltöltött termékeket, egyedi azonosítóval. Itt a termékek tartalmaznak egy kategória mezőt ami segítségével egy bizonyos

kategóriában lehet sorolni, egy leírást, árat, nevet és egy images tömböt ami tárolja a termékhez tartozó képeket. Minden termék hozzá van rendelve egy felhasználóhoz amely feltöltötte a terméket.

Képek tárolása

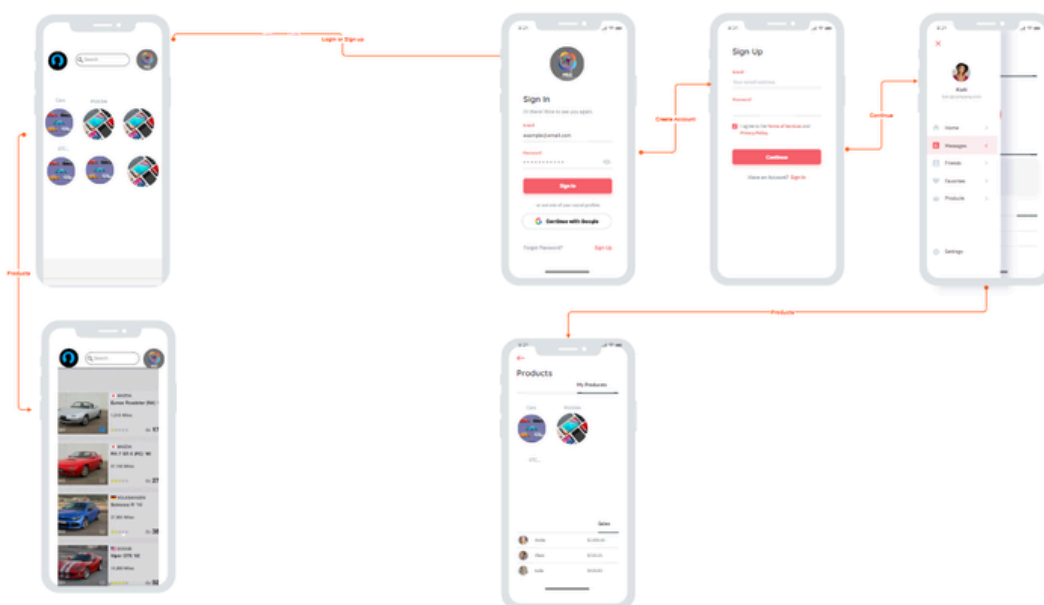
<input type="checkbox"/>	Name	Size	Type	Last modified
<input type="checkbox"/>	Rzf4KbXXhT5HHfInCqcu/	—	Folder	—
<input type="checkbox"/>	j0SsT6YFVQy3Bm9Lw9cc/	—	Folder	—
<input type="checkbox"/>	uyb3KJB8UGodv94kvXQa/	—	Folder	—

Ábra 8: Firebase Storage

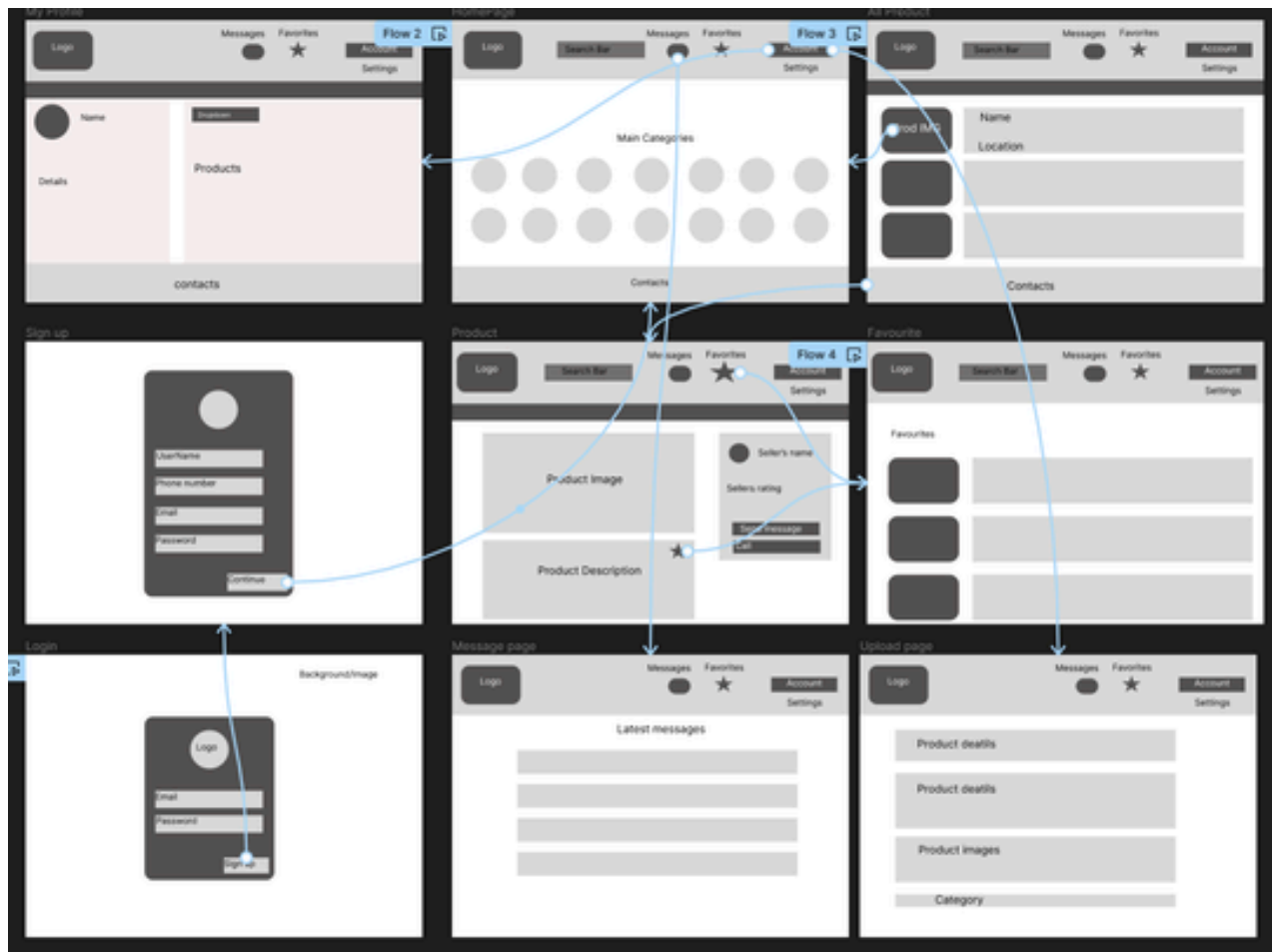
A képeket a firebasen belül a Storage-be tároltuk. A kategóriákhoz szükséges képeket egy **Category** folderben, a termékeket pedig a **Product** folderben amelyben minden terméknek van egy saját folderje amely a termék azonosítóról van elnevezve, minden termék a saját folderében tárolja a hozzá tartozó képeket.

4.4 Wireframe

Annak érdekében, hogy a munka gördülékenyen mehessen, szükség volt egy vázlatra amelyhez a csapat minden tagjának volt hozzáférése. A vázlatot webhez Figmában, mobilhoz pedig Moqupsban valósítottuk meg. A vázlat tartalmazza az oldalakat és azoknak elrendezést.



Ábra 9: Mobile wireframe




Ábra 9: Web wireframe

A projekt megvalósításakor volt amikor eltértünk attól, bővült az applikáció és a web is előre el nem tervezett oldalakkal, de támpontnak tökéletesen megfelelt.

5. Projekt management

5.1. Github



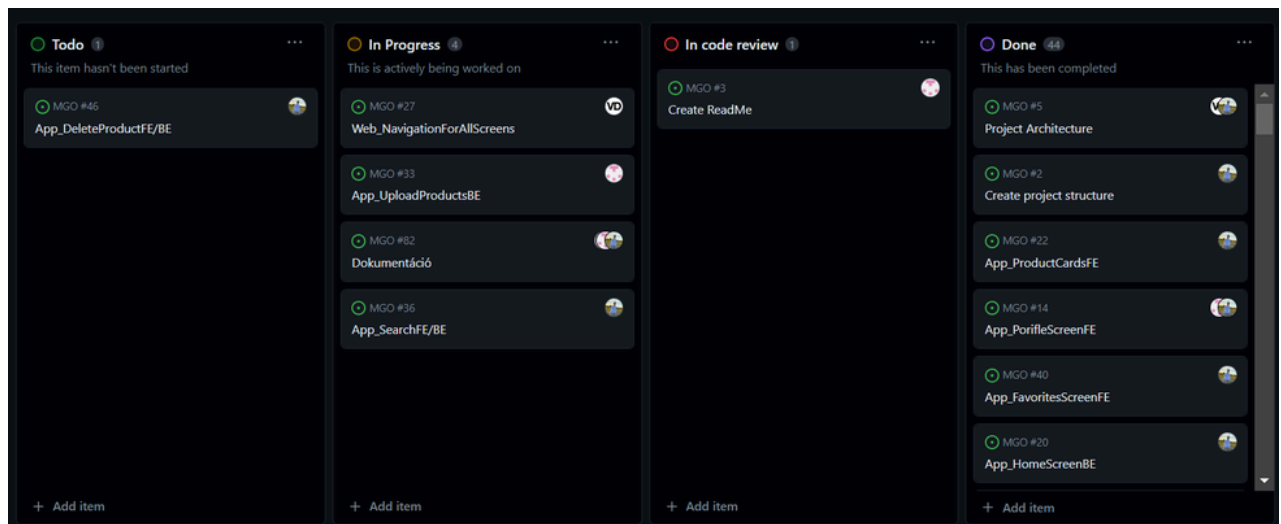
Home Screen bug fixing	Ervin28	12/29/2023 7:53 PM
App setup	MihacsaNandor	12/28/2023 9:05 PM
Merge pull request #76 from Ervin28/Web_AdminPage	VernesDavid-Laszlo*	12/28/2023 6:48 PM
Adjustments on the "My profile page" now shows the products which the users have. Added currency attribute to the		
Beta version of "Admin page " back-end and front-end. Needs some	MihacsaNandor	12/28/2023 3:15 PM
Merge pull request #75 from Ervin28/Web_SearchbarBE	VernesDavid-Laszlo*	12/25/2023 9:12 PM
Merge branch 'main' into Web_SearchbarBE	VernesDavid-Laszlo*	12/25/2023 9:12 PM
searchbar fourth commit, search completed	VernesDavid	12/25/2023 9:10 PM
Merge pull request #70 from Ervin28/main	Ervin28*	12/21/2023 3:52 PM
Update README.md	Ervin28*	12/21/2023 3:51 PM
Create README.md	Ervin28*	12/21/2023 3:48 PM
Beta version of "Product card" back-end a	MihacsaNandor	12/21/2023 3:32 AM
Database hot fix	Ervin28	12/20/2023 12:57 PM
searchbar third commit, search working, n	VernesDavid	12/16/2023 11:06 PM
Created the productcard containers and the home screen	Ervin28	12/15/2023 2:17 PM
Merge pull request #74 from Ervin28/App_SignUpScreenBE	Ervin28*	12/13/2023 2:55 PM
App_ProfileScreen initial commit	Ervin28	12/13/2023 2:12 PM
Signup screen backend finished	Ervin28	12/13/2023 2:11 PM
Merge pull request #73 from Ervin28/Web_SearchbarBE	Ervin28*	12/12/2023 2:12 PM
for merge 2.0	VernesDavid	12/12/2023 2:10 PM
Merge pull request #72 from Ervin28/Web_ProductByCategoryPageI	Ervin28*	12/12/2023 2:06 PM
Merge pull request #71 from Ervin28/main	Ervin28*	12/12/2023 2:03 PM
Merge branch 'WEB_UploadProductsFE' into main	Ervin28*	12/12/2023 2:02 PM
Merge pull request #69 from Ervin28	Ervin28*	12/12/2023 1:57 PM
favourite page backend second comi	VernesDavid	12/12/2023 1:55 PM
favourite page backend first commit, can add and remove proc	VernesDavid	12/11/2023 9:59 PM
Beta version of "upload page" back-end and front-end	MihacsaNandor	12/8/2023 1:10 PM
Beta version of "edit page" back-end	MihacsaNandor	12/8/2023 4:07 AM
Merge pull request #68 from Ervin28/App_SignUpScreenBE	Ervin28*	12/6/2023 3:24 PM
Web hot fix	Ervin28	12/6/2023 3:17 PM
Merge branch 'main' of https://github.com/Ervin28/MGO into App_S	Ervin28	12/6/2023 3:11 PM
Merge pull request #67 from Ervin28/Web_ProductByCategoryPageI	Ervin28*	12/6/2023 3:10 PM
Merge branch 'main' into Web_ProductByCategoryPageBE	Ervin28*	12/6/2023 3:10 PM

Verziókövetésre GitHub-ot használtunk, mely segítségével követni tudtuk egymás munkáját és új fejlesztéseket kipróbálni anélkül, hogy az a célunk rovására menne. Ezek mellett a felület biztonságot nyújt olyan esetben ha valakinek elveszik az információ a számítógépéről. Minden taskra külön branchet használtunk és ezeket a brancheket csatoltuk hozzá a "main" branch-hez. Itt látható a kód struktúrája:

5.1. Kanban

A Github Kanban boardot használtuk a feladatok menedzselésére. Itt mindenki kiválasztotta a saját taskját és így követni tudtuk azt, hogy kinek mi a feladata, illetve, hogy mennyire haladunk a projekttel. A feladatok állapotát 4 tényező szerint követtük:

- **Todo** – ezek voltak a megoldásra váró feladatok
- **In progress** – a csapattagok által kiválasztott, folyamatban lévő feladatok jelennek meg
- **In code review** – a problémás részek jelentek meg
- **Done** – a befejezett task-ok oszlopa



Ábra 11: Kanban

6. Alkalmazás működése

6.1 Front End

A projektünk során a webes frontend részét React segítségével hoztuk létre, ahol az HTML, CSS és JavaScript technológiákat alkalmaztunk. Mobil alkalmazásunk fejlesztése során a React Native keretrendszert választottuk, ahol a TypeScript nyelvet használtuk a hatékonyabb és biztonságosabb kódolás érdekében. Ezek a technológiák lehetővé tették számunkra, hogy egységes felhasználói élményt nyújtsunk mind a webes, mind a mobil platformokon.

6.2 Backend

A backend a szoftver architektúrában az az összetevő, amely a felhasználói felülettől (a mobilalkalmazás esetében React Native-től, és a webalkalmazás esetében React-től) elválasztva működik. Ez a rész felelős az adatok kezeléséért, az alkalmazás logikájáért és az adatbázis-műveletekért. A backend kódját szervereken futtatjuk, amelyek elérhetők az interneten keresztül, és kommunikálnak a frontenddel (React és React Native) a szolgáltatások és adatok biztosítása érdekében.

A backend részben Node.js-t használunk JavaScript és TypeScript nyelveken. Ez lehetővé teszi számunkra, hogy egységes nyelvet használjunk a teljes alkalmazás fejlesztése során, mind a frontend (React és React Native), mind a backend oldalon. Node.js kiválóan alkalmas aszinkron műveletek kezelésére, ami javítja az alkalmazás teljesítményét .

6.2 Adatbázis

Az adatbázisként Firebase-t használtunk, mert gyors és egyszerű beállítást kínál a felhő alapú adatbázis kezeléshez.

Projekt/Adatbázis Függvények

Ebben a részben a projekt néhány fontosabb függvénye kerül szemléltetésre.

Fetch

Ezt a függvényt a projekt szinte minden komponensében használjuk, komponenseken belül változó, attól függően, hogy mit akarunk hívni (FetchData, FetchUsers, FetchProducts, FetchMessages...). A lényege, hogy az adatbázisból adatokat vesz ki. Van, ahol egy dokumentum összes mezőjét "fetch"-eljük, van, ahol csak párat. Mivel a useEffect hook-ot alkalmazzuk, így minden adatbázisban történt módosítás után frissülni fognak az értékek.

```

useEffect( effect () : void => {
  #? Codeium: Refactor Explain Docstring
  // usage: MihacsaiNandor
  const fetchData = async () : Promise<void> => {

    try {
      const currentUser : User = firebase.auth().currentUser;
      console.log(currentUser.uid);

      if (currentUser) {
        const userDoc : DocumentSnapshot<DocumentData, DocumentData> = await getDoc(userDocRef);

        if (userDoc.exists()) {
          const userData : DocumentData = userDoc.data();
          setUsername(userData.userName);
          setUserAddress(userData.address);
          setUserCity(userData.city);
          setUserPhoneNum(userData.phoneNumber);
          setUserEmail(userData.email);
          console.log('User data:', userData);
          myProducts();
        }
      }
    } catch (error) {
      console.error('Error fetching user data:', error.message);
    }
  };

  fetchData();
}, [dep]);

```

Ábra 12: FetchData

Delete

A projekten belül több helyen is használunk delete függvényeket, közülük itt található egy felhasználó által feltöltött termék törlése. A felhasználónak lehetősége van a saját termékei közül törölni, erre a műveletre API hívást használunk.

```

const handleDeleteMP = (productId) : void => {
  deleteDoc(doc(firebase.firestore(), path: 'products', productId))
    .then(() : void => {
      alert('Product deleted');
      // Kitörli a terméket a state-ből
      setProductList( value: (prevProductList : any[] ) : any[] =>
        prevProductList.filter((product) : boolean => product[1] !== productId));
    })
    .catch((error) : void => {
      console.error("Error deleting product: ", error);
    });
};

```

Ábra 13: Delete product

Upload product

Az adatbázisba feltölti a termék információit, illetve a hozzá rendelt képeket, viszont ez csak akkor lehetséges ha minden mezőnek van értéke, másképp visszajelzést küld a felhasználónak, hogy töltsse ki a hiányosságokat. Ha sikeres a feltöltés akkor minden mező értékét törli.

```

const handleSaveChanges = async () : Promise<void> => {
  if (!title || !price || !productDescription || !category || selectedImages.length === 0) {
    alert("Please fill all the fields and select at least one image");
    return;
  }

  try {
    // Fetch category ID asynchronously
    const categoryIdSnapshot : QuerySnapshot<DocumentData> = await firebase.firestore().collection(collectionPath: 'category').where('category_name', '==', category).get();

    if (!categoryIdSnapshot.empty) {
      // Assuming there's only one category with the given name
      const categoryId : string = categoryIdSnapshot.docs[0].id;

      const currentUser : User = firebase.auth().currentUser;

      if (currentUser) {
        const userId : string = currentUser.uid;

        // Create a reference to the 'products' collection
        const productRef : CollectionReference<...> = collection(db, path: 'products');

        // Add product data to Firestore
        const newProductDocRef : DocumentReference<...> = await addDoc(productRef, {
          product_name: title,
          price,
          description: productDescription,
          category: categoryId,
          user: userId,
        });

        // Get the newly created product ID
        const productId : string = newProductDocRef.id;

        // Upload each selected image to Firebase Storage in the 'Products/{productId}' folder
        const storage : FirebaseStorage = getStorage();
        const storageRef : StorageReference = ref(storage, `Products/${productId}`);
      }
    }
  }
};

```

Ábra 13: Upload product 1

```

const storage : FirebaseStorage = getStorage();
const storageRef : StorageReference = ref(storage, 'Products/${productId}');

//Puts the images in firebase
for (const image of selectedImages) {
  const imageRef : StorageReference = ref(storageRef, image.name);
  const uploadTask : UploadTask = uploadBytesResumable(imageRef, image);

  // Wait for the upload to complete
  await uploadTask;

  // Update the product document in Firestore with the image URL
  await updateDoc(doc(db, 'products', productId), {
    images: arrayUnion(image.name),
  });
}

console.log('Product data and images uploaded successfully to Firestore and Storage');
} else {
  alert('User not found');
}
} else {
  console.error("Category not found");
  alert("Category not found");
}
} catch (error) {
  console.error('Error uploading product data to Firestore:', error);
}
setTitle(value: '');
setPrice(value: '');
setProductDescription(value: '');
setCategory(value: '');
setSelectedImages(value: []);
};

```

Ábra 13: Upload product 2

7.Összegzés

A projekt során olyan fázisokba láttunk bele, amiben az eddigiekben nem volt részünk, mint például a kezdeti fejlesztési fázis, amikor megbeszéltük, hogy mit hogyan és miben kell fejlesszünk, illetve hogy egy viszonylagosan kicsi projekt is mekkora munkával jár.

Megtanultuk azt ,hogy milyen más emberekkel együtt dolgozni és hogy milyen módszerekkel kell egymást ösztönözzük ahhoz ,hogy minél eredményesebbek legyünk csapat szintjén. Szerencsénk volt ,hogy volt egy tapasztaltabb társunk aki jobban átlátta a dolgokat és így gyorsabban és magabiztosabban tudtunk haladni,

illetve, hogy rengeteget kommunikáltunk és megosztottuk úgy a pozitív, mint a negatív tapasztalatainkat a projekt készítése során.

Másrészt a fejlesztés során megismerkedtünk a firestore, React, React native technológiákkal , illetve a javascript és typescript nyelvekkel, mindezek olyan ismeretek, amivel a későbbiekben foglalkozni szeretnénk.

Mindannyian arra a következtetésre jutottunk, hogy csapatban dolgozni sokkal érdekesebb és kifizetődőbb az egyéni munkához képest, úgy az élmény, mint tapasztalat szempontjából.

8.1.Hibák javítása

Product Details Page – szebben megjeleníteni a képeket telefonon és weben is.

Product By Search oldalon kijavítani a hibát, hogy megjelenítse a képeket ha el akarjuk érni a Product Details Page- et.

Az oldalakon a reszponzivitást javítani, különböző nézetekre weben.

8.2.Továbbfejlesztési lehetőségek

- Felhasználói felület szépítése
- Admin Page kibővítése “modify” funkcióval
- A My profile oldalon a felhasználó legyen képes profilképet beállítani.
- Hibák kijavítása.

9.Bibliográfia

<https://stackoverflow.com/>

<https://openai.com/blog/chatgpt>

<https://firebase.google.com/docs/auth/web/start>

<https://www.w3schools.com/html/>

<https://www.w3schools.com/css/>

<https://uiverse.io/>

<https://legacy.reactjs.org/tutorial/tutorial.html>

<https://reactnative.dev/docs/tutorial>