

BABEȘ–BOLYAI UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF MATHEMATICS AND INFORMATICS
SPECIALIZATION: COMPUTER SCIENCE

Diploma Thesis

Category based C2C marketplace



ADVISOR:

DR. DOKA-MOLNÁR ANDREA ÉVA,
LECTURER

STUDENT:

BAUER KRISTÓF

2021

UNIVERSITATEA BABEȘ-BOLYAI, CLUJ-NAPOCA
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
SPECIALIZAREA INFORMATICĂ

Lucrare de licență

Piață C2C bazată pe categorizare



CONDUCĂTOR ȘTIINȚIFIC:

LECTOR DR. DOKA-MOLNÁR ANDREA
ÉVA

ABSOLVENT:

BAUER KRISTÓF

2021

BABEŞ–BOLYAI TUDOMÁNYEGYETEM KOLOZSVÁR
MATEMATIKA ÉS INFORMATIKA KAR
INFORMATIKA SZAK

Szakdolgozat

Kategorizáláson alapuló C2C piactér



TÉMAVEZETŐ:

DR. DOKA-MOLNÁR ANDREA ÉVA,
EGYETEMI ADJUNKTUS

SZERZŐ:

BAUER KRISTÓF

2021

BABEŞ–BOLYAI UNIVERSITY OF CLUJ-NAPOCA
FACULTY OF MATHEMATICS AND INFORMATICS
SPECIALIZATION: COMPUTER SCIENCE

Diploma Thesis

Category based C2C marketplace

Abstract

The Marketplace web application is based on a C2C (Customer to Customer) business model, where natural persons can sell their products.

The products for sale on the site are defined by categories, therefore users can quickly and accurately find the products they are looking for, even among the wide range of products.

The objectives of the application include customizability, a user-friendly and simple user interface, and the exploration and rethinking of new features that are not present in current marketplaces.

This work is the result of my own activity. I have neither given nor received unauthorized assistance on this work.

2021

BAUER KRISTÓF

ADVISOR:
DR. DOKA-MOLNÁR ANDREA ÉVA,
LECTURER

Tartalomjegyzék

1. Bevezető	5
1.1. Motiváció	5
1.2. Célok	6
1.3. Hasonló C2C üzletmodellel alapuló weboldalak összehasonlítása	7
1.4. Fejezetenkénti rövid leírás	8
2. The Marketplace projekt	9
2.1. Funkcionalitások	9
2.1.1. A webes kliens funkcionálisai	9
2.1.2. Az alkalmazás felépítése	13
2.2. Architektúra	13
3. Kliens oldali technológiák	14
3.1. React.js	14
3.2. Redux	14
3.3. Axios.js	15
3.4. Material-UI	15
3.5. Ant Design	15
3.6. Moment.js	15
3.7. Socket.IO Client	16
4. Szerver oldali technológiák	17
4.1. Node.js	17
4.2. Express	17
4.3. Socket.IO	17
4.4. Mongoose	18
4.5. MongoDB	18
4.6. JSON Web Token	18
5. Fejlesztési eszközök	19
5.1. Visual Studio Code	19
5.2. Gitlab	19
5.3. MongoDB Compass	20
5.4. Postman	20
6. Megvalósítási részletek	21
6.1. Szerver oldali megvalósítás	21
6.1.1. Node.js és Express szerver megvalósításai	21
6.1.2. Útválasztó réteg	21
6.1.3. Szolgáltatási réteg	22
6.1.4. Adatelérési réteg	22
6.1.5. A Socket.IO szerver megvalósítása	23
6.2. Kliens oldali megvalósítás	24
6.2.1. Egységek	24
6.2.2. Hitelesítés	26
6.3. Az alkalmazás adatmodellje	27
7. The Marketplace projekt működése	29
7.1. Regisztráció, Bejelentkezés és Profil információk kibővítése	29
7.2. Navigáció	31
7.3. Termék hozzáadása	31
7.4. Eladó termékek megtekintése és megjelenítési formái	34
7.5. Termékek szűrése	36

TARTALOMJEGYZÉK

7.6.	Termék adatlapja	41
7.7.	Árajánlat küldése	42
7.8.	Vásárlási kérelmek	43
7.8.1.	Vásárlási kérelmek létrehozása	43
7.8.2.	Vásárlási kérelmek megtekintése	44
7.8.3.	Egy termékre illeszkedő vásárlási kérelmek megtekintése	45
7.8.4.	Termékünk elküldése egy vásárlási kérelemre	45
7.9.	Profil	46
7.9.1.	Profil komponens	46
7.9.2.	Termékek komponens	47
7.9.3.	Ajánlatok komponens	47
7.9.4.	Eladott termékek komponens	50
7.10.	Termékstatisztikák	52
8.	Összefoglaló	54
8.1.	Következtetések	54
8.2.	Továbbfejlesztési lehetőségek	54

Ábrák jegyzéke

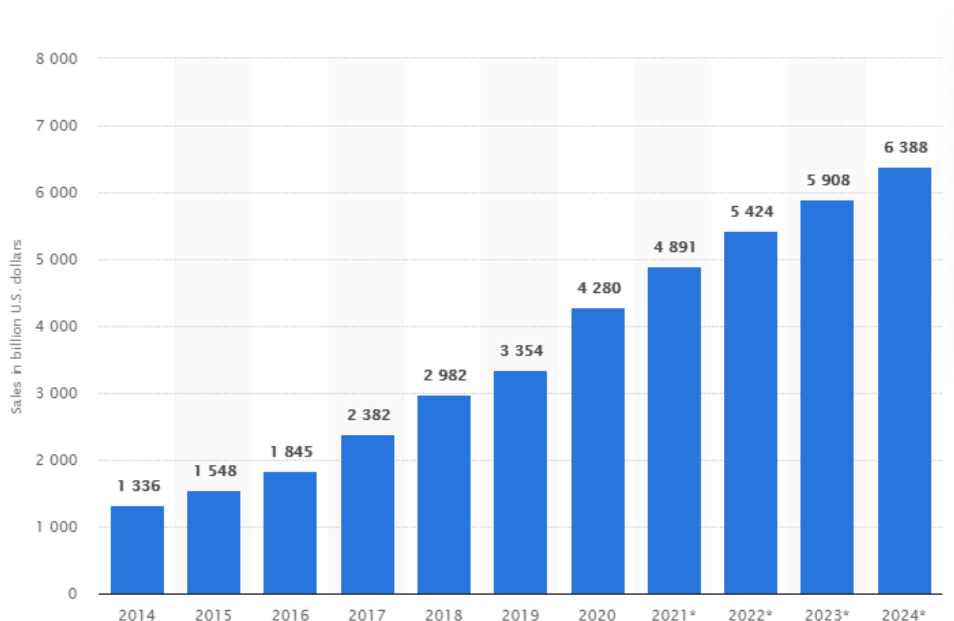
1.1.	Internetes eladások ár alakulása	5
1.2.	Az online történő vásárlások eloszlása platform és üzlettípus szerint	6
2.1.	Árajánlatok és egy vásárlási kérelem	10
2.2.	A The Marketplace használati eset diagramja	12
2.3.	Az alkalmazás architektúrája	13
6.1.	A Redux munkafolyamatai	26
6.2.	A requests kollekció két dokumentuma	28
7.1.	Regisztráció	29
7.2.	Profil információk kibővítése	30
7.3.	Navigációs menü	31
7.4.	Termék főkategóriák	31
7.5.	Autó alkategória előre meghatározott kategóriái	32
7.6.	Példa egy termék képeinek hozzáadására, illetve egy saját érték hozzáadására egy már előre definiált kategóriához	33
7.7.	Példa saját kategóriák hozzáadására, illetve más felhasználók által hozzáadott saját kategóriák újrafelhasználására	33
7.8.	Termékek főoldala	34
7.9.	Csoportosított termékek kártyája és táblázata	35
7.10.	Cím szerinti termékszűrés	36
7.11.	Árintervallum szerinti termékszűrés	36
7.12.	Kulcsszavak szerinti termékszűrés	37
7.13.	Hierarchikus termékszűrés	38
7.14.	Fejlett termékszűrés	39
7.15.	Saját kategóriák hozzáadása a szűrési opciókhoz	40
7.16.	Szűrési kategóriák törlése és újbóli hozzáadása	40
7.17.	Termék adatlap	41
7.18.	Alkudható ártípus esetén ajánlat küldés	42
7.19.	Vásárlási kérelem létrehozása	43
7.20.	Vásárlási kérelmek megtekintése	44
7.21.	Egy termékre illeszkedő összes vásárlási kérelem táblázata	45
7.22.	Termékünk elküldése egy vásárlási kérelemre	45
7.23.	Profil komponensek közötti navigáció	46
7.24.	Profilunk adatainak módosítása	47
7.25.	Termékek komponens	47
7.26.	Ajánlatok komponens főmenü felépítése	48
7.27.	Kapott és küldött árajánlatok	48
7.28.	Létrehozott, kapott és küldött kérelmek	50
7.29.	Egy eladott termék kártyája	51
7.30.	Szállítási állapot	51
7.31.	Kiválasztható időpontok naptára	52
7.32.	Eladott termékek grafikonja	53
7.33.	Adott napon eladott termékek	53

1. fejezet

Bevezető

1.1. Motiváció

Jelenleg ahogy az internet egyre több emberhez elér, úgy mutat növekvő tendenciát az internetes eladások és vásárlások száma is.



1.1. ábra. Onlile eladások áralakulása és előrejelzése 2014-2024 között

<https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/>

A felhasználók két típusú helyről vásárolhatnak online: **üzletekből** azaz storeokból vagy **piactérről** azaz marketplacéről. Mindkettőnek megvannak az előnyei és hátrányai:

- Üzletek: Termékspecifikus, gyors és könnyű termék keresést biztosítanak, viszont limitált egy adott termékkategória szerint és csak vásárolni lehet.

1. FEJEZET: BEVEZETŐ

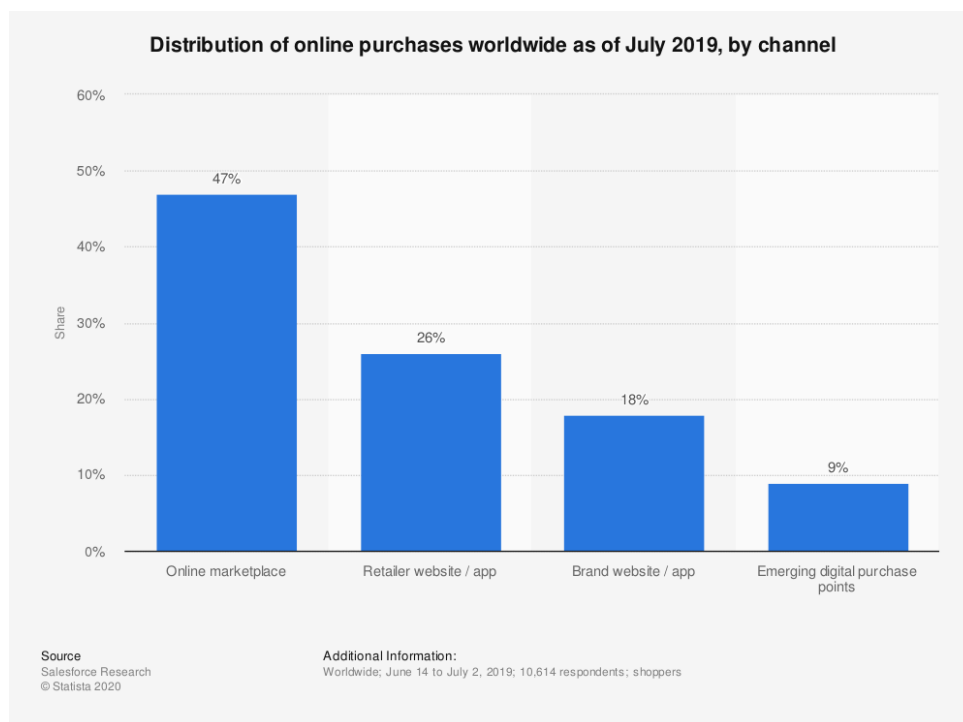
- Piac tér: Bárki lehet eladó és bárki vásárolhat. Nagyobb a termékkínálat, viszont nehezebben lehet termékeket keresni és behatárolni, mivel túl nagy a választék.

Az alábbi elválasztásból adódóan nem létezik olyan üzlet, amely nagy termékkínálattal és termékspecifikus funkciókkal rendelkezne, hogy a felhasználók gyorsan és pontosan megtalálják az általuk keresett terméket. Ez rossz az eladónak, mert kevesebb emberhez jut el a terméke, és rossz a vásárlónak aki sok időt kell eltöltsön az általa vágyott termék megtalálásához.

1.2. Célok

A The Marketplace egy olyan webes alkalmazás, amelyik mindkét üzletforma pozitívumait ötvözi egy eredeti módon, kiegészítve új funkciókkal.

Mivel az online piacterek nagy népszerűségnek örvendenek erre épül a The Marketplace webalkalmazás is. A piactérnek egy **C2C**¹ típusát valósítja meg. A C2C üzletmodell szerint a tranzakció két magánszemély között jön létre áruk eladása vagy vásárlása céljából, bővebben az üzletmodellről: Tarver [11].



1.2. ábra. Az online történő vásárlások eloszlása platform és üzlettípus szerint
<https://www.statista.com/statistics/861336/share-online-shopping-customers-vs-sales-by-platform/>

¹C2C a Customer To Customer rövidítése

1. FEJEZET: BEVEZETŐ

A The Marketplace webalkalmazáson belül lehetősége van egy felhasználónak új vagy használt termékét eladnia. A piactérből adódó színes termékskálát megőrzi az alkalmazás és ötvözi egy üzlet termékspecifikusságával az alábbi módon: a piactérre jellemző cím és termékleírást elhagyja, helyette a felhasználóknak egy főkategóriába és egy alkategóriába kell besorolniuk a termékeiket. Az oldal ezután biztosít nekik az alkategóriához tartozó további specifikus kategóriákat a megfelelő értékekkel.

Ezt követően a teljes körű szerkesztés a felhasználóra van bízva: a felhasználó törölheti a biztosított kategóriákat, hozzáadhat azok értékeihez saját értéket, bevezethet a termékére jellemző saját kategóriát saját értékkel. Az összes hozzáadott egyedi értéket és saját kategóriát más felhasználók is újra fel tudják használni az ő termékük létrehozáskor, ezáltal egy folyamatosan változó, releváns adatokat tartalmazó termék adatbázist fognak kapni a felhasználók és kényelmesebbé válik a terméklétrehozás is.

Az alábbi testreszabhatóság azt eredményezi, hogy az oldalon bármilyen terméket hozzá tudnak adni piactér szerűen a felhasználók és az üzletekre jellemző termékspecifikusságot a kategorizálás valósítja meg, ami által az alkalmazás egy gyors és pontos termék meghatározást biztosít a széles termékskálán.

Az oldalon továbbá olyan új vagy újragondolt egyedi funkciók kerülnek bevezetése, amik az eddigi piactéri alkalmazásokra nem jellemzőek: vásárlási kérelmek, újragondolt relevancián alapuló többféle szűrési mód, eladott termékek áralakulásainak a figyelésére. A felhasználók eladásai és vásárlásai kezelésére egy sokkal egyszerűbb mechanizmust vezet be ami az ajánlatok rendszere, egyszerűbb és gyorsabb lett a felek közötti kommunikáció, illetve eladóként előre megszabhatjuk a termékeinkre érkező árajánlatokat is.

1.3. Hasonló C2C üzletmodellen alapuló weboldalak összehasonlítása

A legnépszerűbb weboldalak, amelyek a piactér szerű üzletek általában a C2C üzletmodell szerint működnek. A teljesség igénye nélkül az alábbi fejezetben egy népszerű külföldi és itthoni C2C weboldal pozitívumai és negatívumai lesznek a The Marketplace webalkalmazáshoz hasonlítva.

Ebay

- Pozitívumok: Nagy a termékfelhozatal és a vásárlói bázis. A Paypalal egyszerű a fizetés. Könnyű a vásárlás. A vásárlókat a cég garantálja és kárpótolja, ha sérült vagy nem érkezett meg egy termék. Az eladókat és a termékeket lehet osztályozni és véleményt fűzni hozzájuk

1. FEJEZET: BEVEZETŐ

- Negatívumok: Jutalékot kér az eladott termékek után. Paypal fizetéssel lehetnek gondok. Az oldal a cégeket jobban támogatja a magánszemélyekkel szemben. Nincs termékspecifikus keresés és a kategóriák túl tágak, hogy pontosan megtaláljuk a keresett terméket, a termékkategóriák nem testreszabhatók, nem lehet vásárlási szándékot létrehozni és termékek áralakulását nézni. Nem látjuk vásárlóként, hogy mások milyen árajánlatokat tettek egy termékre.

OLX

- Pozitívumok: Nem kerül pénzbe a termékek hozzáadása egy limitált számig. Nem csak termékeket lehet eladni, hanem szolgáltatásokat is kínálhatunk. Álláshirdetések is szerepelnek az oldalon, illetve örökbefogadások is. Egyszerű a működése.
- Negatívumok: Az oldalon nincs lehetőség a termékek szűrésére csak termék cím, illetve kategória és ár szerint. A lényegi információkat csak a termék leírásában lehet részletezni. Vásárlóként nehéz a kapcsolatfelvétel az eladóval és eladóként nem tudjuk előre szűrni a beérkező ajánlatokat. Nem látjuk vásárlóként, hogy mások milyen árajánlatokat tettek egy termékre. A termékkategóriák nem testreszabhatók, nem lehet vásárlási szándékot létrehozni és termékek áralakulását nézni.

1.4. Fejezetenkénti rövid leírás

Az első *Bevezető* című fejezetben az alkalmazás motivációi és céljai vannak részletezve. Tartalmaz egy rövid összefoglalást az újdonságokról funkciók terén, illetve összehasonlításra kerül a hasonló tematikán alapuló népszerű külföldi és itthoni weboldallal.

A második *The Marketplace projekt* fejezetben, az alkalmazás funkciói vannak részletezve a webes kliens és architektúrális megközelítés szerint.

A harmadik, negyedik és ötödik fejezet tartalmazza a felhasznált *kliens* és *szerveroldali technológiákat* és az alkalmazás *fejlesztési eszközeit*.

A hatodik fejezetben a *Megvalósítási részletek* kerülnek bemutatásra.

A hetedik fejezetben részletesen, képek segítségével a *The Marketplace projekt működése* van leírva.

A nyolcadik fejezet, melynek címe *Összefoglaló*, egy rövid fejezet a projekt értékeléséről és a továbbfejlesztési lehetőségekről.

2. fejezet

The Marketplace projekt

Összefoglaló: A The Marketpalce egy olyan webalkalmazás, ami arra szolgál, hogy az oldalon szereplő termékeket minél pontosabban illeszkedő kategóriák által határozzuk meg, ezáltal a felhasználók gyorsan és pontosan találhatják meg az általuk keresett termékeket.

2.1. Funkcionalitások

Az alábbi fejezet arra szolgál, hogy bemutassa azokat a fő funkciókat, amelyek a The Marketplacen fellelhetők, és teljesen újak vagy újragondoltak egy C2C piactéri üzletmodell esetében. A funkcionálisok reprezentációja megtekinthető *A The Marketplace használati eset diagramja* fejezetnél.

2.1.1. A webes kliens funkcionálisai

Regisztráció, bejelentkezés és profil kibővítése

Az oldalon lehetőség van regisztrálni és bejelentkezni Google fiókkal vagy egy egyedi felhasználónévvel. Regisztráció után az alkalmazás létrehoz számunkra egy profilt. Az oldal fő funkciói regisztrálás nélkül is elérhetőek, ám ahhoz hogy termékekre ajánlatokat tegyünk vagy vásárlási kérélmeket hozhassunk létre szükséges a regisztráció és a profilunk kibővítése. Profil kibővítése során a meglévő profilunkon meg kell adjuk a személyes adatainkat. Ezeket egyszer kell megadni és később bármikor módosíthatóak. Haszna, hogy egy sokkal gyorsabb és egyszerűbb kommunikációt fog eredményezni köztünk és partnereink között egy sikeres termékadást követően.

Ajánlatok és vásárlási kérélmek

Az oldalon vásárlást és eladást tekintve két fő felhasználói tevékenység valósítható meg:

1. Az oldalon lévő termékekre elküldhetjük saját **ár ajánlatunkat**, ezzel jelezve szándékunkat az eladó felé, hogy szeretnénk megvásárolni a termékét.

2. FEJEZET: THE MARKETPLACE PROJEKT

2. **Vásárlási kérelmeket** hozhatunk létre, amik az apróhirdetésekhöz hasonlítanak. Itt beállíthatunk kategóriák és értékek alapján több kritériumot és árintervallumot is. Ha egy eladónak a termékre megfelel az általunk felállított kritériumoknak, elküldheti a vásárlási kérelmünkre termékét, amit mi meg tudunk vásárolni a kérelem elfogadásakor.

Offers

User	Offer	Date
teszt3	\$56000	2021-06-06 22:59
bauerk	\$60000	2021-06-04 20:22

1-2 of 2 < >

Buy request from 1 user

Criteria
Category: Cars
City: Cluj Napoca, Zalau
Condition: New
Exact Model: 40, E46 Sport
Price starting from 1 to 131186

Send an offer
>

2.1. ábra. Árajánlatok és egy vásárlási kérelem

Ajánlatok rendszere

A termékeinkre érkező árajánlatokat, az általunk küldött árajánlatokat, a létrehozott kérelmeket, a kérelmünkre érkező termékeket és általunk kérelmekre elküldött termékeinket egy közös helyen tudjuk nagyon egyszerűen kezelni. Mindegyik partnertől származó ajánlat vagy kérelem külön van kezelve és mindegyik partnerrel folytathatunk magánbeszélgetést chatelés által, valós időben.

Az összes felhasználói tevékenységnek létrejön egy kis kártya, ami tartalmazza az adott termék információt és az ajánlat részleteit. Egy érkező ajánlatot visszautasíthatunk vagy elfogadhatunk, az általunk küldött ajánlatokat pedig visszavonhatjuk. Egy ajánlat elfogadása esetén beleegyezőnk abba, hogy a termékünket eladtuk a partnerünknek vagy kérelem esetén elfogadtuk a partner termékét. Ekkor ez lekerül a piactérről és az összes felhasználó, aki bármilyen módon érintett volt a termék által, értesítve lesz valós időben socketek útján.

Termék létrehozása

Az oldalon kilenc főkategória található. Mindegyik főkategória alkategóriákkal rendelkezik, ezekből összesen negyvenegy darab van. A főkategória és az alkategória közötti kapcsolat hierarchikus. Az alkategória kiválasztásakor megjelennek az arra jellemző, az oldal által előre

2. FEJEZET: THE MARKETPLACE PROJEKT

definiált kategóriák, ezekből több mint négyszáz van összesen. Ezen kategóriák közötti kapcsolat már relatív, strukturális szempontból az összes ugyan azon a szinten van. Minden kategória, ami az oldal által meghatározott, rendelkezik ugyancsak az oldal által előre meghatározott értékekkel, ezekből is összesen több mint ötszáz van.

Egy felhasználó, amikor terméke eladása mellett dönt be kell sorolnia azt egy fő és alkategóriába. Ezt követően megjelenik egy termékadatlap az oldal által meghatározott kategóriákkal és értékekkel. Itt hozzáadhatja a felhasználó a terméke képeit, illetve az adatlapot terméke szerint személyre szabhatja, kitörölve a kategóriákat, kategóriákhoz saját értéket adhat hozzá vagy saját kategóriát és saját értéket is hozzáadhat ha egyik előre meghatározott kategória sem illeszkedik termékére. Más felhasználók által meghatározott saját kategóriákat is hozzáadhat termékéhez.

Amikor befejezte a terméke adatlapjának a szerkesztését az összes hozzáadott új érték és saját kategória a többi felhasználó számára is elérhetővé válik és felhasználhatják saját termékük adatlapjának a szerkesztésekor.

Termékek szűrése

A termékek szűrésére 5 lehetőségünk van

1. Cím szerint: Termék címe szerint szűrhetünk, szöveg bevitele által
2. Ár szerint: Megszabhatjuk azt az árintervallumot, amiben érdekelnek a termékek
3. Kulcsszavak szerint: Megadhatunk több kulcsszót és bármelyik olyan termék, amelyiknek kategóriáiban vagy értékeiben teljesen vagy részlegesen szerepel a kulcsszó szűrésre kerül
4. Hierarchikus szűrés: célja egy adott termékcsoporthoz minél pontosabb meghatározása és leszűkítése
5. Fejlett szűrés: több termékcsoporthoz pontos meghatározása és leszűkítése

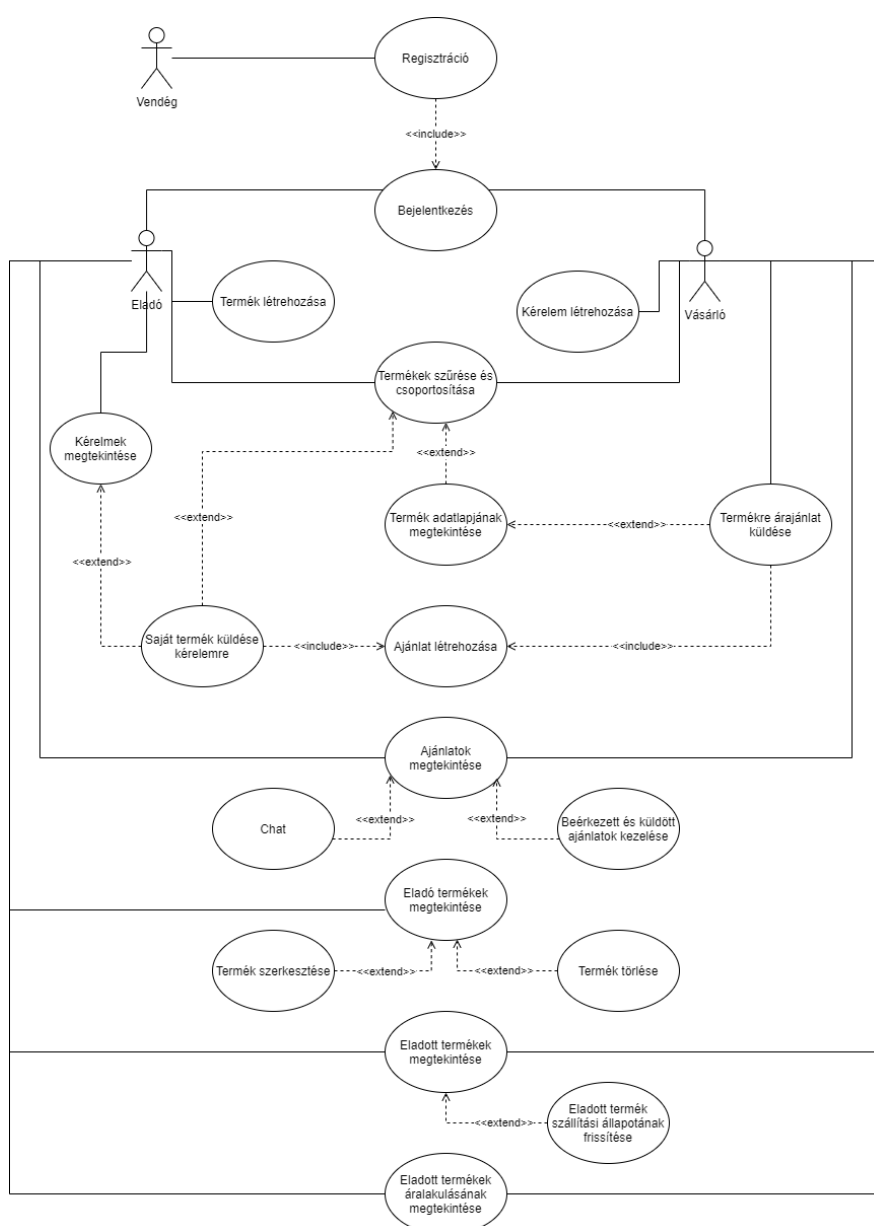
Bár mindegyik szűrési forma más, az összes esetében igaz az, hogy csak releváns adatok szerint szűr. Ez azt jelenti, hogy a szűrési kategóriák és értékek folyamatosan dinamikusan változnak és csak olyan értékek fognak megjelenni, amik az eladó termékek adatlapjában is szerepelnek.

Ezáltal mindig pontos termékeket kapunk és nem fordul az elő, mint sok internetes weboldal esetében, hogy szűrünk egy olyan kategória vagy érték szerint, ami már nem létezik a termékek között.

Eladott termékek áralakulása

Lehetőségünk van az eladott termékek áralakulásainak a megtekintésére. Ki kell válasszunk egy fő és alkategóriát, megadhatunk különböző kritériumokat, majd egy dátumintervallum kiválasztásával meg fog jelenni egy grafikon az árak alakulásáról. A grafikonon lévő árak az adott napon eladott termékek átlagárából számolódik. A grafikon csomópontjaira kattintva pedig megnézhetjük az adott napon eladott termékeket és adatlapjaikat.

A The Marketplace használati eset diagramja

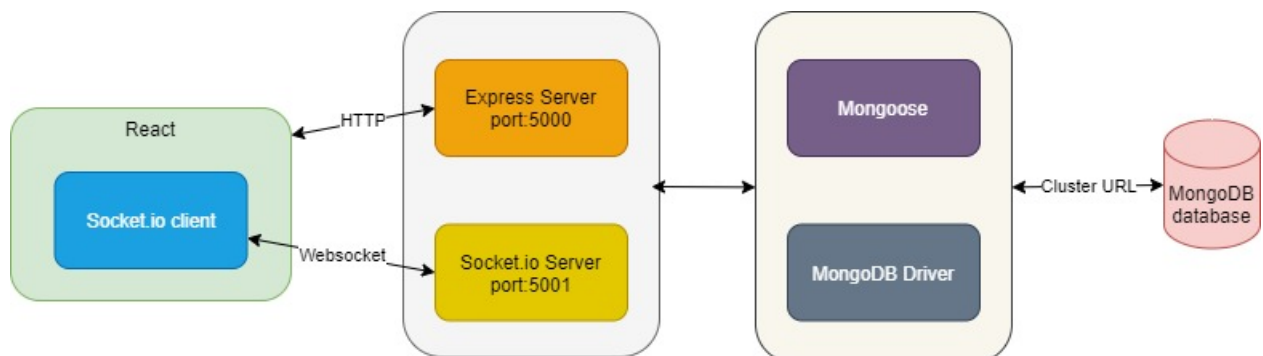


2.2. ábra. A The Marketplace használati eset diagramja

2.1.2. Az alkalmazás felépítése

A The Marketplace alkalmazás kliens oldali része a React.js-re épül. A szerver oldalon a Node.js-en futó Express, illetve Socket.IO webserverek fogadják a kliens oldal kéréseit. Az adatokat a MongoDB NoSQL adatbázisa perzisztálja.

2.2. Architektúra



2.3. ábra. Az alkalmazás architektúrája

A fenti ábra az alkalmazás architektúráját szemlélteti. A kliens oldali React alkalmazás kommunikálhat a Node.js Express szerverrel HTTP kérések által, valamint Socket.IO szerverrel, a Socket.IO kliensoldali Websocket API-ának segítségével. A Socket.IO segít a felcsatlakozott kliensek közötti valós idejű, full-duplex kommunikáció megvalósításában.

A szervereink és a MongoDB adatbázis közötti kapcsolatot a Mongoose, valamint a MongoDB Driver valósítják meg. A két könyvtár használata az adatstruktúrából adódik, ugyanis az ODM¹ struktúrájú adatmodelleket a Mongooseal, a félig strukturált adatmodelleket és bonyolultabb lekérdezéseket a Node.js MongoDB Driver segítségével használjuk.

¹ODM - Object Data Modeling rövidítése

3. fejezet

Kliens oldali technológiák

***Összefoglaló:** Az alábbi fejezetben a kliensoldali technológiák kerülnek bemutatásra.*

3.1. React.js

A **React.js** ([5]) napjaink egyik legnépszerűbb kliens oldali könyvtára . Segítségével egyszerű és deklaratív komponensekből könnyen tudunk bonyolult felhasználói felületeket is létrehozni. Mindegyik komponens újrafelhasználható és saját belső állapottal rendelkezik, amiket a többi komponenssel is megoszthat, ekkor azok tulajdonsága lesz. A komponensek elrendeződése hierarchikus, a virtuális DOM fa struktúráját követi. Csak akkor renderelődik újra egy komponens, amikor annak állapota is változik, és ezt anélkül teszi, hogy újratöltődne az oldal. A kinézetet a React könyvtár által bevezetett JSX-be ¹ írjuk, ami egy HTML szerű szintaxis és megengedi a fejlesztők számára a HTML kód írását JavaScriptben.

Összességében a React.js egy gyors, skálázható és egyszerű könyvtár, ami által könnyedén készíthetünk bonyolult alkalmazásokat.

3.2. Redux

A **Redux** ([6]) egy centralizált állapot kezelő rendszer. Konzisztencián alapszik, ezáltal kiszámítható az alkalmazás állapota. Ez, a közvetlen módon változtathatatlan (immutable) állapot miatt valósul meg amelyet egy storeban tárol. A store több kisebb állapot rész szerint van felosztva, mindegyik változtatásáért egy-egy reducer felel. A reducereket a felhasználói felülettől érkező actionokra reagálnak. Egy actiont az összes reducer megkapja, viszont annak típusa alapján, csak az annak megfelelő reducer változtatja az állapotrészt, a többi nem. Ez egy olyan globális alkalmazás állapotot eredményez, ami könnyen tesztelhető és minden időpillanatban egyértelműen megfigyelhető.

¹JSX - JavaScript XML rövidítése

3.3. Axios.js

Az **Axios.js** ([1]) egy Promise alapú HTTP kliens könyvtár, ami által kéréseket küldhetünk kliens oldalról a Node.js szerveroldalunknak. Kérés küldésekor több előre definiált opciót is tartalmaz, ami megkönnyíti a kérés fejlécének és testének beállítását. A szervertől kapott választ az Axios Promise-ra alakítja, ami által egy könnyebb és biztonságosabb adatfeldolgozást tudunk végrehajtani a kliens oldalon.

3.4. Material-UI

React használata esetén a **Material-UI** ² az egyik legnépszerűbb CSS keretrendszer. Egy olyan komponens könyvtár, amely egyesíti a Reactot és a **Material Design** ³. A könyvtár előre meghatározott és stilizált React komponenseket tartalmaz, amiket aztán személyre szabhatunk alkalmazásunknak megfelelően. A Material UI komponenseinek kombinációja által gyorsan felépíthető és szép kinézetet tudunk létrehozni, amely a stílus standardoknak is megfelel.

3.5. Ant Design

Egy másik népszerű komponens könyvtár az **Ant Design** ⁴, amely üzlet szintű web alkalmazások felületének a létrehozásában segít. Hasonlóan a Material-UI-hoz, az Ant Design is előre stilizált komponenseket nyújt. A Material UI-hoz képest, az Ant Design több előre meghatározott komponens kombinációt sűrít egy komponensbe, így a komplexebb felhasználói felületek létrehozását még egyszerűbbé teszi.

3.6. Moment.js

A **Moment.js** ([2]) könyvtár által könnyen és egyszerűen formázhatjuk, lokalizálhatjuk, validálhatjuk, manipulálhatjuk, összehasonlíthatjuk és átalakíthatjuk dátumainkat. A beépített függvényei és sémái által a dátumainkat igényeink szerint bármilyen dátum formátumban testreszabhatjuk és megjeleníthetjük olvasható formában.

²Material-UI Hivatalos weboldala: <https://material-ui.com/> (Látogatva: 2020.06.07)

³Material Design Hivatalos weboldala: <https://material.io/resources/get-started> (Látogatva: 2020.06.07)

⁴Ant Design Hivatalos dokumentációja: <https://ant.design/docs/react/introduce> (Látogatva: 2020.06.07)

3.7. Socket.IO Client

Ahhoz hogy a kliens oldalunk valós időben egy bidirekcionális kommunikációt valósítson meg a szerver oldali Socket.IO szerverünkkel szükséges a **Socket.IO Client** ([8]) könyvtárat használnunk. A Socket.IO Client egy burkoló könyvtára a Socket.IO saját WebSocket API-nak. A kliens által tudjuk létrehozni és beállítani a WebSocket kapcsolatunkat a szerverrel, illetve az adatok küldésére és válaszok fogadására használjuk.

4. fejezet

Szerver oldali technológiák

Összefoglaló: Az alábbi fejezetben a szerveroldali technológiák kerülnek bemutatásra.

4.1. Node.js

A **Node.js** ([4]) a Google V8-as JavaScript értelmező motorja elválasztva a böngészőtől. Ezáltal egy olyan szoftver rendszer, amely skálázható hálózati alkalmazások készítésére lett tervezve. Bár önmagában nem szerveroldali technológia, a rá épülő különböző könyvtárak és keretrendszerek által, főként webszerverek készítésére használják. Hatékony, mivel aszinkron és esemény vezérelt és előnyös adatintenzív alkalmazások esetén. Node.js-ben a fő programozási nyelv a JavaScript, de támogatott a Typescript is.

4.2. Express

Az **Express** ¹ egy minimalista és rugalmas Node.js keretrendszer, amely mögött robusztus funkciók találhatók. Az Express tartalmazza azokat az alapvető webalkalmazás funkciókat, amelyek segítségével gyorsan és egyszerűen tudunk fejleszteni nagy teljesítményű alkalmazásokat. Ingyenes és nyílt forráskódú ezáltal bárki számára elérhető.

4.3. Socket.IO

A **Socket.IO** ([7]) könyvtár által lehetőségünk van valós idejű, bidirekcionális full-duplex esemény vezérelt kommunikáció megvalósítására egy kliens és szerver között. A Socket.IO a WebSocketken alapuló kliens-szerver közötti kommunikációt könnyíti meg azáltal, hogy egy saját kliens és szerver oldali WebSocket API-t biztosít és azon keresztül kommunikál. A szerver oldali Socket.IO egy standard Node.js HTTP szerverre épül és WebSocket csatlakozásokra vár,

¹Express Hivatalos dokumentációja: <https://expressjs.com/en/4x/api.html> (Látogatva: 2020.06.07)

4. FEJEZET: SZERVER OLDALI TECHNOLOGIÁK

ha nem tudja a WebSocket protokollt létrehozni a kliens és szerver között, akkor visszaesik a kommunikáció egy hosszan tartó HTTP kapcsolatra.

A Websocketek előnye, hogy háromszor gyorsabb kommunikációt eredményez, mint a HTTP kérések, ezáltal valós idejű kommunikációra tehetünk szert. Negatívuma, hogy egy saját WebSocket szerver kialakítása sok időt vesz igénybe, viszont a Socket.IO által egy biztonságos, egyszerű és hatékony implementációt kapunk.

4.4. Mongoose

A **Mongoose**² egy Object Data Modeling (**ODM**) könyvtár a Node.js számára. Segítségével objektumainkat sémákká alakíthatjuk és a Mongoose ezeket MongoDB dokumentumokként reprezentálja. Így, a MongoDB Node.js Driverrel által, különböző műveleteket végezhetünk velük. A Mongoose által séma validációt tudunk végezni és elegáns módon burkol több olyan funkciót is, amit a MongoDB Driverrel sokkal bonyolultabban tudunk elérni.

4.5. MongoDB

A **MongoDB** ([3]) a legnépszerűbb dokumentum-orientált adatbázis szoftver. Típusát tekintve a NoSQL adatbázisszervek közé tartozik. A dokumentumokat BSON reprezentáció szerint tárolja, ez egy JSON-szerű struktúra, kulcs értékek szerepelnek bennük. A MongoDB egy skálázható és nagyon flexibilis adatbázis, amely gyors és könnyű adatelérést és indexelést biztosít.

4.6. JSON Web Token

A **JSON Web Token**³ rövidítve **JWT**, két fél között kompakt, önleíró, biztonságos információcserét biztosít JSON objektumok formájában. A biztonságos adattovábbításért felel, ezt úgy éri el, hogy a JSON-ban lévő adatokat digitálisan aláírja. Az aláírás egy privát és egy publikus kulcspárt igényel, ebben az esetben a privát kulccsal rendelkező felek tekinthetők megbízható feleknek. Szinte minden programozási nyelvhez található egy JWT könyvtár, ahogy a Node.js esetében is.

A JWT, az alkalmazás hitelesítési részének egyik kulcsfontosságú eleme, hiszen enélkül bárki hozzáférhetne az adatbázisunkhoz és módosíthatná annak adatait.

²Mongoose Hivatalos dokumentációja: <https://mongoosejs.com/docs/guide.html> (Látogatva: 2020.06.07)

³JSON Web Token Hivatalos dokumentációja: <https://jwt.io/introduction> (Látogatva: 2020.06.07)

5. fejezet

Fejlesztési eszközök

Összefoglaló: Az alábbi fejezet bemutatja az alkalmazás fejlesztése során használt eszközöket, amik segítettek az alkalmazás gyorsabb és kényelmesebb implementációjában.

5.1. Visual Studio Code

A **Visual Studio Code** ¹ egy nyílt forráskódú kódszerkesztő program, könnyű, ám ugyanakkor mégis nagyon erős fejlesztési eszköz. Segítségével egyszerűen lehet bővítményeket hozzáadni, ilyen például a fejlesztés során használt **ES7 React/Redux/GraphQL/React-Native snippets**, ami olyan funkciókat biztosított, mint: automatikus kódgenerálás, kód színezés és kiemelés, szintaxis ellenőrzés. A Visual Studio Code alapfunkciói hasznosak voltak verziókövetés során, kódformázáskor, kódrészlet keresésében és átnevezésekben is. A Visual Studio Code ezáltal nagyon sok kényelmi és ellenőrzési funkciót biztosított a fejlesztés során.

5.2. Gitlab

A **Gitlab** ² egy olyan platform, amely lehetővé teszi a felhasználók számára a teljes szoftverfejlesztési és a működési életciklus kezelését. Tartalmazza az alap Git funkciókat, mint a verziókövetés, kódunk tárolása és megtekintése, folyamatos integráció, de további új funkciókat is biztosít a fejlesztőknek, mint például az ágak gráfszerű ábrázolása, merge requestek, kódok elemzése és értékelése. A fejlesztés során az átláthatóság és kezelhetőség szempontjából hasznosak bizonyultak a merge requestek és a gráfszerű nézet.

¹Visual Studio Code Hivatalos dokumentációja: <https://code.visualstudio.com/docs> (Látogatva: 2020.06.07)

²Gitlab Hivatalos weboldala: <https://about.gitlab.com/stages-devops-lifecycle/> (Látogatva: 2020.06.07)

5.3. MongoDB Compass

A **MongoDB Compass** ³ a MongoDB grafikus felülete, amely egy letölthető webes alkalmazás. Segítségével felcsatlakozhatunk az felhőben lévő MongoDB Atlas-hoz és meg tudjuk tekinteni adatainkat, módosíthatjuk őket, indexeket hozhatunk létre, elemezhetjük az adatainkat és az adatforgalmakat. Lehetőség van emellett a MongoDB *aggregation* csővezetékek létrehozására, azokon belül pedig kódírással és tesztelésre. Az adatok módosítása és csővezetékek által biztosított tesztelési lehetőség nagyban megkönnyítette a szerver oldali implementációját a különböző lekérdezéseknek.

5.4. Postman

A **Postman** ⁴ egy kollaborációs platform API fejlesztők számára. Egy olyan komplett eszköztárral rendelkezik, amely által gyorsan és hatékonyan lehet dolgozni szerverünk API-val, mivel támogatja a fejlesztők minden munkafolyamatát. A projekt fejlesztése során segített a erőforrások helyes működésének a tesztelésében.

³MongoDB Compass Hivatalos dokumentációja: <https://docs.mongodb.com/compass/current/> (Látogatva: 2020.06.07)

⁴Postman Hivatalos dokumentációja: <https://learning.postman.com/docs/getting-started/introduction/> (Látogatva: 2020.06.07)

6. fejezet

Megvalósítási részletek

Összefoglaló: A projekt egy **MERN**-stacken alapul, azaz

- **MongoDB** - dokumentum-orientált adatbázis
- **Express** - a Node.js webes keretrendszere
- **React** - a kliens oldali könyvtár
- **Node.js** - a szerver oldali webszerver

A **MERN**-stack ([10]) az egymást jól kiegészítő, kompatibilis technológiákat és keretrendszereket ötvözi. Ez által egy gyors és rugalmas alkalmazás készítésére van lehetőségünk, amelynek megvalósítási részletei az alábbi fejezetben kerülnek bemutatásra.

6.1. Szerver oldali megvalósítás

6.1.1. Node.js és Express szerver megvalósításai

A Node.js Express szerver kialakításakor a cél egy biztonságos **Rest API** ([9]) megírása volt, amely interface-ként szolgál a webes kliensnek. Ezáltal a szerveroldal több rétegre lett osztva.

6.1.2. Útválasztó réteg

Az alábbi réteg felelős a beérkező HTTP kérések leosztásáért. Minden erőforrásnak saját útválasztó rétege van. A réteg feladata, hogy a hívást az erőforrásnak megfelelő controllerhez ossza ki, emellett az autentikációs middleware meghívásáért is felelős.

```
1 import express from 'express'
import auth from '../middleware/auth.js'
import {findProductById, deleteProdById} from '../controllers/product.js'

const router = express.Router();

6 router.get('/:productId', findProductById)
router.delete('/:productId', auth, deleteProdById)
```

Listing 6.1. Kódrészlet egy termék útválasztó rétegéből

6.1.3. Szolgáltatási réteg

A szolgáltatási réteget a kontrollerek összessége alkotja. Mindegyik erőforrás rendelkezik saját szolgáltatási réteggel és a különböző kéréseknek egyedi controllerük van. A szolgáltatási réteg általános célja egy kérés továbbítása az adatelérési réteg felé, majd az adatelérési üzleti logika végeredményének ellenőrzése és a beérkező kérésre válasz küldése, annak végeredményével.

Egy controller működése függhet a kérés entitásától, ugyanis a **Moongose** által kezelt entitásaink esetében az üzleti logika és az adatelérés a controllerünkben valósul meg, amiatt, mert a Moongose burkoló függvényei ezt leegyszerűsítik, ilyenkor az adatelérési réteg a Mongoosé sémáinkat tartalmazza. Abban az esetben, ha az entitásunkkal komplexebb műveleteket kell végezzünk és a **Node.js MongoDB Driverét** használjuk, ekkor a controller nem lát el üzleti logikát, csupán továbbítja az adatokat az adatelérési réteg felé.

```

2 export const findProductById = (req, res) => {
    let productId = req.params.productId
    const result = getProductById(productId);
    result.then((product) => {
        return res.status(200).json({ success: true, product });
    })
7    .catch((err) => {
        return res.status(404).json({ success: false });
    })
    }

```

Listing 6.2. Kódrészlet egy termék szolgáltatási rétegből

6.1.4. Adatelérési réteg

Az adatelérési réteg által kapcsolatba tudunk lépni az adatbázisunkkal és végrehajthatjuk a kéréseknek megfelelő műveleteket. Az adatelérési réteg Mongoosé esetében az objektumunk séma reprezentációját tartalmazza, míg a MongoDB Driver által történő lekérdezésekben az üzleti logikát és az adatelérési műveleteket valósítja meg.

```

export const getProductById = async (productId) => {
    if (mongodb.ObjectId.isValid(productId)) {
        const db = getDB()
        const product = db.collection('products')
        const productTransformedId = ObjectId(productId)
        let result = await product.findOne({ '_id': productTransformedId })
        return result
    } else {
10    throw "Illegal Argument Format"
    }
}

```

Listing 6.3. Példa egy termék lekérdezésére azonosítója alapján

6. FEJEZET: MEGVALÓSÍTÁSI RÉSZLETEK

```
import mongoose from "mongoose";

const notificationSchema = mongoose.Schema({
  4   userId: { type:String},
      Message: { type: Array},
});

export default mongoose.model("Notification", notificationSchema);
```

Listing 6.4. Példa az Értesítések sémára

6.1.5. A Socket.IO szerver megvalósítása

A Socket.IO szerver által valós idejű WebSocket kommunikációt biztosítunk a kliens oldalunkkal. Ezt egy külön modulba hoztuk létre, ami által egységes helyen kezelhetjük a beérkező kapcsolatokat. A Socket.IO szerver egy Node.js standard HTTP szerverre épül, ahol megadjuk, hogy milyen címről várjuk a különböző eseményeket.

Ezt követően a Socket.IO szerver várja a beérkező websocket kapcsolatokat. Egy sikeres websocket kapcsolat esetén elmentjük a socket azonosítóját és a kliens azonosítóját, akihez a socket tartozik, majd figyelünk a különböző eseményekre, amit felcsatlakozott kliens socketje küldhet és erre reagálunk.

- Beérkező események lehetnek: SEND_OFFER, SEND_BUYOFFER, ADD_USER, SENT_MESSAGE, NOTIFY_OTHER_CLIENT, DELETE_USER, MODIFIED_ALL
- Kimenő események: NOTIFICATION, RENDER_OFFERS, RECIEVED_RESPONSE, RECIEVED_PRODUCTOFFER, REFRESH_SHIPMENTSTATUS

```
socket.on('NOTIFY_OTHER_CLIENT', (otherClientId, notMessage,
  2   typeOfNotifictaion = '') => {
      const offerBasedActions = ['RECIEVED_PRODUCTOFFER', '
          RECIEVED_REQUESTOFFER', 'MODIFIED_ALL']
      const recieverSocket = onlineSockets.find(sockets => sockets.userid ===
          otherClientId)
      if (typeof recieverSocket !== 'undefined') {
          io.to(recieverSocket.socketid).emit('NOTIFICATION', notMessage)
          if (typeOfNotifictaion.length > 0) {
              7   if (offerBasedActions.includes(typeOfNotifictaion)) {
                  io.to(recieverSocket.socketid).emit('RENDER_OFFERS', { type:
                      typeOfNotifictaion })
                  } else {
                      io.to(recieverSocket.socketid).emit('REFRESH_SHIPMENTSTATUS')
                  }
              }
              12  } else { addNotificationByUserId(otherClientId, notMessage) }
          })
```

Listing 6.5. Példa egy felhasználó értesítésére különböző típusú műveleteknek megfelelően

A beérkező események függvényében lehetőségünk van a kérést intéző kliensnek válaszolni, az érintett partnert értesíteni vagy az összes felhasználó számára egy komponenst vagy al-komponentst frissíteni valós időben. Ha egy olyan felhasználót akarnánk értesíteni akinek nem elérhető a socketje, mert offline, a Socket.IO szerver kapcsolatban van az Express szerver Szolgáltatási rétegének kontrollereivel, ami által elmenthetünk értesítéseket és egy felhasználó bejelentkezésekor megjelenítjük azokat.

6.2. Kliens oldali megvalósítás

A kliens oldali megvalósítás során az architekturális rétegek a háttérbe szorulnak. Bár a React bármilyen architekturális kialakítást megenged, mégis inkább a lokális megjelenítési és üzleti logika ötvözetére sarkalja a fejlesztőket. Ennek következményeképp, a kliens oldal funkcionalitás szerint csoportosított könyvtárakra osztozik.

6.2.1. Egységek

A kliens oldal három fő csoportba osztható:

1. **Adapterek:** Az *api* és *service* könyvtárakat foglalja magába. Ezek valósítják meg a HTTP és WebSocket kommunikációt a kliens és szerver oldal között
2. **React Komponensek:** A *components* könyvtárat foglalja magába, az oldal felületének a megjelenítésért és az üzleti logikáért felelős
3. **Redux Komponensek:** Az *actions*, *constants*, *reducers* könyvtárakat foglalja magába, segítségükkel átláthatóan tudjuk a Redux centralizált állapotait kezelni

Adapterek

A **Socket.IO kliens** egyszerű beállítása a *service* könyvtár *socket.js* fájlban történik. Itt beállítva a Socket.IO szerverünk útvonalát kiexportáljuk a példányosított kliens socketünket, amit aztán a komponenseink kezelni tudnak.

```
1 import io from "socket.io-client";  
  const SOCKET_URL='localhost:5001'  
  
  export const socket = io(SOCKET_URL);
```

Listing 6.6. Socket.IO Client példányosítása

A HTTP kérések küldése az *api* könyvtár *index.js* állományában valósul meg. Itt vannak az erőforrások szerint csoportosítva az összes Rest API elérések. Az **Axiost**, ami a kérések küldésért felel példányosítjuk, és minden egyes kéréshez egy kiexportálható függvényt társítunk.

6. FEJEZET: MEGVALÓSÍTÁSI RÉSZLETEK

Ezáltal a komponenseinkből ezeket könnyedén meg tudjuk hívni és a visszakapott Promise-t egyszerűen le tudjuk kezelni. Az Axios lehetőséget biztosít az autorizációs fejléc beállítására minden kérés esetében, ami által elküldhetjük a JSON Web Tokenünket az ezt ellenőrző middleware-nak.

```
1 import axios from 'axios';

const API = axios.create({ baseURL: 'http://localhost:5000' });

API.interceptors.request.use((req) => {
6   if (localStorage.getItem('profile')) {
      req.headers.authorization = `Bearer ${JSON.parse(localStorage.getItem
        ('profile')).token}`;
    }

    return req;
11 });

// További erőforrás eléréseket is tartalmaz
export const getProductById = (productId) => API.get(`/api/products/${
  productId}`)
```

Listing 6.7. Példa a HTTP kérések küldésére

React komponensek

Az összes React komponens, ami a megjelenítésért és az üzleti logikáért felelős, a *components* könyvtárban található. Mindegyik komponensnek saját könyvtára van, ha egy fő komponens több kisebb komponens alkotja, akkor azokat *Section* alkönyvtárakban tároljuk. A *utils* könyvtár tartalmazza azokat az újrafelhasználható komponenseket amelyek az oldalon több helyen is előfordulnak.

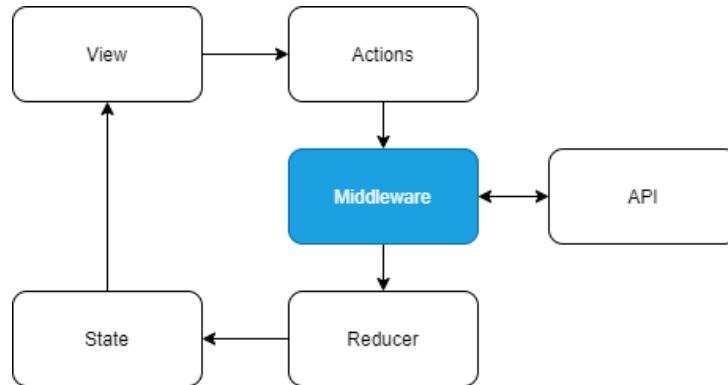
A React komponensek JavaScript függvények, amely saját állapotokkal és tulajdonságokkal rendelkeznek. Az állapotok a komponens saját állapotai, a tulajdonságok pedig más komponensek átadott állapotai, függvényei. A *return* metódus tartalmazza azokat a **JSX** HTML-szerű elemeket vagy függvényeket (*akár komponens is lehet*), amelyek az oldal felületi megjelenésért felelnek, ez hívódik meg rendereléskor. Ha egy komponens állapota változik, a komponens újra renderelődik és változik az oldal kinézete is anélkül, hogy azt újra kellene tölteni. A komponensünk tartalmazhat más függvényeket is, amelyek az üzleti logika megvalósításáért felelnek. Ezáltal az üzleti logika és megjelenítés ötvözve lesz.

Redux komponensek

A Redux segítségével a React komponenseink állapotát tudjuk globálisan kezelni egy *store* által, anélkül hogy a komponensek az állapotaikat propsként (*tulajdonságként*) kellene átadniuk a többi komponensnek. Mivel az oldalon a különböző felugró figyelmeztető jelzések (*popupok*)

6. FEJEZET: MEGVALÓSÍTÁSI RÉSZLETEK

és a regisztráció is használja a Reduxot, ezért a Redux munkafolyamatának fő elemeit *actions*, *reducers* és *constants* könyvtárakba rendeztük.



6.1. ábra. A Redux munkafolyamatai

- Actions könyvtár: tartalmazza azokat a komponens függvényeket, amelyeket *dispatch()* függvény által meghívva adatot küldhetünk a storenak
- Constants könyvtár: minden actionnek van egy *típusa*, ami egy név. Azért, hogy ezeket számon tartsuk és egy actiont biztonságosan tudjunk meghívni az összes típusnevet itt deklaráltuk.
- Reducers könyvtár: olyan komponens függvényeket tartalmaz amelyekkel a globális store aktuális állapotát módosíthatjuk

```
1 import { SET_SNACKBAR } from '../constants/actionTypes';  
  
export const setSnackbar = ( snackbarOpen, snackbarType = "success",  
  snackbarMessage = "" ) => (  
  {  
    type: SET_SNACKBAR,  
6    snackbarOpen,  
    snackbarType,  
    snackbarMessage  
  } );
```

Listing 6.8. Egy Popup jelzés Redux Actionje

6.2.2. Hitelesítés

A weboldalon a hitelesítés JSON Web Tokenek által történik. Egy felhasználó bejelentkezésekor, a felhasználóneve és azonosítója által generálunk neki egy tokent, amit a localStorageában tárolunk. A localStorageban kulcs-érték párokat tárolhatunk a memóriában és ezáltal, ha a felhasználó be is zárja az oldalt a tokenje megmarad. Egy token lejártával töröljük azt a felhasználó localStorageából és kijelentkeztetjük az illetőt.

6.3. Az alkalmazás adatmodellje

Az oldal a teljes testreszabhatóság jellemzi, ezáltal a dokumentum orientált adatbázisokban használható félig strukturált adatmodell kiváló opciónak bizonyult a folyamatosan változó adatok és értékek tárolásában. Az adatbázisban több kollekcióban vannak tárolva az adataink: *categories*, *users*, *products*, *notifications*, *offers*, *requests* és *soldproducts*. Ezek a kollekciók alább kerülnek bemutatásra.

Az oldal által előre meghatározott összes alkategória és az hozzájuk kapcsolódó kategóriák és értékek *categories* kollekcióban tárolódnak. Egy dokumentum tartalmazza attribútumként a főkategória nevét, az alkategória nevét, illetve további két beágyazott objektumot a *ProvidedCategories* és a *UserAddedCategories*. A *ProvidedCategories* felel az előre meghatározott kategóriák és értékek tárolásában, a *UserAddedCategories* pedig a felhasználók által hozzáadott, saját kategóriákat tárolja.

A *users* kollekcióban tároljuk egy felhasználó profiljának adatait. Minden egyes dokumentum attribútumként tartalmazza a felhasználónevet, jelszavat (titkosítva), illetve azt, hogy kibővítette-e már a felhasználó a profilját, és abban az esetben, ha igen olyan személyes adatokat amelyeket a profil kibővítéskor adott meg.

A *products* kollekció tartalmazza a marketplacen lévő összes eladó terméket. Egy dokumentum a termék adatlapjának létrehozásakor került beszúrára és azon kategóriák neveit és értékeit tartalmazza, amiket a felhasználó a létrehozás során megadott. Ezen kívül még szerepel a hozzáadott képek útvonalai egy asszociatív tömbben, a termék hozzáadás dátuma, a felhasználó neve és azonosítója aki hozzáadta a terméket.

A *notifications* kollekcióban tárolódnak azok az értesítések, amiket egy adott felhasználó még nem kapott meg, mert offline volt. Egy dokumentum tartalmazza a felhasználó nevét és egy asszociatív tömböt, amiben az összes olvasatlan értesítés tárolódik beágyazott objektumok formájában. Miután egy felhasználó megtekintette az értesítéseit törlődik a dokumentuma.

Az *offers* kollekcióban a felhasználók által küldött termék árajánlatok, vagy kérelmekre küldött termékek ajánlatai tárolódnak. Egy dokumentum attribútumai tartalmazzák az ajánlat típusát, ha kérelemre érkező ajánlat annak azonosítóját, illetve a termék, a küldő, és a fogadó azonosítóját. Ezen kívül az ajánlat értékét és a hozzáadás dátumát. Az oldalon lehetőség van minden ajánlat esetében chatelni ezáltal a két partner közötti beszélgetések is itt vannak beágyazva egy *Messages* asszociatív tömbbe.

A felhasználók által létrehozott kérelmek a *requests* kollekcióban találhatók. Mezői azok a kategóriák nevei, amikre a felhasználó kritériumokat szabott és az értékei típusa asszociatív tömbök, amelyek tartalmazzák a kiválasztott kritérium értékeit. Ezen kívül tartalmaz egy *userInfo* attribútumot, ez egy beágyazott objektum, ahol azon felhasználók nevei és azonosítói tárolódnak, akik ugyan azokat a kritériumokat szabták meg. Ha egy felhasználó törli az általa

6. FEJEZET: MEGVALÓSÍTÁSI RÉSZLETEK

létrehozott kérelmet és a userInfo tömbbe csak ő szerepelt, akkor a dokumentum is törlődik, ha nem egyedülként szerepelt, csak kitörlődik az asszociatív tömbből.

```
_id: ObjectId("60b62cb56873e5a25535bc8a")
  Category: Array
    0: "Cars"
  City: Array
    0: "Cluj Napoca"
    1: "Zalau"
  Condition: Array
    0: "New"
  Exact Model: Array
    0: "40"
    1: "E46 Sport"
  Price: Array
    0: 1
    1: 131186
  userInfo: Array
    0: Object
      userId: "6060d3cd8231610848942f69"
      userName: "teszt2"

_id: ObjectId("60b409e40b15fe082e26623d")
  Category: Array
    0: "Houses"
  Price: Array
    0: 1
    1: 58832
  userInfo: Array
    0: Object
      userId: "60b102562b01952b28a400a7"
      userName: "teszt1"
```

6.2. ábra. A requests kollekció két dokumentuma

A félig strukturált adatmodell által nyújtott lehetőségeket szemlélteti a 6.2. ábra is, ahol a requests kollekció két különböző dokumentumán keresztül mutatjuk meg azt, hogy a kérelmek mennyire különböznek egymástól.

Ha egy termék eladásra kerül, akkor az átkerül az *products* kollekcióból a *soldproducts* kollekcióba. Ezen kollekció dokumentumai tartalmazzák a termék adataalapjára vonatkozó attribútumokat, kiegészítve még olyan adatokkal, mint a végső eladási ár, eladási dátum, szállítási állapotokra vonatkozó mezők, az ajánlat azonosítója, ami által el lett fogadva a termék. Emellett beágyazva tartalmazza az eladó és vásárló adatait is.

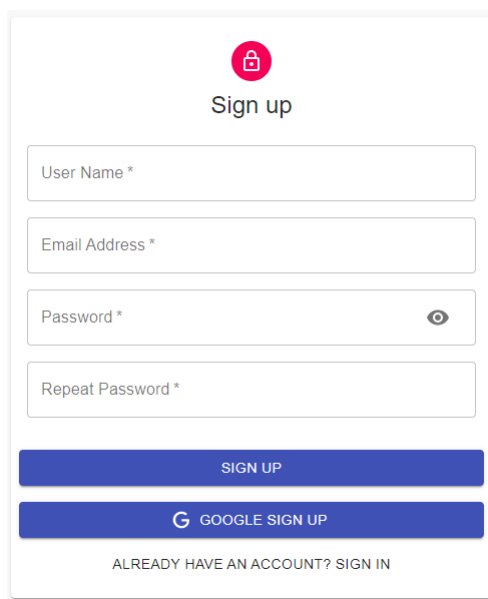
7. fejezet

The Marketplace projekt működése

7.1. Regisztráció, Bejelentkezés és Profil információk kibővítése

Regisztráció

A felhasználóknak lehetőségük van az oldal szinte összes funkciójának a használatára regisztráció nélkül is. Azonban, ha el szeretne adni terméket vagy ajánlatot szeretne küldeni más felhasználók felé abban az esetben szükséges regisztrálnia.



The image shows a registration form titled "Sign up" with a red padlock icon. It contains four input fields: "User Name *", "Email Address *", "Password *" (with a toggle eye icon), and "Repeat Password *". Below the fields are two blue buttons: "SIGN UP" and "GOOGLE SIGN UP" (with a Google 'G' logo). At the bottom, there is a link that says "ALREADY HAVE AN ACCOUNT? SIGN IN".

7.1. ábra. Regisztráció

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

Regisztráció kétféle módon valósulhat meg:

1. Felhasználónév, email cím és jelszó megadásával: A minimális követelmény az, hogy a felhasználónévnek egyedinek kell lennie
2. Google fiókkal: Az oldal támogatja a Google fiókkal történő regisztrációt, így a felhasználók gyorsan és egyszerűen tudnak regisztrálni

Sikeres regisztráció után az oldal létrehoz a felhasználónak egy profilt, és a felhasználót automatikusan bejelentkezteti.

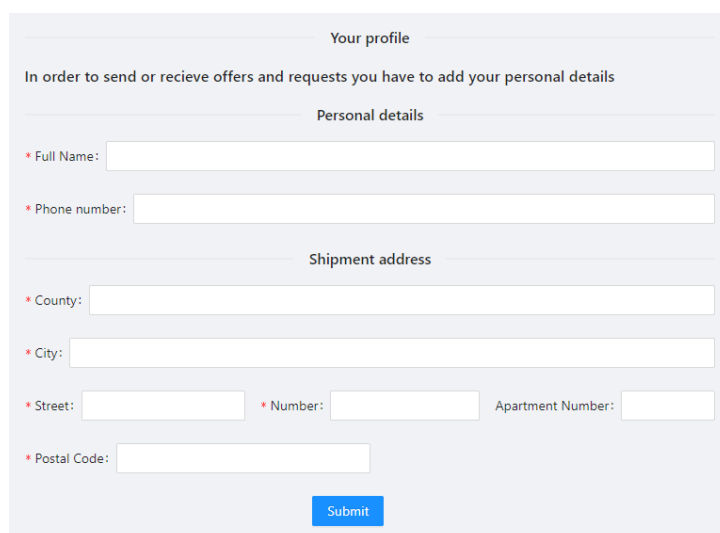
Bejelentkezés

A bejelentkezéshez szükséges megadni a regisztrált felhasználónevünket és jelszavunkat vagy használhatjuk a Google fiókunkat is. Miután egy felhasználó sikeresen bejelentkezett egy JWT generálódik számára, aminek egy órás lejárat ideje van.

Ennek biztonsági szerepe van, ugyanis a token lejártakor automatikusan kijelentkeztetjük a felhasználót így elkerülve az illetéktelen hozzáférést.

Profil információk kibővítése

Sikeres regisztráció és bejelentkezés után a felhasználók ahhoz hogy vásárolni vagy eladni tudjanak, még ki kell bővíteniük a profiljukat olyan személyes adatokkal, mint: teljes név, telefonszám, megye, város, utca, házszám, lakás szám (*opcionális*), postakód.



The screenshot shows a web form titled "Your profile". Below the title is a message: "In order to send or receive offers and requests you have to add your personal details". The form is divided into two sections: "Personal details" and "Shipment address".

Personal details section:

- * Full Name: [text input field]
- * Phone number: [text input field]

Shipment address section:

- * County: [text input field]
- * City: [text input field]
- * Street: [text input field] * Number: [text input field] Apartment Number: [text input field]
- * Postal Code: [text input field]

At the bottom of the form is a blue "Submit" button.

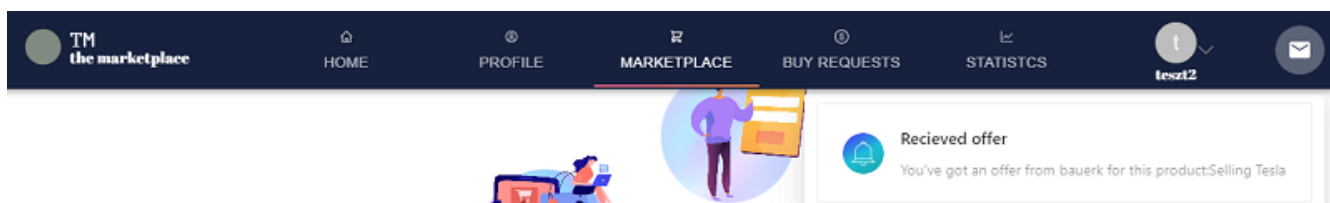
7.2. ábra. Profil információk kibővítése

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

A profil információk kibővítése egy egyszeri alkalom és a későbbiek során bármelyik megadott adat módosítható. Célja az, hogy sikeres eladás vagy vásárlás esetén a személyes adatokat azonnal fogja látni a felhasználónk partnere, ezzel is megkönnyítve a felek közti adatcserét, így kényelmesebbé téve az üzletkötést.

7.2. Navigáció

Az oldal tetején található egy navigációs menü ahol a fő komponenseket tudjuk elérni, mint: Home, Profile, Marketplace, Buy Requests, Statistics. Ezen kívül a bejelentkezett felhasználóknak lehetőségük van az online vagy offline kapott értesítések megtekintésére és visszanézésére.

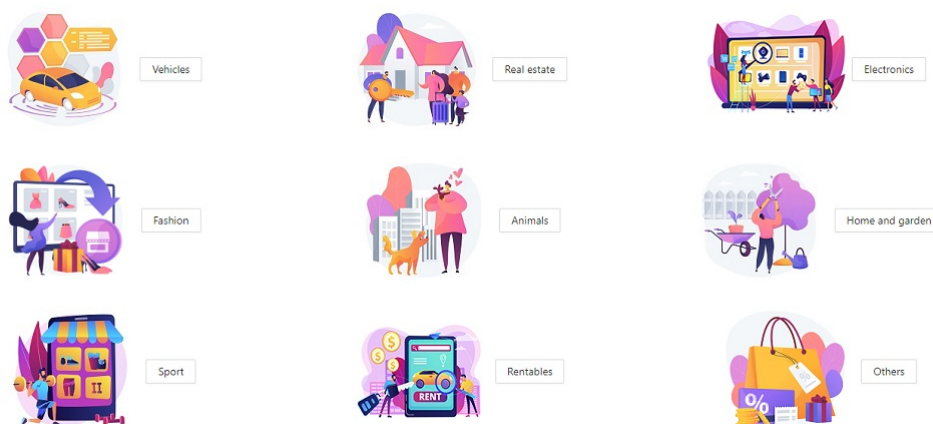


7.3. ábra. Navigációs menü

Egy értesítésre kattintva az oldal automatikusan átirányít arra az oldalra ahonnan az értesítés származik. A navigációs menüből tud a felhasználó kijelentkezni a profiljából.

7.3. Termék hozzáadása

Az oldalon kilenc fő termék kategória létezik: Járművek, Ingatlanok, Elektronikus eszközök, Divat, Állatok, Otthon és kert, Sport, Bérelhető és Egyéb kategória.



7.4. ábra. Termék főkategóriák

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

Mindegyik termékkategória rendelkezik a neki megfelelő alkategóriákkal, ezekből összesen negyvenegy darab van. Például Járművek esetében az alábbi alkategóriák léteznek: Autók, Motorbiciklik, Kamionok, Furgonok, Emberi meghajtású járművek. Az alkategóriától függően megjelennek az arra jellemző, az oldal által előre biztosított további kategóriák és azok értékei. Ezen kategóriákból összesen több mint négyszáz van és az előre meghatározott értékekből több, mint ötszáz, mindegyik egyedi és releváns értékekkel rendelkezik. Például az Autók alkategóriára jellemző Lóerő kategória nem fog megjelenni a PC és Laptopok kategóriái között.

Egy felhasználónak be kell sorolnia a termékét főkategóriába és alkategóriába, ha pedig egyikre se illeszkedik, akkor választhatja az Egyéb kategóriát és teljesen testreszabhatja termékének adatlapját.

Create your product

* Title:

* Type:

—

* Brand:

—

* Model family:

—

* Exact Model:

—

* Production age:

—

* Fuel Type:

—

* Color:

—

* Mileage(km):

—

* Horsepower:

—

* Condition:

* City:

* Price:

* Price Type:

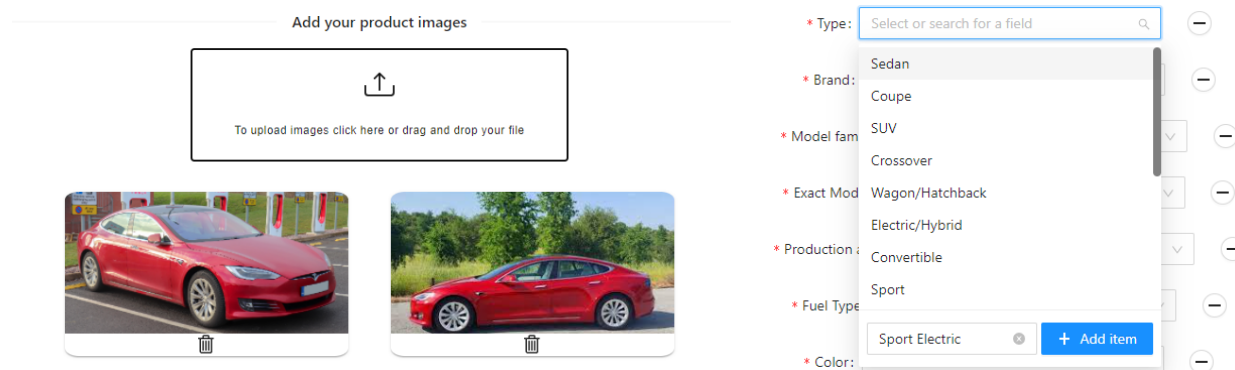
7.5. ábra. Autó alkategória előre meghatározott kategóriái

Miután a felhasználó kiválasztotta a termékének megfelelő fő-alkategóriát, megjelenik a termék létrehozásának adatlapja, ami tartalmazza az alkategóriához hozzárendelt kategóriákat és azok releváns értékeit, lásd 7.5. ábra. Egy adatlapon minden esetben meg kell adni a termék címét, helyét, árát és ár típusát.

Ezen kategóriákon kívül, a felhasználó teljesen szabad kezet kap az adatlap létrehozásában: hozzáadhatja termékeinek a képeit és törölheti azokat, kereshet az előre meghatározott kategó-

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

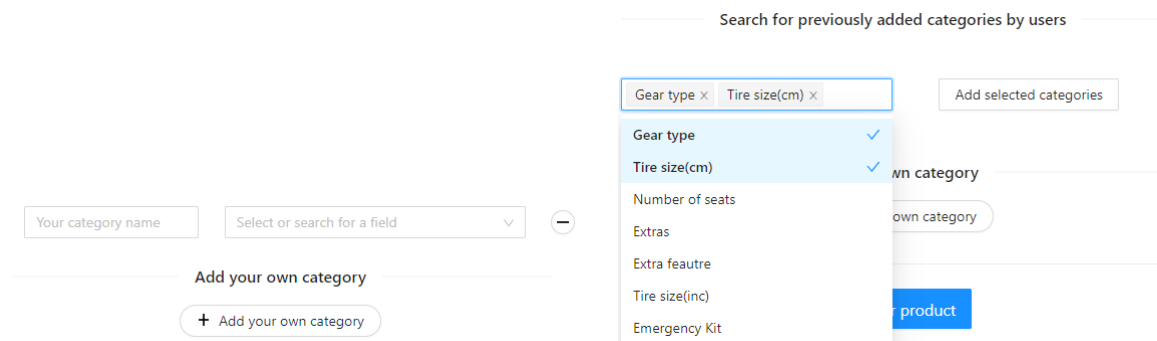
riák értékei között, hozzáadhat saját értéket, ha nem létezik ilyen érték az adott kategóriában. Kitörölhet előre definiált kategóriákat, de ha meggondolta magát vissza is tudja azokat helyezni az adatlapra.



7.6. ábra. Példa egy termék képeinek hozzáadására, illetve egy saját érték hozzáadására egy már előre definiált kategóriához

Egyes kategóriák között függőség van. Ez azt jelenti, hogy ha a függő kategóriát töröljük a függőségei is automatikusan törlődnek, fordítva ez nem igaz. Egy fontos aspektus, hogy ha függő kategória értéke változik, akkor annak megfelelően a függőségek értékei változnak. Ez megkönnyíti a termék létrehozást és átláthatóbbá teszi azt. Egy egyszerű példa erre egy Autó alkategória esetén a Brand és a Model Family közötti függőségek. A Brand az autó márkáját tartalmazza az értékei között szerepel például a BMW, Opel. A Model family értékei aszerint változnak, hogy mi melyik Brand értéket választottuk ki, például BMW esetében a Model Family értékei M3, M4, M5 lesz, Opel esetén Astra, Corsa, Kadett, Combo.

Egy felhasználó ezen kívül hozzáadhat általa meghatározott saját kategóriát és annak saját értéket, abban az esetben hogyha az előre meghatározott kategóriák egyike sem illeszkedik a termékére. Kereshet más felhasználók által már hozzáadott saját kategóriák között is, és kiválaszthatja őket terméke adatlapjához.



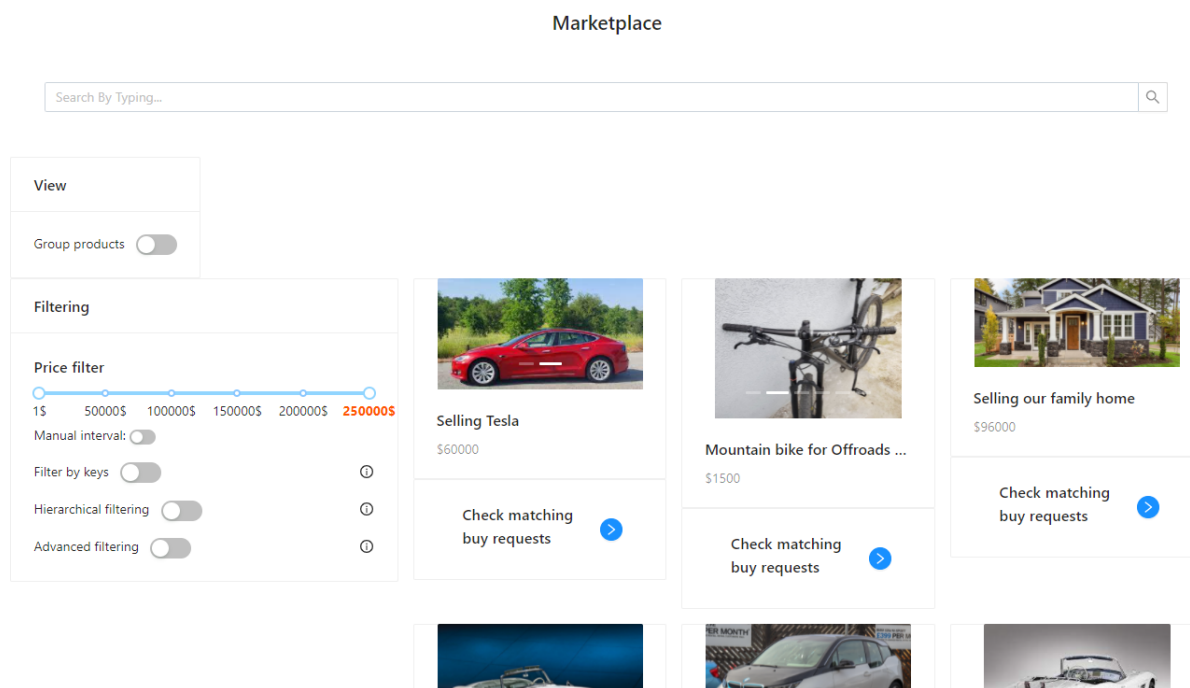
7.7. ábra. Példa saját kategóriák hozzáadására, illetve más felhasználók által hozzáadott saját kategóriák újrafelhasználására

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

Egy termékhez öt saját kategóriát adhatunk hozzá általánosan, ez alól kivétel az Egyéb kategóriák ahol tízet. Egy kategóriához akárhány értéket hozzáadhatunk. Ha a felhasználó befejezte a termék adatlapjának a létrehozását és az adatlapban van olyan hozzáadott érték vagy kategória, amik termék létrehozás során eddig nem szerepeltek az adatbázisban, akkor ezeket releváns adatokként kezeljük és elmentjük őket. Fontos szempont, hogy bár a felhasználó akárhány értéket hozzáadhat egy kategóriához, csak az fog legvégül bekerülni az adatbázisba is, ami termék létrehozása esetén is kiválasztva szerepel, ezzel elkerülve azt, hogy a felesleges értékek az adatbázisba kerüljenek. Az új adatok ezután elérhetőek lesznek a többi felhasználónak is, akik újrahasznosíthatják ezeket saját terméküknél. Ezzel egy naprakészebb adatbázist kapunk, kategóriákra lebontva releváns információkkal és egy gyorsabb és kényelmesebb felhasználói élményt biztosítunk.

7.4. Eladó termékek megtekintése és megjelenítési formái

A hozzáadott termékek összességét meg lehet tekinteni, ha kiválasztjuk a navigációban a Marketplace opciót. Ekkor megjelennek a mások által hozzáadott termékek, mindegyik egy-egy kártyában. Egy kártya tartalmazza a termék képeit, ezek másodpercenként folyton váltakoznak, a termék címét és árát.



7.8. ábra. Termékek főoldala

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE


A termékek listázására két opció van

1. Egyszerű nézet: Ilyenkor a termékek sorban kártyákként jelennek meg
2. Group nézet: Az azonos kategóriákkal és értékkel rendelkező termékeket csoportosítja, ezzel megkönnyítve az azonos termékek közötti keresést


Egyszerű nézet esetében a termék kártyájára kattintva meg tudjuk tekinteni a termék adatlapját. A check matching buy requestsre kattintva pedig a termékre illeszkedő vásárlási kérelmeket ¹ tudjuk táblázat formájában kilistázni.

Group nézet kiválasztása esetén a kártyán megjelenik a csoportosított termékek száma és azok a kategóriák és értékek, amik szerint csoportosítva lettek. A nézet esetében nem vesszük figyelembe az olyan adatokat, mint hogy mikor lett hozzáadva, ki által lett hozzáadva, a termék címét és az árának típusát.

Grouped 2 products



Grouped by
Price Type: Fixed Price
Type: Vintage
Brand: BMW
Model family: 1-5 series
Exact Model: 40
Production age: 1960
Fuel Type: Diesel
Color: White
Mileage(km): 15000
Condition: New
Price: 90000
Main Category: Vehicles
Category: Cars

Check products 

Grouped products																
Title	Price Type	Type	Brand	Model family	Exact Model	Production age	Fuel Type	Color	Mileage(km)	Condition	City	Price	Main Category	Category	Added on	Username
BMW B40 Vintage Masterpiece quicksell!	Fixed Price	Vintage	BMW	1-5 series	40	1960	Diesel	White	15000	New	Cluj Napoca	90000	Vehicles	Cars	2021-05-28	test2
Selling Vintage BMW	Fixed Price	Vintage	BMW	1-5 series	40	1960	Diesel	White	15000	New	Zalau	90000	Vehicles	Cars	2021-05-27	bauerk

7.9. ábra. Csoportosított termékek kártyája és táblázata

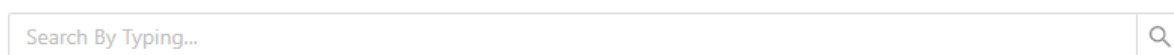
A kártya alján lévő Check products ikonra kattintva egy felugró ablakban megjeleníti a kártya értékeit táblázat formában, lásd jobboldali 7.9.-es ábrán. Itt lehetőséget biztosít arra, hogy mindegyik termék adatlapját meg tudjuk nézni.

¹A Check matching buy request funkcionalitás az *Egy termékre illeszkedő vásárlási kérelmek megtekintése* fejezetben van részletezve.

7.5. Termékek szűrése

Ahhoz hogy minél pontosabban megtaláljuk az általunk keresett terméket az oldal ötféle termékszűrési lehetőséget támogat, mindegyik opcionálisan használható és felhasználóbarát módon testreszabható.

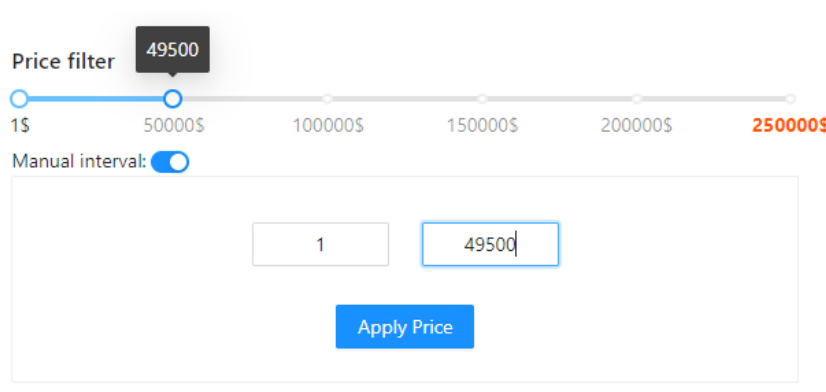
1. Cím szerinti termékszűrés



7.10. ábra. Cím szerinti termékszűrés

Egy search boxba beírhatjuk az általunk keresett terméket és minden olyan termék kilistázásra kerül, amelynek a címében szerepel a keresett szavunk.

2. Ár szerinti termékszűrés



The image shows a price filter interface. At the top, there's a 'Price filter' label and a value of '49500' in a dark box. Below this is a horizontal slider with a blue track and white dots. The slider has labels: '1\$', '50000\$', '100000\$', '150000\$', '200000\$', and '250000\$'. The '250000\$' label is in red. Below the slider, there's a 'Manual interval:' label with a toggle switch that is currently turned on. Underneath the toggle, there are two input fields: the first contains '1' and the second contains '49500'. Below these input fields is a blue button labeled 'Apply Price'.

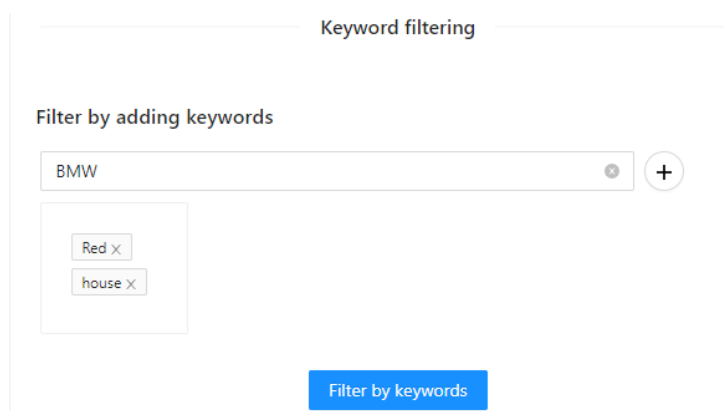
7.11. ábra. Árintervallum szerinti termékszűrés

A felhasználók szűrhetik a termékeket azok ára szerint. Ezt egy slider segítségével tehetik meg, ennek két végpontjának húzogatásával állítható az érdekelt ár intervalluma. Amikor a felhasználó a slider egyik végpontjára helyezi az egerét akkor megjelenik az ár, ahol a slider található. Ha pontosabban szeretné az árat beállítani bekapcsolhatja a *Manual interval* funkciót, ami által begépelheti mindkét végpont árát. Gépelés alatt slider végpontjai és ezáltal az intervalluma is automatikusan változik a begépelte ár függvényében, így ha a felhasználó későbbiek során úgy dönt, hogy a slidert szeretné használni nem kell beállítsa azt, mert az aktuális értékekre mutat.

A slider húzásának abbahagyása vagy a gépelés befejezésekor a termékek automatikusan szűrésre kerülnek. Az Apply Price gomb csak a Manual interval opció bekapcsolásakor jelenik meg azért, hogy egy biztonsági érzetet nyújtson, azon felhasználók számára akik esetleg nem vették észre, hogy a gépelésük alatt automatikusan frissültek a termékek.

3. Kulcsszavak szerinti termékszűrés

A kulcsszavak szerinti termékszűrés során, a felhasználó megadhatja az általa keresett kulcsszavakat.



7.12. ábra. Kulcsszavak szerinti termékszűrés

A kulcsszavakat szóközzel elválasztva kell megadni, majd a plusz gomb vagy enter billentyű lenyomásával egy listába helyezheti őket. A listából a listaelem mellett lévő kis *X* ikonnal törölheti. Miután előállította a keresési listáját, a *Filter by keywords* gomb lenyomásával szűrheti a termékeket. Minden olyan termék szűrésre kerül, amelynek vagy a kategóriáiban vagy az értékeiben, teljesen vagy részlegesen szerepel bármelyik a kulcsszavak közül. A kisbetű-nagybetű érzékenységet nem vesszük figyelembe, tehát bármilyen módon megadhatóak a kulcsszavak.

4. Hierarchikus termékszűrés

A hierarchikus termékszűrés célja egyetlen termékcsoporthoz minél pontosabb meghatározása azáltal, hogy automatikusan szűkíti a szűrési kategóriákat és azok értékeit aszerint, hogy a felhasználó milyen szűrési értékeket választ.

A hierarchikus szűréskor a felhasználónak először ki kell választania egy főkategóriát és egy alkategóriát. A főkategória kiválasztása függvényében módosulhat az ár szerinti szűrés intervalluma is, például amíg Járművek és Ingatlanok esetében kétszázötvenezerig tart az árintervallum, addig az Elektronikus eszközöknél tizenötezerig, ez a termékárak egyszerűbb és pontosabb behatárolását eredményezi.

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

The image displays two side-by-side screenshots of a web application's filtering interface, titled 'Hierarchical filtering'. Both screenshots show a 'Main Category' dropdown set to 'Vehicles' and a 'Sub Category' dropdown set to 'Cars'. Below these, there are radio buttons for various categories: Vehicles (selected), Real estate, Electronics, Fashion, Animals, Home and garden, Sport, Rentables, and Others. Under 'Sub Category', there are radio buttons for Cars (selected), Motorcycles, Trucks, Vans, and Human Powered. Below the hierarchical filters, there is an 'Additional filters' section with a 'Select additional filters' dropdown. This section contains a list of filters: Brand (with checkboxes for BMW and Tesla), Color (with checkboxes for Red and White), Condition (with checkboxes for New and Used), and Exact Model. The right screenshot shows the state after selecting 'Tesla' under the 'Brand' filter. In this state, the 'Exact Model' filter is no longer visible, and the 'Brand' filter now shows a checked box next to 'Tesla'. The 'Color', 'Condition', 'Fuel Type', 'Mileage(km)', and 'Model family' filters are still visible.

7.13. ábra. Hierarchikus termékszűrés

A fő és alkategória kiválasztása után megjelennek a *releváns kategóriák és értékek*², ezáltal biztosak lehetünk abban, hogy minden szűrési kategória és érték fellelhető a termékek között.

A hierarchikus termékszűrés abból adódik, hogy kiválasztva egy szűrési kategória értékét megtörténik a kritérium szerint a termékek megjelenítése, majd a szűrési kategóriák és értékei automatikusan frissülnek úgy, hogy azok már csak a kiválasztott kritérium szerinti elemekre jellemzőek. Ezt jól mutatja a 2.10. ábra, ahol kezdetben több kategória és érték is szerepel, amik a Járművek főkategória, Autók alkategóriájában fellelhetőek. Kiválasztva a Brandből a Tesla értéket a szűrési kategóriák és értékek is újratöltődnek, eltűnik az Exact Model kategória és több érték is, mivel azok már nem szerepelnek olyan termékekben ahol a Brand értéke a Tesla. Ennek eredményül pedig újfent csak a releváns szűrési lehetőségek jelennek meg. További szűrési értékek kiválasztásával a termékek és a szűrési lehetőségek egyre szűkülni fognak és az általunk keresett termék egyre pontosabban meghatározható lesz. Ha a felhasználó úgy dönt, hogy tágítani szeretné a szűrési kritériumokat vagy megváltoztatná egy szűrési kritérium értékét elég csak kipipálnia azt

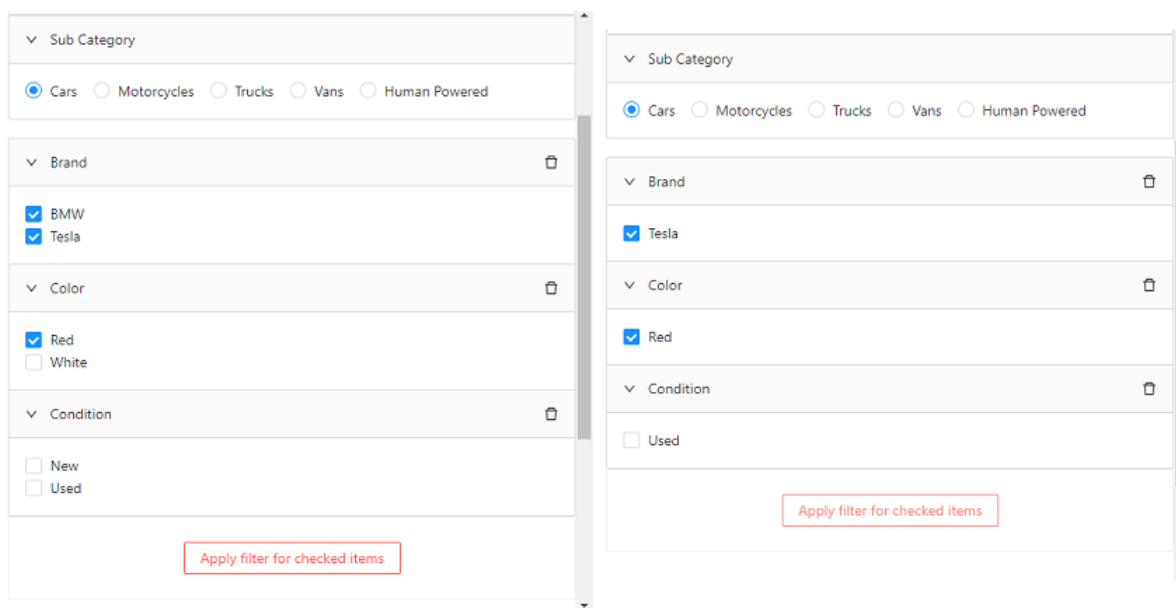
²Releváns kategóriák és értékek alatt azt értjük, hogy a szűrési kategóriák és azok értékei nem előre, az oldal által meghatározottak, hanem csak azok az értékek szerepelnek, amiket az eladó termékek is tartalmaznak

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

kritériumot, amely szerint nem akar szűrni, és újratöltődnek a szűrési lehetőségek.

5. Fejlett termékszűrés

A fejlett termékszűrés, a hierarchikus termékszűrést gondolja tovább úgy, hogy itt már lehetőség van nem csak egy termékcsoporthoz szűkítésre, hanem egyszerre többre is.



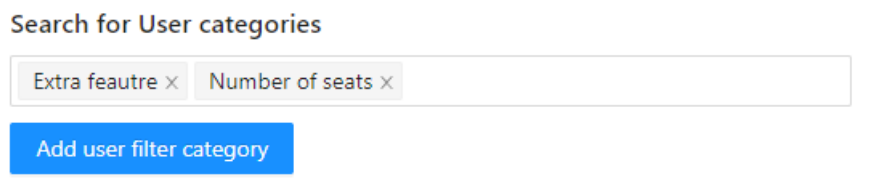
7.14. ábra. Fejlett termékszűrés

Működésileg hasonló a hierarchikus szűréshez, az alkategória kiválasztása után betöltődnek a releváns szűrési kategóriák az értékekkel. Ekkor egy szűrési kategórián belül több értéket is kiválaszthat a felhasználó, majd az *Apply filter for checked items* gombra kattintva elvégezheti a szűrést és megjelenítődnek a kritériumoknak megfelelő termékek. A szűrési kategóriák és értékek esetében is szintén a relevancia elvét követi azáltal, hogy az előzőleg beállított szűrési értékeket újra alakítja úgy, hogy azokból csak azt őrzi meg, amelyek az összes kritériumnak megfelelnek. Egy jó példa erre 2.14. ábra, ahol a felhasználó kiválasztotta Brandnek a BMW és Tesla értékeket és a Colornak a Red-et. Mivel a Color Red azokra a termékekre jellemző, ahol a Brandnek Tesla volt beállítva emiatt a szűrést követően a BMW eltűnt a Brandből és a Condition értékei is megváltoztak.

6. Közös szűrési alapelvek

- A kulcsalapú, a hierarchikus és a fejlett termékszűrés eredményeit még tovább lehet szűrni bármikor cím és ár szerint.
- Úgy a hierarchikus, mint a fejlett termékszűrés esetében, az elsődlegesen támogatott szűrési kategóriák az oldal által előre meghatározott kategóriák. Ám emellett a fel-

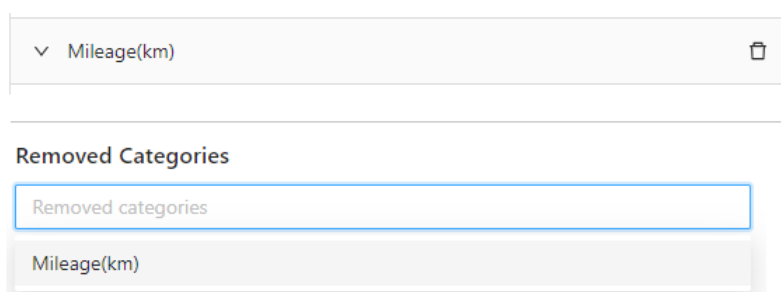
használóknak lehetőségük van a terméklétrehozáskor hozzáadott, saját kategóriák szerinti szűrésre is.



7.15. ábra. Saját kategóriák hozzáadása a szűrési opciókhoz

Ezeket egy lenti listában kereshetik meg kategórianév szerint vagy a legördülő listából kiválaszthatják őket, majd az *Add user filter category* gomb lenyomásával, ezek bekerülnek a filterezési kategóriák közé. Szűrés esetén úgy a lista ahonnan a saját kategóriákat adjuk hozzá, mint a hozzáadott saját kategóriák értékei is relevancia szerint dinamikusan változnak.

- A hierarchikus és fejlett termékszűrés esetében a szűrési kategóriák teljesen testreszabhatóak. Ezeket lehet törölni, ha a felhasználó a szűrési kategória mellett lévő Kuka ikonra kattint. Ekkor egy új szűrést hajt végre az oldal, és a szűrési kategóriák és értékek újragenerálódnak úgy, hogy már nem lesz figyelembe véve a kitörölt kategória.



7.16. ábra. Szűrési kategóriák törlése és újbóli hozzáadása

A kitörölt szűrési kategóriákat a felhasználók egy legördülő listában láthatják vagy kereshetik kategórianév szerint, és ezeket újból hozzáadhatják a szűrési kategóriákhoz bármikor. Ilyenkor az értékeik szintén relevánsak maradnak az aktuális szűrési állapot szerint.

7.6. Termék adatlapja

Egy termék adatlapján az információk három csoportba vannak osztva: Termék adatok, Termék részletek és Ajánlatok.

Product data

Data	Value
Title	Selling Tesla
Price Type	Negotiable Price
Condition	Used
City	Zalaú
Price	\$60000

1-5 of 7 < >

Details

Category	Value
Type	Electric/hybrid
Brand	Tesla
Model family	Model S
Production age	2016
Fuel Type	Electric

1-5 of 9 < >

Offers

User	Offer	Date
bauerk	\$60000	2021-05-04 20:22

1-1 of 1 < >

Send your offer

[MAKE AN OFFER](#)

7.17. ábra. Termék adatlap

- Termék adatok: A termékre jellemző főbb információk, mint például: Cím, Ár típus, Állapot, Város, Ár, Hozzáadás dátuma, Létrehozó felhasználóneve.
- Termék részletek: Az összes hozzáadott kategória és értékeik.
- Ajánlatok: A termékre érkező árajánlatok. A transzparencia érdekében szerepel az ajánlatot küldő felhasználóneve, az árajánlat és az árajánlat küldésének dátuma.

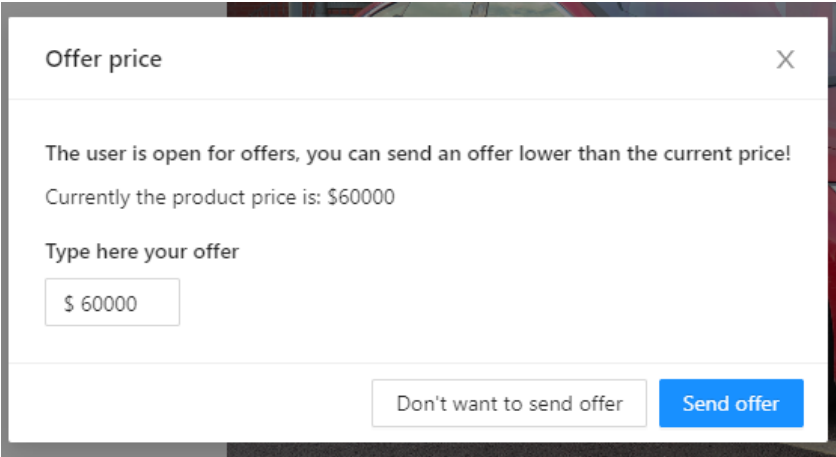
Ezen kívül a felhasználók a képen lévő nyilakkal lapozhatnak a hozzáadott képek között, a jobb alsó sarokban pedig megtekinthetik a termék képeit teljes nagyságukban.

7.7. Árajánlat küldése

Árajánlat küldéséhez a vásárlónak előzőleg ki kell bővítenie a profilját. Árajánlatot a *Make an offer* gombra kattintva tud elküldeni egy termék adatlapján. Ugyanaz a vásárló ugyanazt a pénzbeli árajánlatot csak egyszer küldheti el, ezzel is szűrve az eladó számára a beérkező felesleges ajánlatokat.

Az ár típusától függően kétféle ajánlat küldés lehetséges:

1. Fix ártípus, azaz Fixed Price esetén - A vásárló nem alkudhat az árból, ha az ajánlat gombra kattint akkor a termék árának megfelelő teljes összeget küldi el ajánlatként
2. Alkudható ártípus, azaz Negotiable Price vagy bármilyen más ártípus esetén - A vásárló alkudhat az árból ezáltal vagy a termék teljes összegét vagy az alatt küldhet ajánlatokat. Ebben az esetben egy ablak ugrik fel a vásárlónak ahol megadhatja az általa gondolt összeget.



7.18. ábra. Alkudható ártípus esetén ajánlat küldés

Miután egy ajánlat elküldésre került, a termék adatlapján lévő Ajánlatok táblázat frissül az új vásárló adataival. Az eladó értesítve lesz az új ajánlatokról és az új ajánlat bekerül úgy az eladó, mint a vásárló profiljához. Ezen komponensek valós időben frissülnek, így a többi felhasználó is azonnal látja mikor valaki ajánlatot tett, illetve az eladó is rögtön értesül az új ajánlatról.

7.8. Vásárlási kérelmek

A vásárlási kérelem által a vásárló jelezni tudja azon szándékát, hogy termékeket vásárolna, amik megegyeznek az általa meghatározott kritériumokkal. Ha az eladóknak van ilyen termékük elküldhetik a vásárló kérelmére azokat, aki később elfogadhatja vagy elutasíthatja a kérelmére érkező termékajánlatokat.

7.8.1. Vásárlási kérelmek létrehozása

Show buy requests ☐

Show relevant buy requests based on your products ☐

Create buy request ☒

Create your buy request

Select your buying price range

1\$

50000\$

100000\$

150000\$

200000\$

250000\$

Manual interval: ☐

Select the buy request type

Buy request type

☒ Buy request for any product
 ☐ Buy request only for available marketplace products

Select a Main Category and Sub Category

Main Category

☒ Vehicles
 ☐ Real estate
 ☐ Electronics
 ☐ Fashion
 ☐ Animals
 ☐ Home and garden
 ☐ Sport
 ☐ Rentables
 ☐ Others

Sub Category

☒ Cars
 ☐ Motorcycles
 ☐ Trucks
 ☐ Vans
 ☐ Human Powered

Select additional criteria

Select all options for which you want to create buy offer

> Brand

> City

> Color

> Condition

> Exact Model

> Fuel Type

> Horsepower

> Mileage(km)

> Model family

Search for User categories

Search here

Add user filter category

Add all selected values to buy request

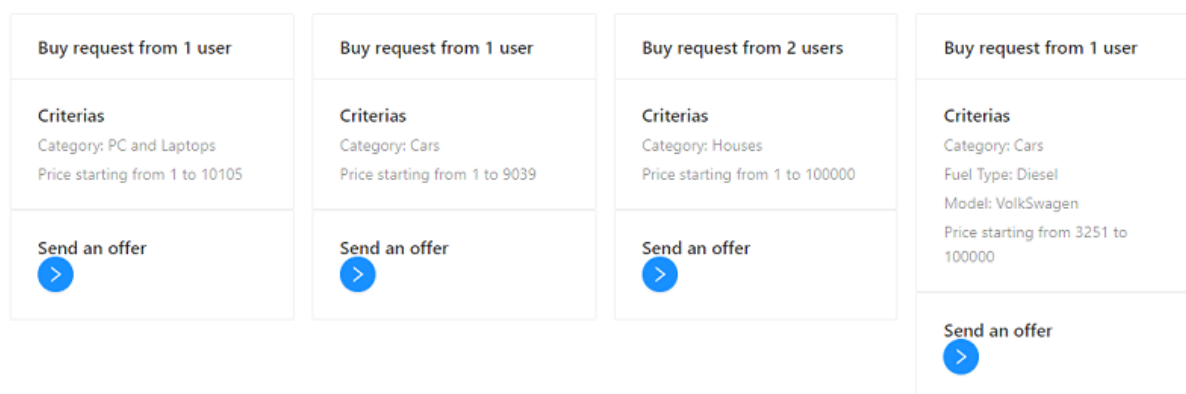
7.19. ábra. Vásárlási kérelem létrehozása

A vásárlási kérelemhez a Navigációs menüben a *Buy Request* gombra kell kattintson a felhasználó, ezután a *Create Buy Request* opciót engedélyeznie kell. Elsőnek a Buy request typeot, azaz a Vásárlási kérelem típusát kell megadnia. Ezáltal el tudja dönteni, hogy az összes kategória összes értéke érdekeli vagy csak azok a kategóriák és értékek fontosak számára, amik a jelenleg eladó termékekben is szerepelnek.

Ezt követően ki kell választania egy fő és alkategóriát, lásd 7.19. ábra. Be tud állítani ár-intervallumot, ami a kiválasztott kategória alapján dinamikusan változik. A Select additional criteria részben, hasonlóan a szűréshez, lenyíló menüben megjelennek a releváns kategóriák és értékeik, amik által opcionális kritériumokat tud a vásárlási kérelmének megszabni a felhasználó. Ezeket a kategóriákat a szűréshez hasonlóan lehet törölni. A hozzáadott saját kategóriák között is kereshet, amiket berakhat az addicionális kritériumok közé. Az *Add all selected values to buy request* gombra kattintva létre tudja hozni a vásárlási kérelmet.

7.8.2. Vásárlási kérelmek megtekintése

A felhasználók által létrehozott összes vásárlási kérelmet meg tudjuk tekinteni a *Show Buy Request* menü opció engedélyezésével. Ezek után a vásárlási kérelmek kártyák formájában betöltődnek és a Criteria cím alatt szerepelnek a felhasználók által beállított kritériumok.

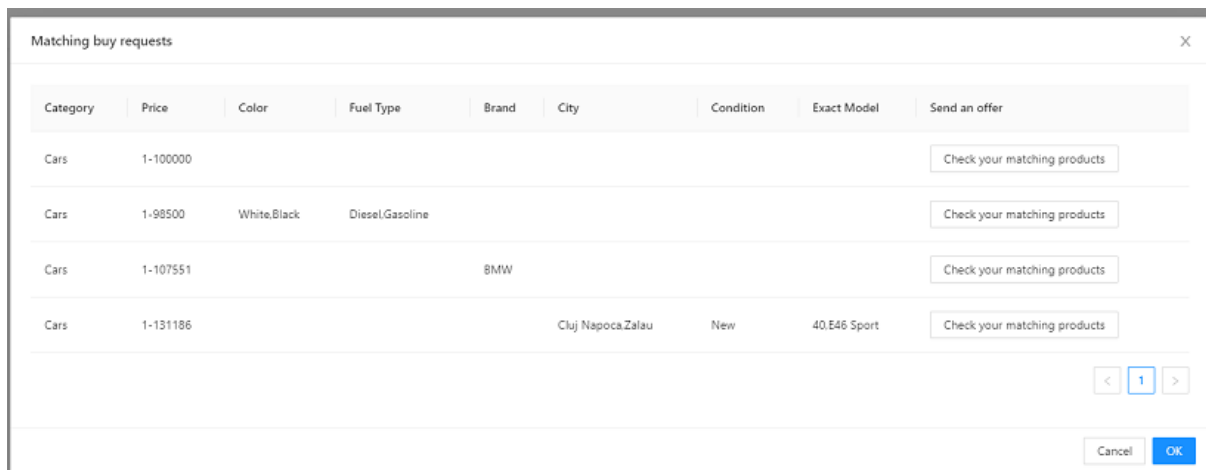


7.20. ábra. Vásárlási kérelmek megtekintése

Ha több felhasználó ugyan azokat a kritériumokat állította be egy vásárlási kérelemnek ezeket a felhasználókat csoportosítjuk. Ebben az esetben, amikor egy eladó elküldi egy csoportosított vásárlási kérelemre a termékét, az összes felhasználó számára elküldi ezt a terméket. A *Send an offer* ikonra kattintva megnézhetjük, hogy az adott vásárlási kérelemre van-e olyan eladó termékünk, ami megfelel a kritériumoknak, és abban az esetben, ha igen elküldhetjük rá termékünket.

7.8.3. Egy termékre illeszkedő vásárlási kérelmek megtekintése

Egy termékre illeszkedő vásárlási kérelmeket a *Check Matching Buy Request* opció engedélyezése által érjük el. Ekkor egy táblázat formájában listázva lesz az összes olyan vásárlási kérelem és kritérium, amire illeszkedik az adott termék. Egy vásárlási kritérium egy sora a táblázatnak, a kritériumkategóriák pedig az oszlopok.



Category	Price	Color	Fuel Type	Brand	City	Condition	Exact Model	Send an offer
Cars	1-100000							Check your matching products
Cars	1-98500	White,Black	Diesel,Gasoline					Check your matching products
Cars	1-107551			BMW				Check your matching products
Cars	1-131186				Cluj Napoca,Zalau	New	40.E46 Sport	Check your matching products

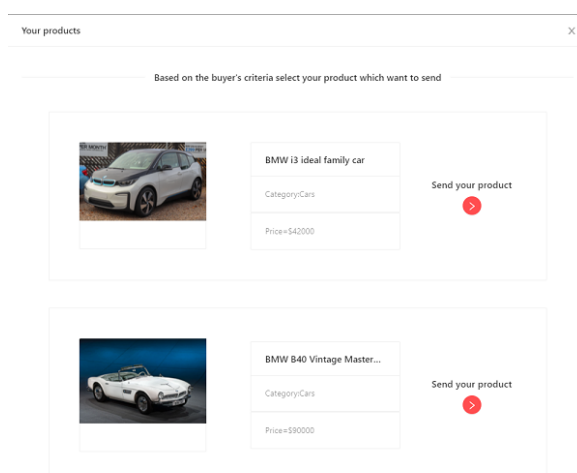
Navigation: < 1 > | Cancel OK

7.21. ábra. Egy termékre illeszkedő összes vásárlási kérelem táblázata


A *Check your matching products*-ra kattintva megnézhetjük, hogy a mi eladó termékein közül van-e olyan, ami illeszkedik az adott vásárlási kritériumra.

7.8.4. Termékünk elküldése egy vásárlási kérelemre

Amennyiben rákattintottunk a *Check your matching products* gombra vagy a *Send Offer*-re a vásárlási kérelmek megjelenítésekor egy ablak ugrik elő.



Based on the buyer's criteria select your product which want to send




BMW i3 ideal family car

Category: Cars

Price: \$42000

Send your product



BMW B40 Vintage Master...

Category: Cars

Price: \$90000

Send your product

7.22. ábra. Termékünk elküldése egy vásárlási kérelemre

Csak azokat az eladó termékeinket tartalmazza, amik megfelelnek a vásárlási kérelem összes kritériumának. A *Send Offer* gomb lenyomásával elküldhetjük a termékünket magunkon kívül összes olyan felhasználónak, aki a vásárlási kérelemben van, lásd 7.22. ábra.

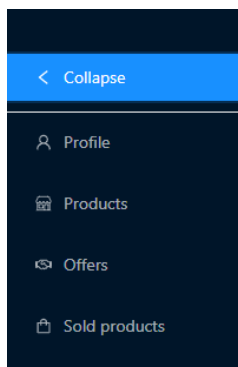
Az ajánlat elküldésekor az összes felhasználónál megjelenik a mi termékünk, és azonnal értesítve lesznek, hogy a vásárlási kérelmükre érkezett egy termék.

7.9. Profil

A profilunkat a navigációs menüre kattintva tudjuk elérni. Ezt négy fő komponens alkotja.

1. Profil komponens: A személyes adatok kibővítésére és módosítására van lehetőség.
2. Termékek komponens: A jelenleg eladó termékeinket itt tudjuk megtekinteni, szerkeszteni és törölni.
3. Ajánlatok komponens: Az összes küldött és kapott árajánlatot és vásárlási kérelmet kezelhetjük.
4. Eladott termékek komponens: Eladott és vásárolt termékeink itt vannak listázva.

A komponensek közötti navigációt egy összezsukható és kinyitható oldalnavigációs menü valósítja meg.



7.23. ábra. Profil komponensek közötti navigáció

7.9.1. Profil komponens

A profil komponensre kattintva, a profil kibővítése során megadott értékek jelennek meg. Ezeket a mellettük lévő ikonra kattintva akármikor módosíthatjuk



Your profile	
Username	teszt2 ✎
Your name	Nagy János ✎

7.24. ábra. Profilunk adatainak módosítása

7.9.2. Termékek komponens

A termékek komponensre kattintva meg tudjuk tekinteni a jelenleg eladó termékeink listáját.

Az *Edit* gombra kattintva módosíthatjuk termékeinket. Ilyenkor visszakerül a Termék létrehozása oldalra és az összes olyan kategória és érték ki lesz választva, amiket a termékünk jelenleg is tartalmaz. Ezután a testreszabási lehetőségek ugyan azok, mint a terméklétrehozás esetében is. A *Delete Product* gombra kattintva törölhetjük az eladó termékeinket. Ilyenkor minden ajánlat és kérelem, ami a termékre érkezett automatikusan törölve lesz, az érintett feleket pedig valós időben értesítjük erről.

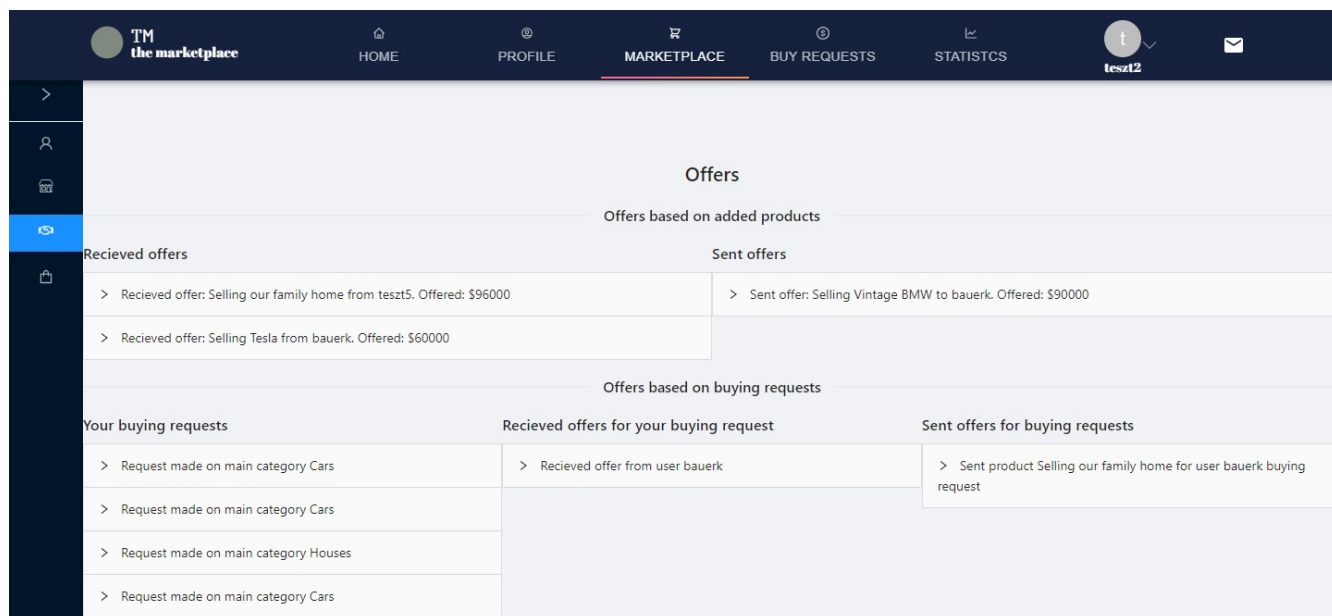
	<table><tr><td>Mountain bike for Offroads Pros!</td></tr><tr><td>Category: Human Powered</td></tr><tr><td>Price: \$1500</td></tr></table>	Mountain bike for Offroads Pros!	Category: Human Powered	Price: \$1500	Edit product Delete product
Mountain bike for Offroads Pros!					
Category: Human Powered					
Price: \$1500					
	<table><tr><td>Selling Tesla</td></tr><tr><td>Category: Cars</td></tr><tr><td>Price: \$60000</td></tr></table>	Selling Tesla	Category: Cars	Price: \$60000	Edit product Delete product
Selling Tesla					
Category: Cars					
Price: \$60000					

7.25. ábra. Termékek komponens

7.9.3. Ajánlatok komponens

Az ajánlatok komponens közös gyűjtőhelye a felhasználó árajánlat tételeinek és kérelmeinek. Az itt fellelhető elemek árajánlatok és kérelmek szerint csoportosítva vannak, illetve szeparálva attól függően hogy kapott vagy küldött. Minden elem kezdetben csak egy menüsávként jelenik meg, ám rákattintva ezek legördülnek és megjelenítődik az összes fontos részlet.

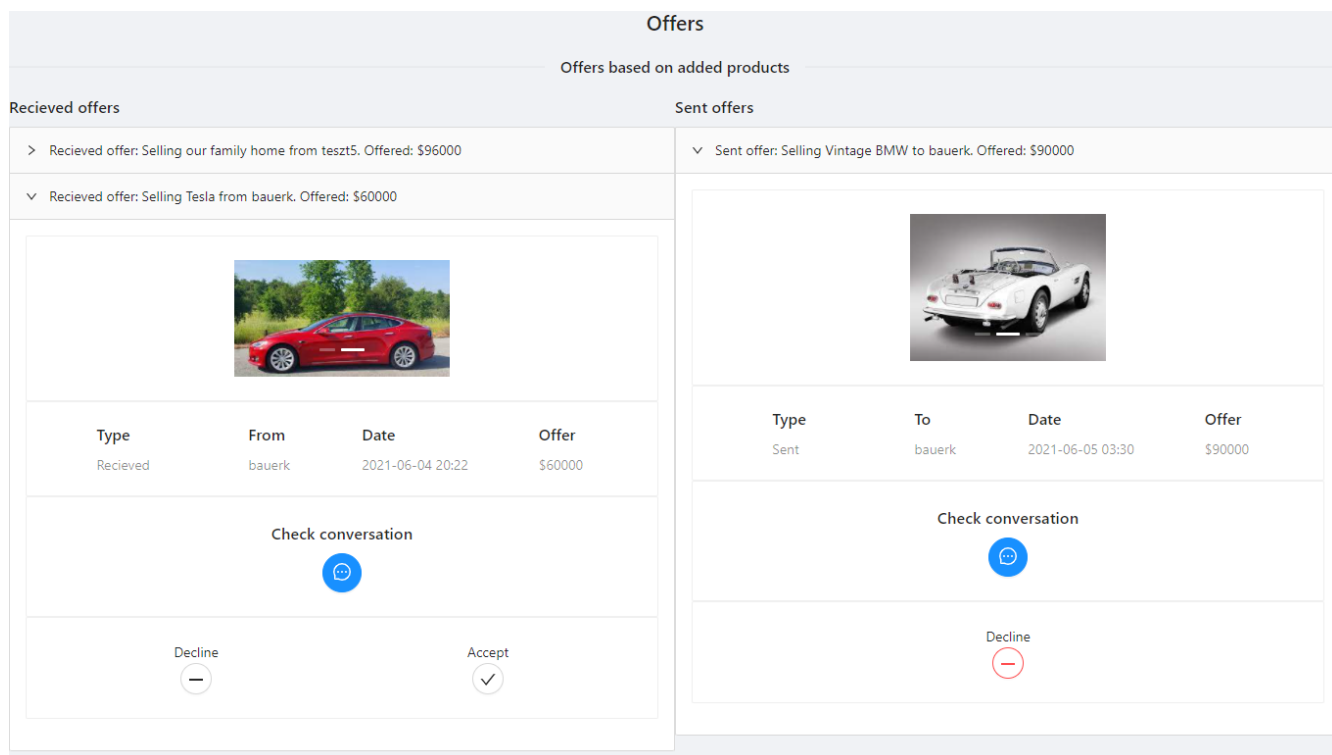
7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE



7.26. ábra. Ajánlatok komponens főmenü felépítése

Ajánlatok komponens árajánlatai

A beérkező árajánlatokat a vásárlók küldték a mi eladó termékünkre a küldött árajánlat esetében pedig mi vagyunk a vásárlók, és ajánlatot tettünk más eladók termékeire.



7.27. ábra. Kapott és küldött árajánlatok

Az árajánlatok elemeit lekattintva megjelennek az ajánlatok termékkártyái. Egy termékkártya tartalmazza a termék képét, az árajánlat típusát, annak a felhasználónak a nevét akiktől kaptunk vagy akinek küldtünk árajánlatot, illetve az árajánlat értékét. A termék képére kattintva átirányítódik a felhasználó a termék adatlapjára.

A termékünkre beérkező árajánlatot lehetőségünk van visszautasítani vagy elfogadni. Visszautasítás esetén mindkét felhasználó számára törlődik az ajánlat, illetve a termék adatlapján lévő ajánlatok táblázatból is törlődik az. Egy termékre érkező ajánlat elfogadásakor, a termékünk úgy az eladó, mint a vásárló számára bekerül a profiljában szereplő *Eladott termékek* közé és törlődik a marketplacerről, valamint a *Termékek* komponensből.

Az általunk küldött ajánlatokat visszavonhatjuk a *Decline* gombra kattintva. Ekkor mindkét fél számára törlődik az ajánlat.

Legyen szó ajánlatok elfogadásáról vagy visszautasításáról az érintett fél mindig valós időben értesítve lesz, és az ő ajánlatai is azonnal frissülnek.

Ajánlatok komponens kérelmei

A kérelmek három részre vannak osztva: létrehozott kérelmek, a kérelmeinkre érkezett termékek és más felhasználó kérelmeire elküldött saját termékeink. Az érkezett és elküldött kérelmek esetében a lenyíló elem további két lenyíló menüt tartalmaz, ami által meg tudjuk nézni, hogy melyik kérelemre milyen terméket kaptunk vagy küldtünk.

Az általunk létrehozott kérelmek törlése esetén automatikusan törlődnek a kérelemre beérkező ajánlatok is. Az összes olyan felhasználónak, aki a kérelemre elküldte a termékét, úgyszintén törlődik ez a kérelem. Ha az általunk létrehozott kérelem egyedi, akkor a *Buy Requests* menüpontban lévő összes kérelem közül is törlődni fog. Csoportosított kérelem esetén nem fog törlődni, csupán nem fogunk ezután ajánlatokat kapni az adott kérelem kritériumai szerint. Ahogy az árajánlatok esetében, itt is ugyanúgy lehetőségünk van egy kérelemre elküldött termékünk visszavonásához.

Kérelem elfogadásakor, hasonlóan az árajánlat elfogadáshoz, mindkét fél számára bekerül a termék a *Sold Products*ba. A kérelmünk még továbbra is él, és elfogadhatunk más termékeket is, amik az adott kérelemre érkeztek.

Akár csak az árajánlatok esetében, itt is az összes felhasználói tevékenység során valós időben értesítve lesz a partnerünk és az ő kérelmei is automatikusan frissülnek.

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

Offers based on buying requests

Your buying requests

▼ Request made on main category Cars

Request

Criteria

Category: Cars

Price starting from 1 to 100000

Delete request

> Request made on main category Cars

> Request made on main category Houses

> Request made on main category Cars

Recieved offers for your buying request

▼ Recieved offer from user bauerk

> Request made on main category Cars

> Recieved offer: Selling Vintage BMW from bauerk. Offered: \$90000

Check conversation

Decline

Accept

Sent offers for buying requests

▼ Sent product Selling our family home for user bauerk buying request

▼ Request made on main category Houses


Request

Criteria

Category: Houses

Price starting from 1 to 100000

▼ Sent offer: Selling our family home to bauerk. Offered: \$96000



Type	To	Date	Offer
Sent	bauerk	2021-05-28 18:59	\$96000

Check conversation

Decline

7.28. ábra. Létrehozott, kapott és küldött kérelmek

Üzenetváltás a felhasználók között

Minden ajánlat esetében a legördülő elemekben szerepel *Check Conversation* gomb. Erre rákattintva a felhasználó át lesz irányítva egy privát chat felületre, ahol a partnerével cseveghet. A csevegés is valós időben történik illetve, ha egy felhasználónak érkezett olvasatlan üzenete értesítve lesz.

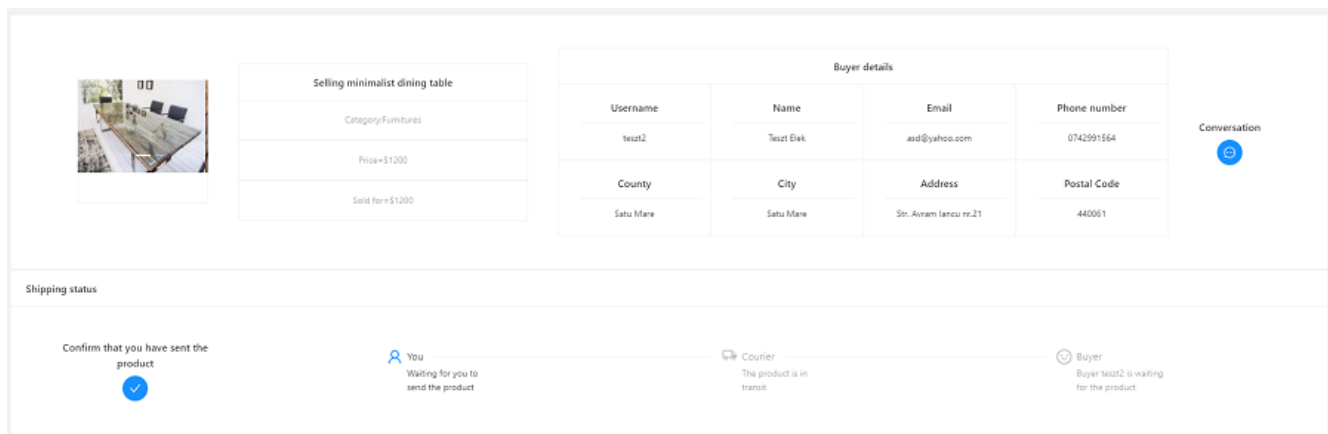
7.9.4. Eladott termékek komponens

Egy termék elfogadása esetén mindkét partner számára a termék bekerül az Eladott termékek komponenshez. Az eladó feladata lesz a csomag elküldése. Mivel csak úgy lehet vásárolni és eladni, hogy megadtuk személyes adatainkat, így a partnerek gyorsabban lebonyolíthatják az üzletet. A komponens kettéoszlik a felhasználó által eladott és vásárolt termékek szerint.

50

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE

Egy termék kártyáján megjelenő információk elkülönülnek a termékre és a partnerre vonatkozó információk, illetve a szállítási állapot szerint.



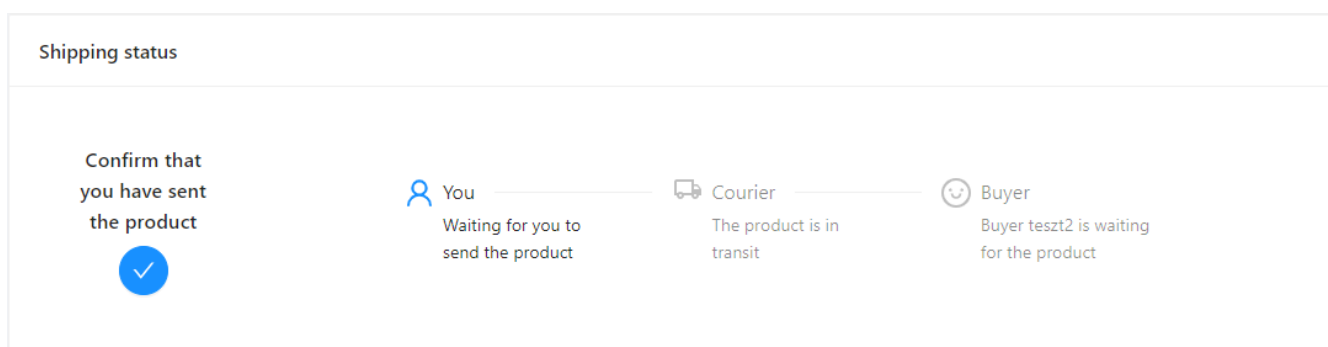
7.29. ábra. Egy eladott termék kártyája

1. Termék és partner információk

A termék képére kattintva átirányít annak adatlapjára. A mellette lévő táblázatban szerepel a termék címe, eredeti ára és a végső eladási ár. A következő táblázat tartalmazza a partnerünk személyes adatait, úgy az eladó, mint a vásárló számára így segítve a kommunikációt és a gyorsabb termékküldést. A conversation gombra kattintva chatelhetnek egymással a partnerek.

2. Szállítási állapot

Annak érdekében, hogy egy termék aktuális helyzete átláthatóbb legyen egy folyamatgrafikonon szemléltetjük azt.



7.30. ábra. Szállítási állapot

Egy termék eladásakor az eladó számára megjelenik egy *Confirm that you have sent the product* gomb. Amikor átadta a termékét a futárszolgáltatásnak a gombra kattintva

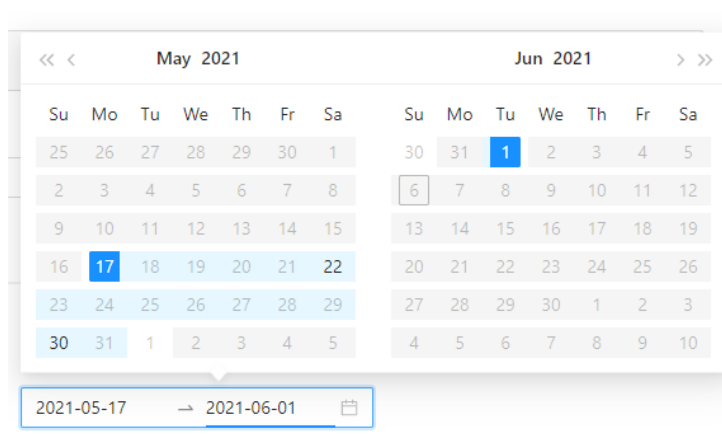
visszaigazolhatja azt a vásárló felé. Ekkor frissül a grafikon, és a vásárló számára is megjelenik egy gomb, ami által vissza tud jelezni, ha megkapta a csomagot és minden rendben volt. Mindkét visszaigazolás során, a felhasználók értesítést kapnak erről és a folyamatgrafikon is frissül valós időben.

7.10. Termékstatisztikák

Az oldalon lehetőség van eladott termékek dátum szerinti áralakulásának a követésére. Ahhoz, hogy ezt a funkciót elérjük, ki kell válasszuk a navigációban a *Statistics* menüpontot.

Ezután szűrhetünk ár szerint, illetve hierarchikus szűrés által ki kell válasszunk egy érdekelt fő és alkategóriát, esetlegesen opcionális kritériumokat is szabhatunk.

A kritériumok kiválasztása után megjelenik egy naptár az elérhető dátumokkal. Azok a napok ahol nem volt termékadás szürkével vannak jelölve és nem kiválaszthatóak. A kiválasztható dátumokra kattintva meghatározhatjuk az időintervallum két végpontját, ahol érdekelnek az eladott termékek árai.



7.31. ábra. Kiválasztható időpontok naptára

A dátumintervallum kiválasztása után megjelenik az áralakulás grafikonja, lásd 7.32. ábra. Az itt szereplő árak, az adott napon eladott termékárak átlagaiból számolódnak. A grafikon csomópontjára vagy annak közelében lévő vonalára kattintva meg tudjuk nézni az adott dátumon eladott termékeket, lásd 7.33. ábra.


A termékkártyákon szerepel a termék képe, amire kattintva megtekinthetjük a termék adatlapját. Ezen kívül, az eredeti ár és az eladási ár is.

7. FEJEZET: THE MARKETPLACE PROJEKT MŰKÖDÉSE




7.32. ábra. Eladott termékek grafikonja

Matched products of selected date



Selling Opel Moavo truck in great condition
Sold for \$22000
Original price \$25000



Selling NEWEST!! Opel Corsa
Sold for \$35000
Original price \$35000

7.33. ábra. Adott napon eladott termékek

8. fejezet

Összefoglaló

8.1. Következtetések

Következtetésként elmondható, hogy a **The Marketplace** webalkalmazás sikeresen meg lett valósítva. A fő cél az volt, hogy egy olyan piactéri alkalmazás szülessen, ahol a termékeket pontosan és gyorsan meg lehessen találni anélkül, hogy a felhasználókat keretek közé szorítaná. Ezek ötvözésével egy felhasználóbarát alkalmazást biztosítottunk, ami teljesíti a fenti célt. Az erre épülő új és újragondolt funkciók pedig új lehetőségeket tárnak fel egy piactéri alkalmazás működésében.

8.2. Továbbfejlesztési lehetőségek

A The Marketplace alkalmazás számos továbbfejlesztést igényel még, hogy valós körülmények között is megállja a helyét. Ezekről a következő sorokban lesz egy pár felsorolva:

- Az alkalmazás kitelepítése, ezáltal valós adatokkal és valós terhelés alatt lehetne az oldalt kipróbálni, ez pedig segítene a felbukkanó hibák megtalálásában, illetve a jelenlegi működés javításában.
- Adminisztrációs felület hozzáadása.
- A MongoDB adatbázis struktúra újraépítése Tree Design Patternt követve, amely által jobban kihasználnánk a beágyazottságból adódó lehetőségeket.
- Nemzetköziesítés: jelenleg az oldal csak az angol nyelvet támogatja.
- Responsive felhasználói felület kialakítása.
- A személyes adatok biztonságosabb kezelése.
- A React-Native segítségével az alkalmazást meg lehetne írni mobil eszközökre is, ezáltal több felhasználóhoz jutna el.

Irodalomjegyzék

- [1] Axios Hivatalos dokumentációja. URL <https://axios-http.com/docs/intro>. Látogatva: 2020.06.07.
- [2] MomentJS Hivatalos dokumentációja. URL <https://momentjs.com/docs/>. Látogatva: 2020.06.07.
- [3] MongoDB Hivatalos weboldala. URL <https://docs.mongodb.com/manual/introduction/>. Látogatva: 2020.06.07.
- [4] Node.js Hivatalos dokumentációja. URL <https://nodejs.org/en/about/>. Látogatva: 2020.06.07.
- [5] React Hivatalos dokumentációja. URL <https://hu.reactjs.org/tutorial/tutorial.html>. Látogatva: 2020.06.07.
- [6] Redux Hivatalos dokumentációja. URL <https://redux.js.org/introduction/getting-started>. Látogatva: 2020.06.07.
- [7] Socket.IO Hivatalos dokumentációja. URL <https://socket.io/docs/v3>. Látogatva: 2020.06.07.
- [8] Lewington, G. Everything You Need To Know About Socket.IO. Technical report, Ably, 2021. URL <https://ably.com/topic/socketio#client>.
- [9] Massé, M. *Rest API Design Rulebook*. O'Reilly Media, 2012.
- [10] Subramanian, V. *Pro Mern Stack*. Apress, 2019.
- [11] Tarver, E. Customer to Customer (C2C). Technical report, Investopedia, 2020. URL <https://www.investopedia.com/terms/c/ctoc.asp>.