

Szegedi SzB School of Business Technikum



Szakképzés neve: Szoftverfejlesztő és -tesztelő

Témavezető: Bodrogi Péter Róbert

Készítő: Bali Martin Márk

Szeged, 2023.

Szoftverfejlesztés - és tesztelés vizsgaremek

Szakmai azonosító: 5-0613-12-03

_Eliminator

I. Tartalomjegyzék

I. Tartalomjegyzék.....	2
II. Vizsgaremek adatai	3
III. A projekt témájának kifejtése	3
IV. Időmenedzsment	5
a.) Vendég felhasználó	6
b.) Regisztrált felhasználó	8
c.) Ügynök szintű felhasználó	12
d.) Admin	14
VI. Az alkalmazás futtatása	16
a.) Frontend	17
b.) Backend	17
c.) Adatbázis	17
VII. Dizájn	17
VIII. Front-end	18
IX. Back-end	20
X. Adatbázis	22
XI. Tesztelés	23
a.) Teszt eset I.	24
b.) Teszt eset II.	24
c.) Test este III.	25
XII. Hardver követelmények	25
XIII. Szoftver követelmények	25
XIV. SWOT	26
XV. Felhasznált szoftverek	27
XVI. Források	27

II. Vizsgaremek adatai

Projekt neve: Eliminator

Projekt készítői: Bali Martin Márk, Földi Áron, Krisztin Németh Martin

Projekt időtartama: 2022. 09. - 2023. 05.

Készítők email címei: xxxmartyn69@gmail.com, foldi.aron220@gmail.com, krisztinnemeth.martin64@gmail.com

Képzés neve: Szoftverfejlesztő és - tesztelő

Szaktanárok nevei: Babinszky Mónika, Bodrogi Róbert Péter

Projekt témája: Különböző ügynökök felbérelésének lehetősége

Projekt tevékenysége: Ügynökök bérlete

III. A projekt témájának kifejtése

Egy webalkalmazás, ahol olyan személyeket, vagyis ügynököket lehet felbérelni, akik a felhasználók által megadott célpontokat likvidálják. A felhasználóknak a munka elvégzése után, lehetőségük van egy egytől-ötig terjedő skálán értékelni a szolgáltatást. A felhasználóélmény javítása érdekében az ügynökök neve mellett színes csillagok formájában jelenik meg a szolgáltatást igénybevevők visszajelzése. Valamint, ezek mellett megjelenik a várható költség, egy kitalált pénznemben (M\$).

A projektünk alapgondolata az volt, hogy ne egy mindenki által már átvett témát dolgozzunk fel, mint például pizza rendelő alkalmazás, vagy egy általános webshop. A témaválasztásnál fő szempont volt, hogy egy olyan gondolat szülessen meg, ami elég nagy ahhoz, hogy - ha szükség van rá - le lehessen szűkíteni úgy, hogy közben a gondolat lényegi eleme ne vesszen el.

A weboldal fő váza és alapfunkciói közül a legalapabb, hogy be lehessen jelentkezni és különböző felhasználókat különböztethessünk meg (Vendég, Regisztrált felhasználó, Ügynök és Admin). A különböző jogkörök különböző funkciókat foglalnak magukba, például a vendég nem

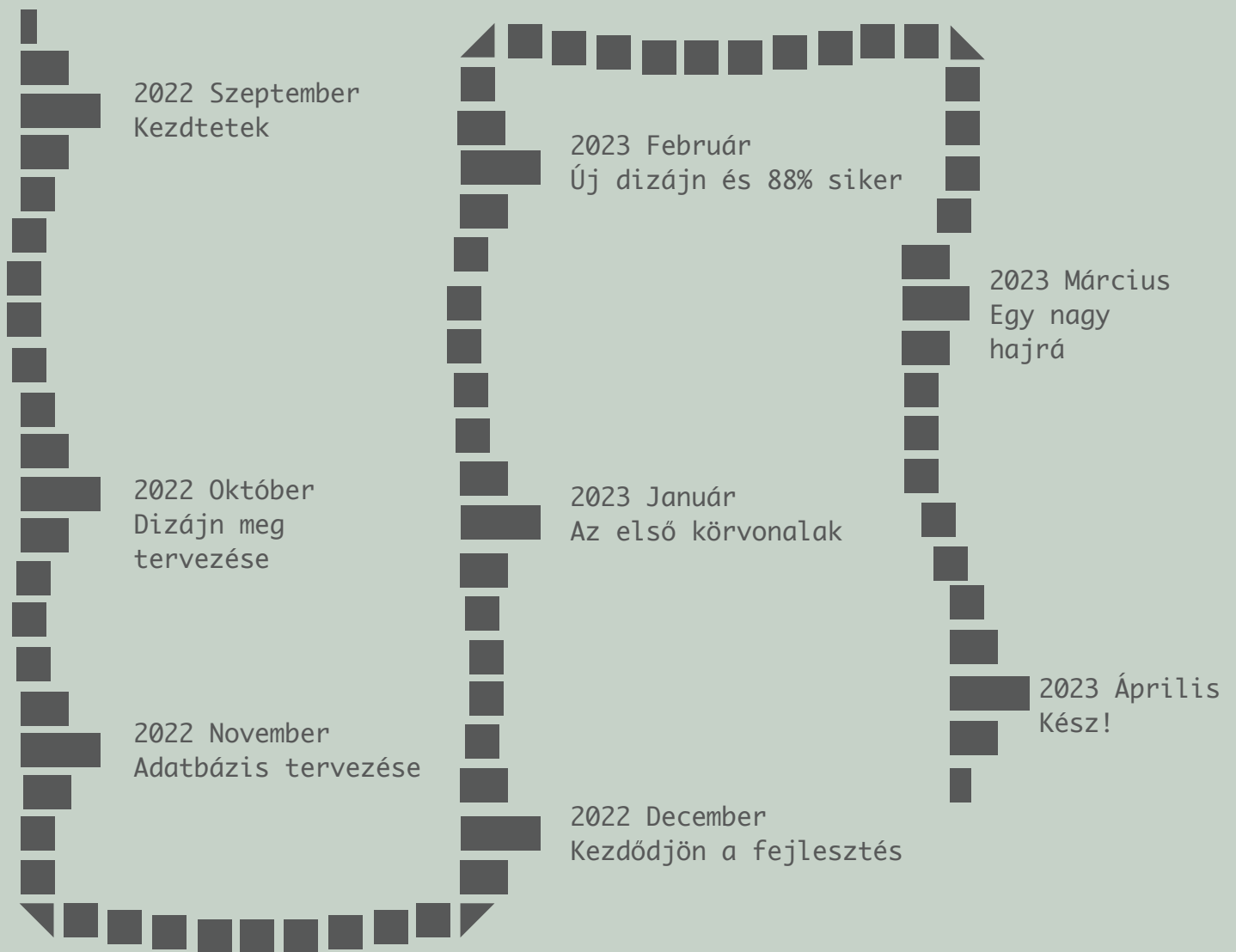
tudja kilistázni az ügynököket; az egyszerű regisztrált nem tud ügynököt regisztrálni. Emellett, alapnak tekinthető az értékelési rendszer, valamint a munkák nyomon követése. Az ügynökök előlnézetére, részletesebb megtekintésére, valamint a megrendelésükre is van lehetőség.

A weboldalunk célközönségének alapvetően a tehetőbb, módosabb réteget tekintjük, ezt munkatársaink bérezése is tükrözi.



IV. Időmenedzsment

A projekt elkészítésének folyamatát, időmenedzsmentjét a következő ábrán szemléltetjük.

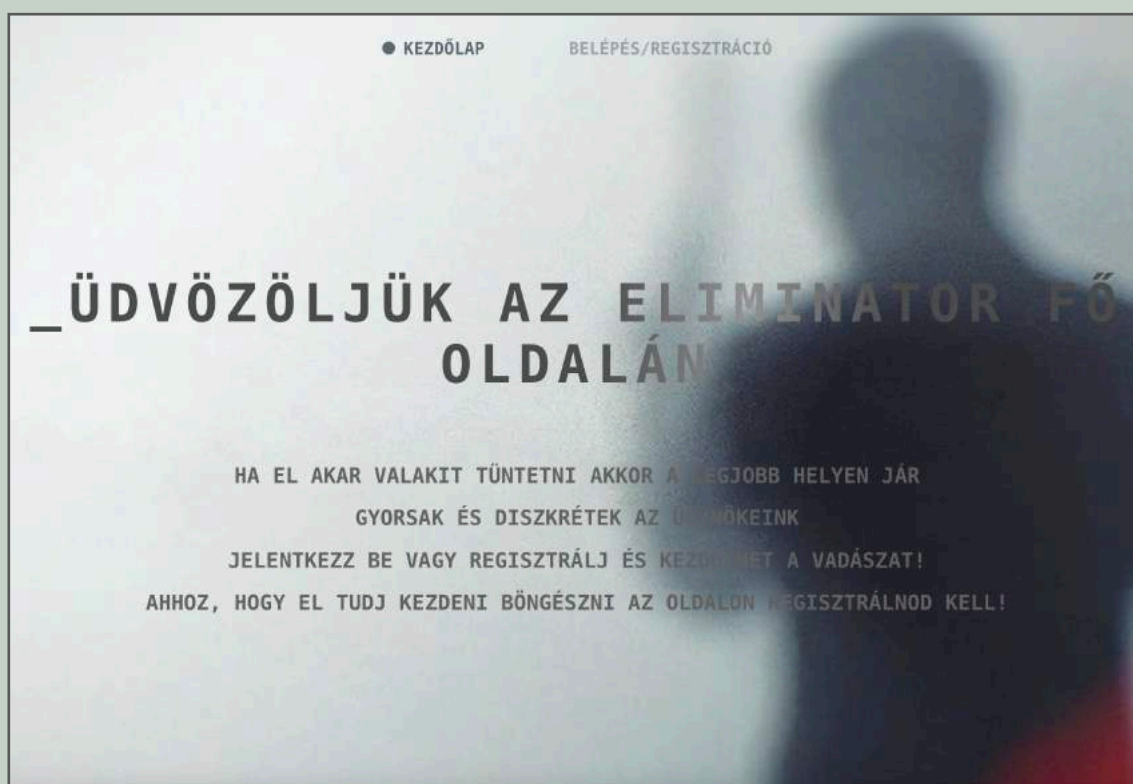


V. A felhasználói felületről részletesebben

A felület részletes bemutatása érdekében a fentebb említett különböző jogkörök alapján szedtük pontokba a felhasználói nézeteket.

a.) Vendég felhasználó

Ez a legkisebb jogkörrel rendelkező felhasználótípus. Ennek a fajta felhasználónak csak a kezdőoldal megtekintésére, regisztrációra és/vagy belépésre van lehetősége. Az oldalak közötti váltásra a fenti navigációs bárban van lehetőség. Kivételt képez ez alól a regisztráció, hiszen ez a belépés fülön belül egy külön gombon található.

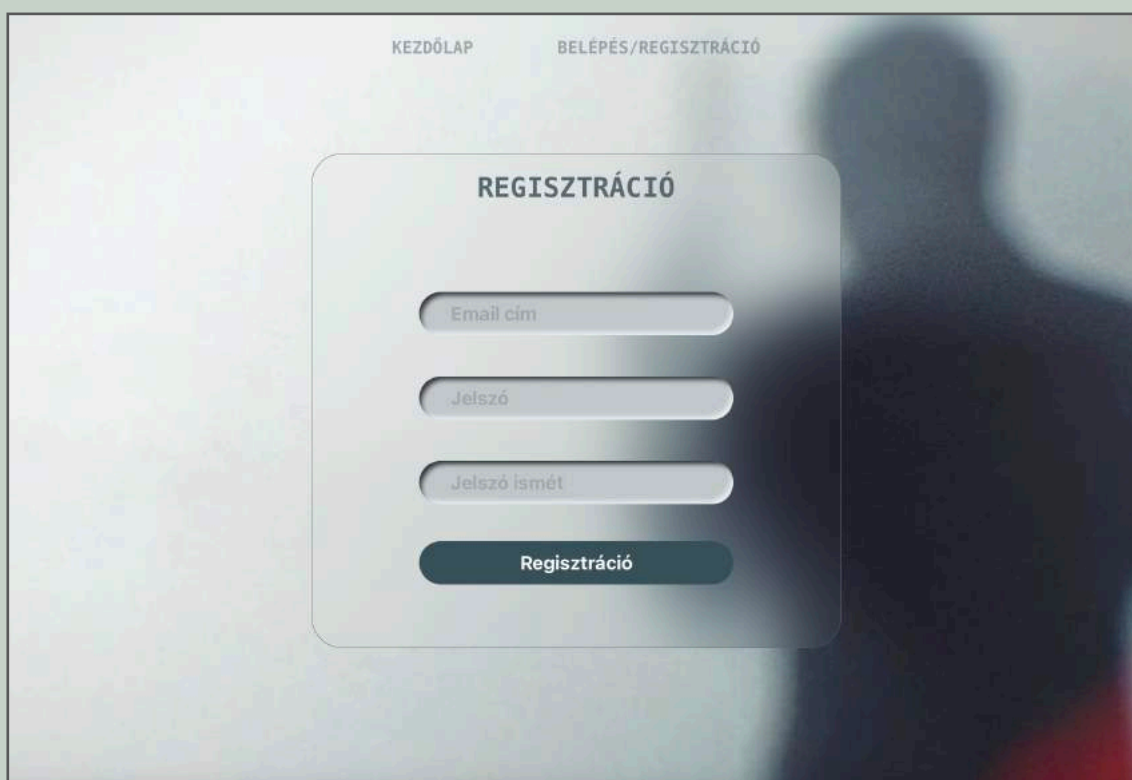


A kezdőlapon egy üdvözlő szöveg és egy rövid ismertető jelenik meg.



The image shows a website header with two tabs: 'KEZDŐLAP' and 'BELÉPÉS/REGISZTRÁCIÓ'. The 'BELÉPÉS/REGISZTRÁCIÓ' tab is active, indicated by a black dot. Below the tabs is a white rounded rectangle containing the login form. The form is titled 'BELÉPÉS' in bold. It features two input fields: 'Email cím' and 'Jelszó'. Below these fields is a dark green button labeled 'Belépés'. Underneath the button is a horizontal line with the word 'vagy' in the center. Below the line is another dark green button labeled 'Regisztráció'.

A bejelentkezés fülnél kettő inputmező jelenik meg, az egyik a felhasználói azonosító, a másik a jelszó. Valamint egy belépési gomb.

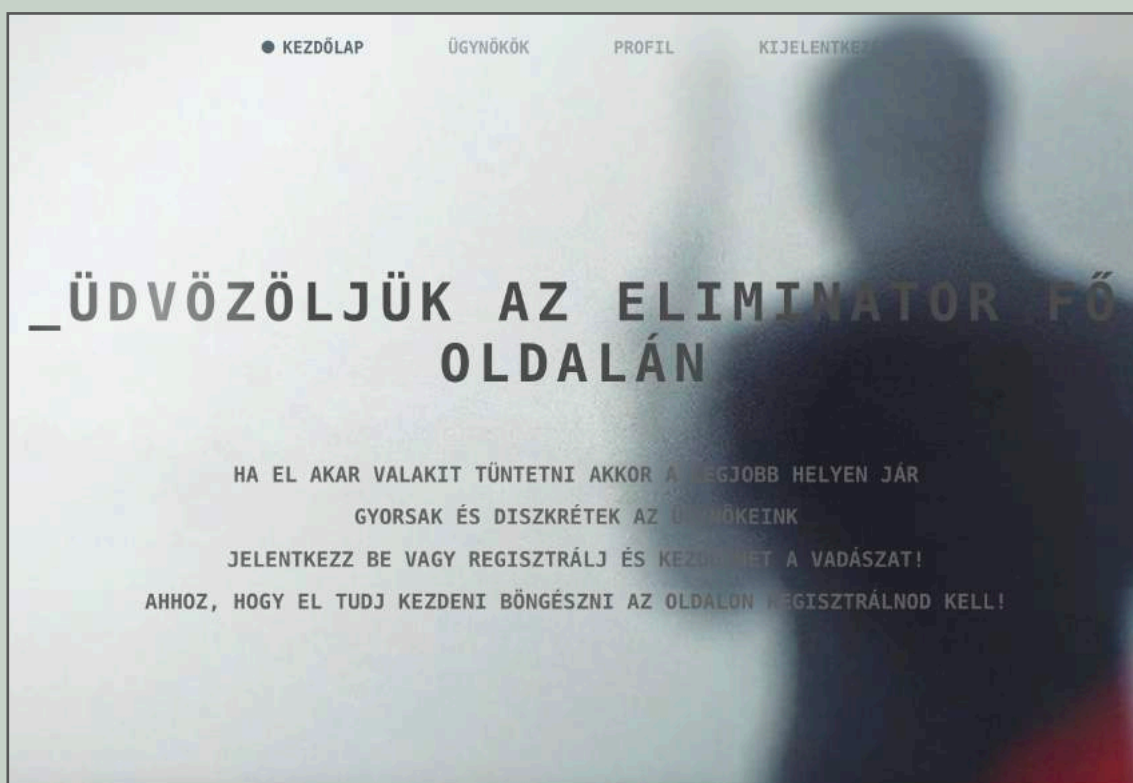


The image shows a website header with two tabs: 'KEZDŐLAP' and 'BELÉPÉS/REGISZTRÁCIÓ'. The 'BELÉPÉS/REGISZTRÁCIÓ' tab is active, indicated by a black dot. Below the tabs is a white rounded rectangle containing the registration form. The form is titled 'REGISZTRÁCIÓ' in bold. It features three input fields: 'Email cím', 'Jelszó', and 'Jelszó ismét'. Below these fields is a dark green button labeled 'Regisztráció'.

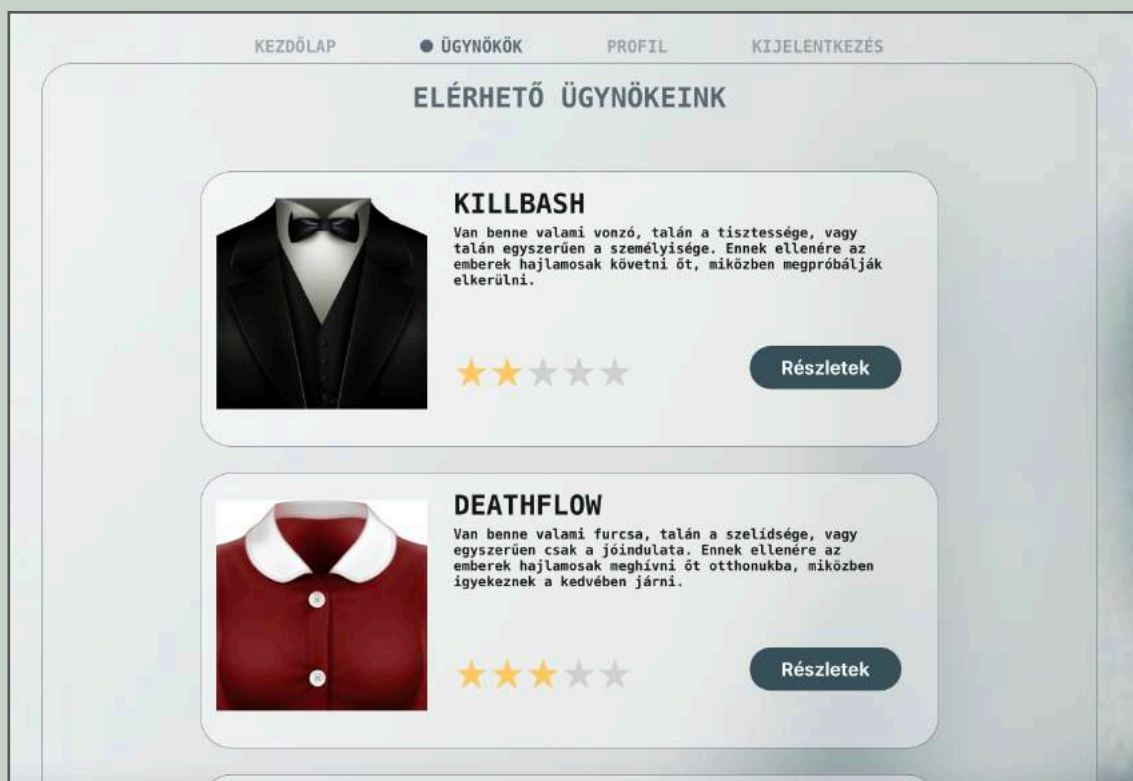
A regisztráció lehetőségnél három inputmező jelenik meg, valamint egy regisztráció gomb. Az egyik a felhasználói azonosító, a második a jelszó, a harmadik a jelszó ismét.

b.) Regisztrált felhasználó

Ez a mindenki számára elérhető jogkör, egy regisztráció és egy belépés után érheti el a felhasználó. Amiben eltér a fentebb említett Vendégtől, hogy neki lehetősége van részletesebben böngészni az oldalon, valamint ügynököket tud rendelni.



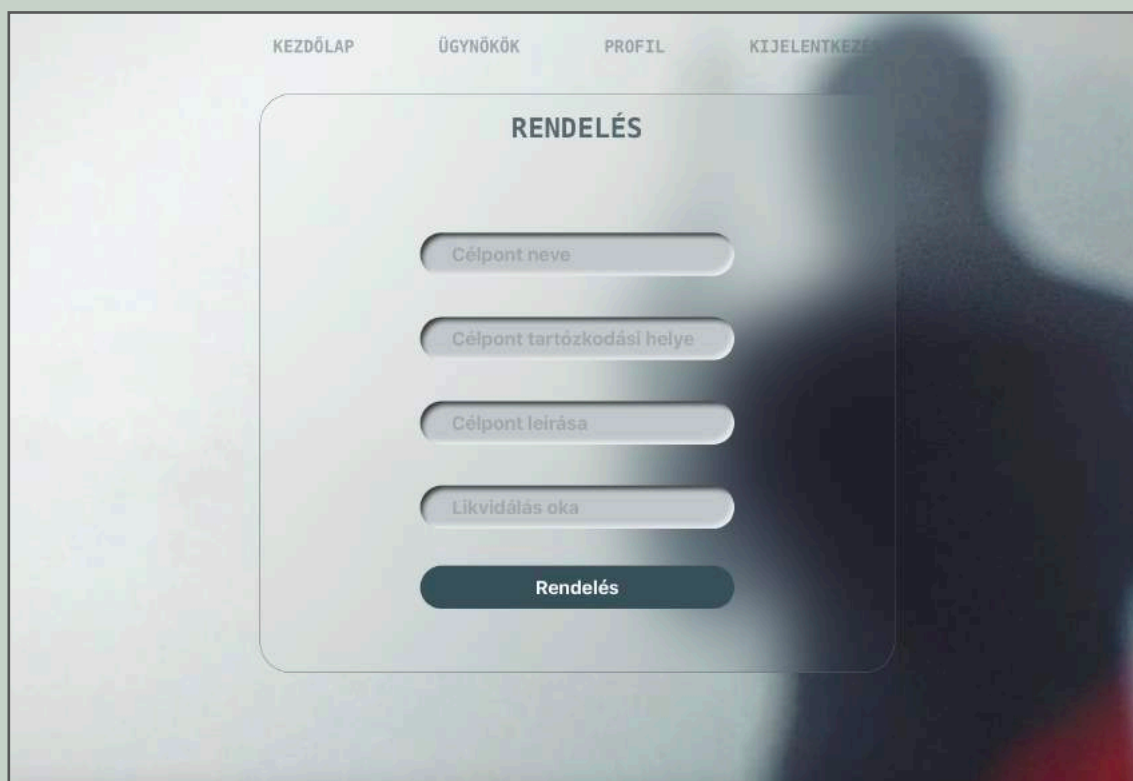
A kezdőlap változatlanul jelenik meg e felhasználó esetében. Lásd a részleteit az a.) Vendég felhasználó pontban.



Ebben a menüpontban jelennek meg az elérhető ügynökök névvel, egy illusztrációval, rövid leírással és az aktuális értékelésükkel, valamint egy részletek gombbal. Elérési útvonala, a navigációs menüben az ügynökök pontra kattintva.



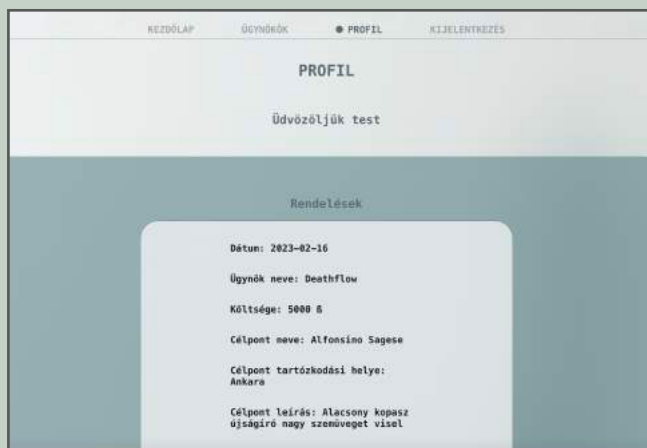
Itt már csak a kiválasztott munkatárs jelenik meg a fentebbi pontban említett információkkal, de itt a leírás sokkal átfogóbb, valamint ehhez párosul egy várható költség is. A felhasználónak ebben a menüpontban már lehetősége van felbérelni őt. Elérési útvonala az ügynökök pontban egy választott, részletek gombra kattintással érhető el.



The screenshot shows a web application interface with a navigation bar at the top containing the links: KEZDŐLAP, ÜGYNÖKÖK, PROFIL, and KIJELENTKEZÉS. The main content area is titled 'RENDELÉS' and contains a form with five input fields: 'Célpont neve', 'Célpont tartózkodási helye', 'Célpont leírása', and 'Likvidálás oka'. Below these fields is a dark blue button labeled 'Rendelés'. The form is set against a background image of a person's face.

Ezen a felületen egy űrlap jelenik meg, négy darab mezővel (Célpont neve, Célpont tartózkodási helye, Célpont leírása, Likvidálás oka). Ezek értelemszerű kitöltésével és a rendelésre gombra kattintással a kiválasztott ügynök megkapja a munkát. Elérési útvonala, az ügynökök menüpontban a részletek gombra kattintással, a rendelés gombra kattintva érhető el.

V. A felhasználói felületről részletesebben



Ez a felület név szerint üdvözlí a felhasználót, alpontjában a rendelések címszó alatt kilistázódik az összes rendelése az ügyfélnek, részletekbe menően. Valamint ezek státuszát is megjeleníti az oldal. Nagyobb felhasználóélmény miatt különböző színek jelölik a rendelések állapotát. Elérési útvonala, a bejelentkezés után rögtön ide dob be az oldal, valamint a navigációs felületen a profil fül megnyitásával érhető el.



Ezen a ponton a felhasználó az elvégzett munkát tudja értékelni egy egytől-ötig terjedő skálán, amelyek csillagokkal vannak jelölve. Elérési útvonala, a profilon belül egy olyan rendelés amely még nem lett értékelve, de már el van végezve. Ezt egy gomb is jelzi, értékelésre vár címszóval.

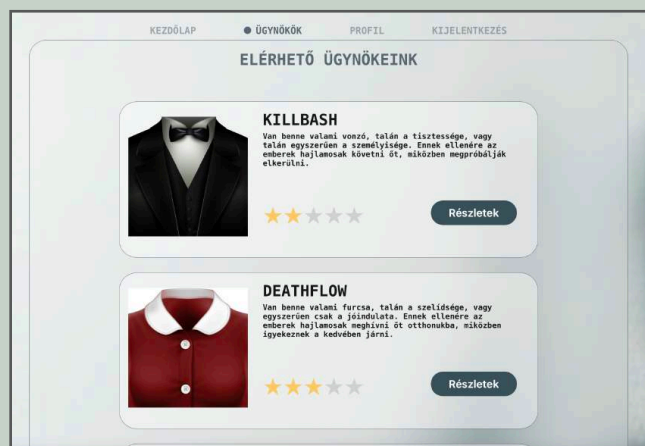
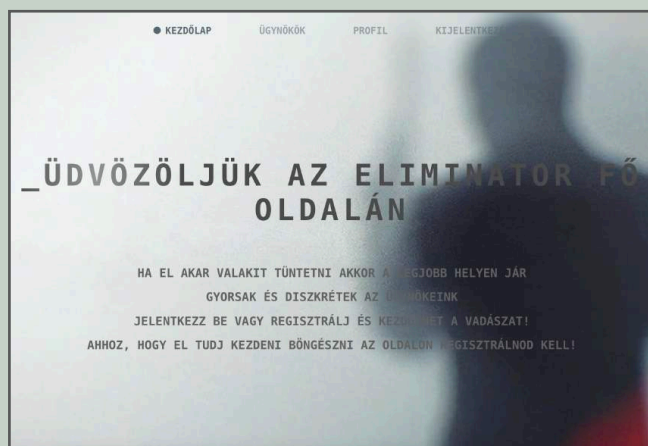
Kijelentkezés menüpont:

Ez egy külön menüpont, amit a navigációs bárban lehet elérni. Funkciója kijelentkeztetni az ügyfelet az oldalról.

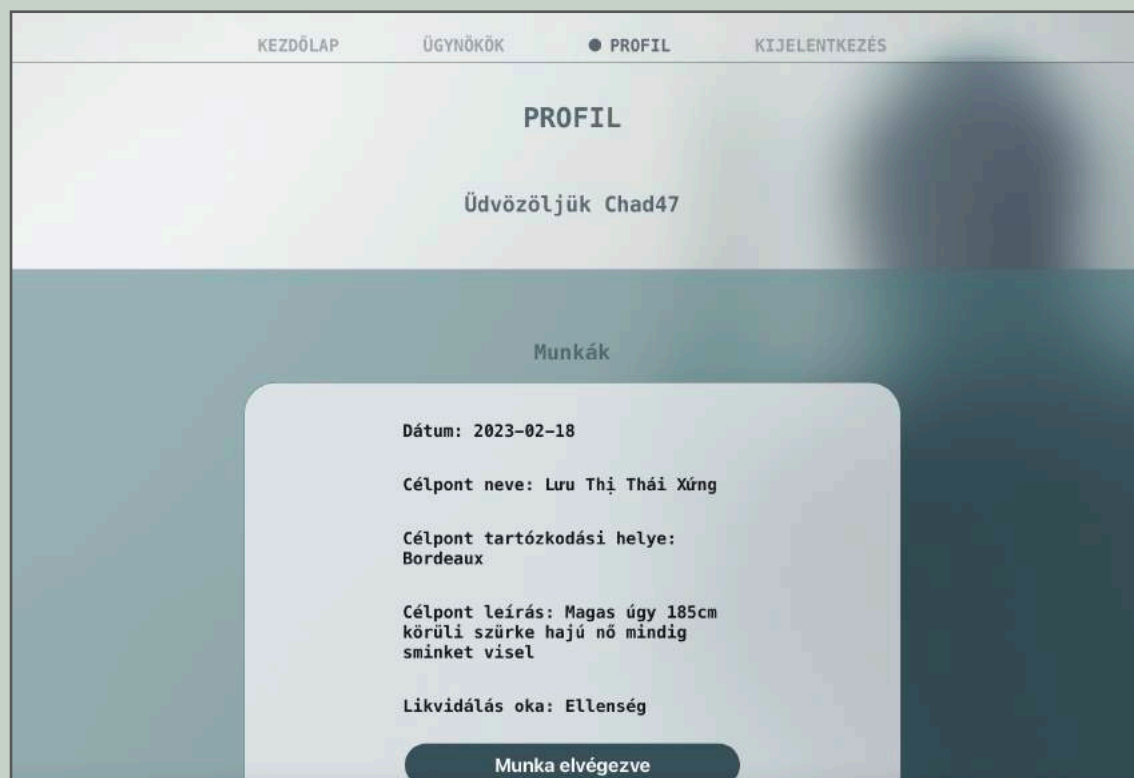
c.) Ügynök szintű felhasználó

Ez egy különlegesebb jogkör hiszen, ezt csak Admin tudja létrehozni valamint, a fentebbi pontokban említett ügynökök felhasználói oldala ez.

V. A felhasználói felületről részletesebben



Ugyanaz mint a fentebbi b.) Regisztrált felhasználó pontban említett.



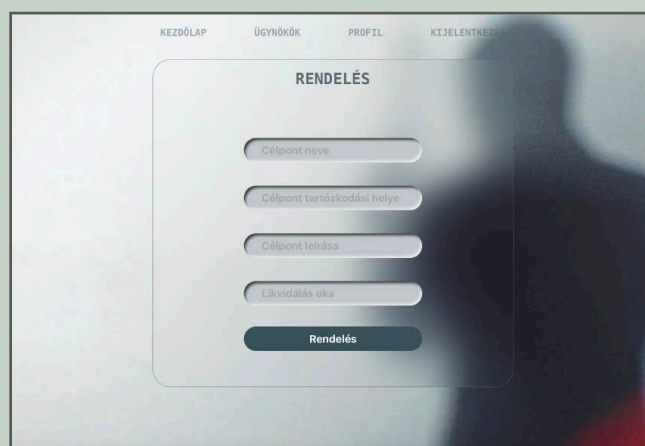
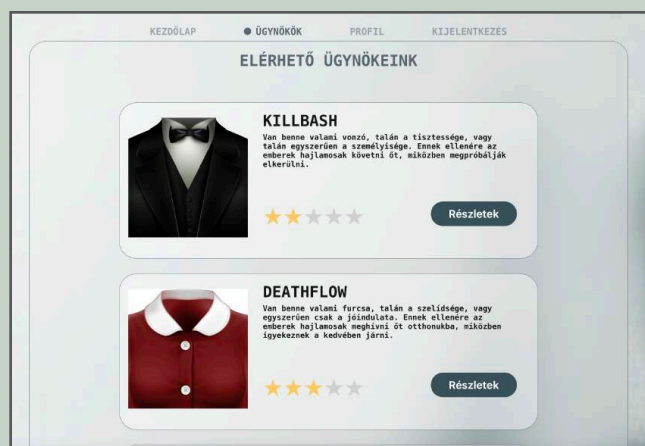
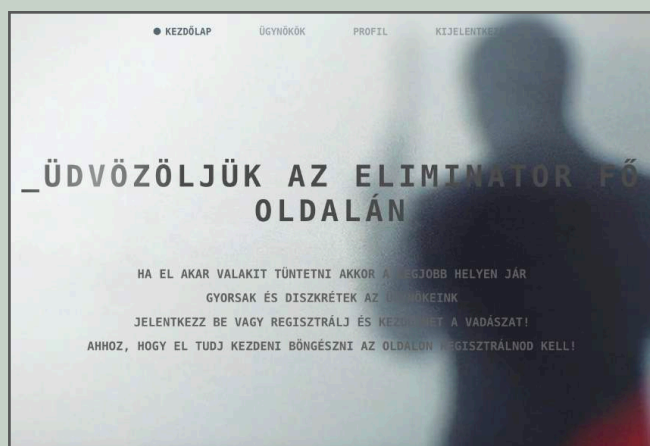
A profil menüpontban szintúgy mint az előzőben, név szerint üdvözlí a felhasználót, de itt az alponban a munkák címszó alatt az ügynök feladatai láthatóak (az elvégzendőek és az elvégzettek is). Az ügynök az új munkákat itt találja és itt is tudja lezárni azokat, a munka elvégzése gombra kattintva.

Kijelentkezés menüpont:

Ez egy külön menüpont, amit a navigációs bárban lehet elérni. Funkciója kijelentkeztetni az ügyfelet az oldalról.

d.) Admin

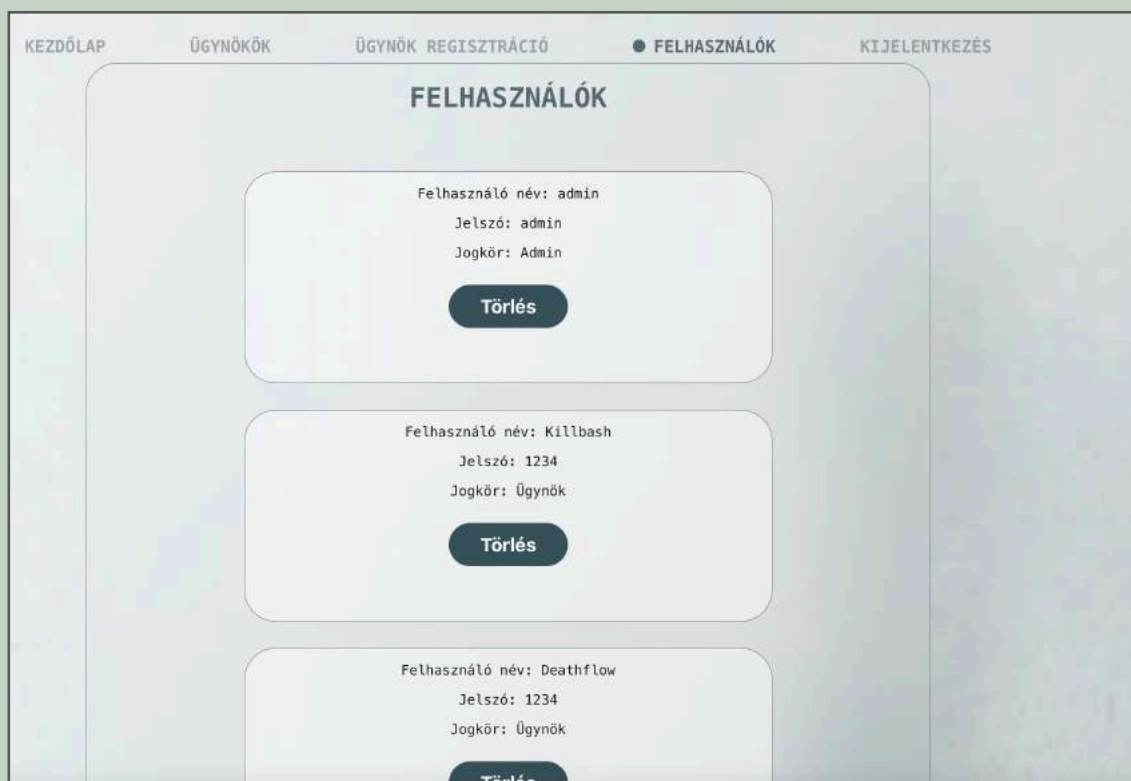
Ez a legmagasabb szintű jogkörrel rendelkező felhasználói szint. Ezt nem lehet regisztrálni az oldalon, ezt csak külsőleg lehet az adatbázisba juttatni. Erre biztonságtechnikai okokból van szükség, hiszen ez a felhasználó képes törölni az adatbázisból az adatokat.



Ugyanaz mint a fentebbi b.) Regisztrált felhasználó pontban említett.

The screenshot shows a web application interface with a navigation bar at the top containing four items: 'KEZDŐLAP', 'ÜGYNÖKÖK', '● ÜGYNÖK REGISZTRÁCIÓ' (which is the active page), and 'FELHASZNÁLÓK'. The main content area is a registration form for agents. It consists of several input fields with rounded corners and light blue borders, each with a label above it: 'Ügynök email címe', 'Ügynök jelszava', 'Ügynök jelszava ismét', 'Ügynök neve', 'Ügynök rövid leírása', 'Ügynök leírása', 'Ügynök ára', and 'Ügynök profil képe'. Below the last input field is a small vertical double-headed arrow icon. At the bottom of the form is a dark blue button with the white text 'Regisztráció'. The right side of the image is partially obscured by a dark, blurry vertical bar.

Az ügynök regisztrálása oldalon lehet új ügynököket hozzáadni a listához, az ügynökök mellé egy felhasználói fiók is társul. Az oldalon az űrlapmező teljes kitöltésével és a regisztráció gomb megnyomásával lehet megtenni ezt. Elérési útvonal a navigációs listában az ügynökregisztráció menüpont alatt.



Ezen a menüponton megjelenik az összes felhasználó és azok adatai mint felhasználónév, jelszó, jogkör, valamint itt van lehetőség törölni a felhasználókat, a törlés gombra kattintással. Elérési útvonal a navigációs listában, a felhasználók lista.

Kijelentkezés menüpont:

Ez egy külön menüpont, amit a navigációs bárban lehet elérni. Funkciója kijelentkeztetni az ügyfelet az oldalról.

VI. Az alkalmazás futtatása

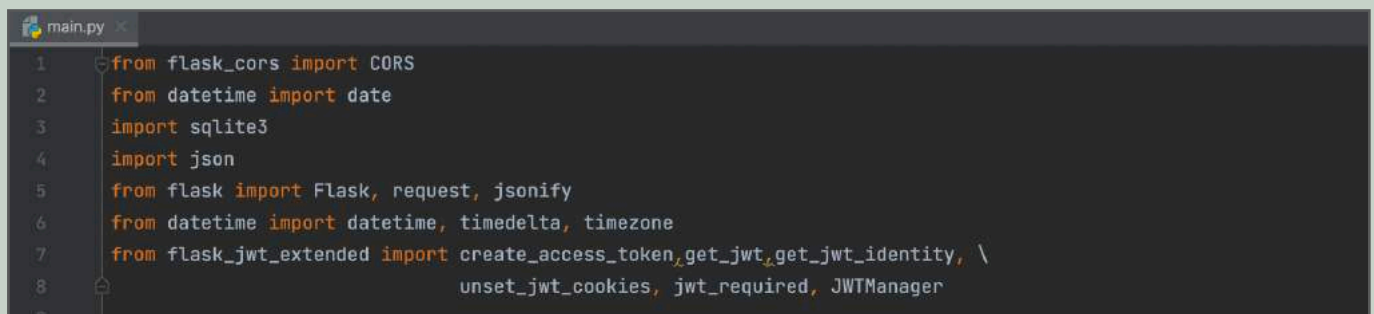
A dokumentáció ezen részében az alkalmazás localhost-on való futtatását mutatjuk be, külön pontokba szedve a frontend és a backend részeket. FONTOS! Először a Backend-et indítsuk el, hogy a Frontend már megfelelő adatokat kapjon. Ezen pont be nem tartása előre nem látható hibákat okozhat, amennyiben ez mégis megtörténik, állítsuk le a Frontend szerverünket, majd a Backend-et és indítsuk el a megfelelő sorrendben újra.

a.) Frontend

A Frontend elindításához nem kell mást tennünk mint, hogy megnyitjuk a Frontend mappát Visual Studio Code, majd a terminálba beírjuk a következő parancsot: `npm start`. Ezután a szerver elkezd betölteni a megfelelő fájlokat és ha ez befejeződött akkor az alapértelmezett böngészőben felugrik az oldal.

b.) Backend

Backend futtatására a legegyszerűbb mód, ha megnyitjuk a Backend mappát Pycharmmal, majd a következő képen látható csomagokat beimportáljuk.



```

1  from flask_cors import CORS
2  from datetime import date
3  import sqlite3
4  import json
5  from flask import Flask, request, jsonify
6  from datetime import datetime, timedelta, timezone
7  from flask_jwt_extended import create_access_token, get_jwt, get_jwt_identity, \
8  unset_jwt_cookies, jwt_required, JWTManager

```

Majd ha ezeket beimportáltuk és nem látunk pirossal aláhúzva semmit a szerkesztőben, akkor indítsuk el a Script-et. Ha ez megtörtént akkor a terminál visszajelzést fog adni nekünk, hogy minden megfelelően működik és elindult a szerverünk sikeresen.

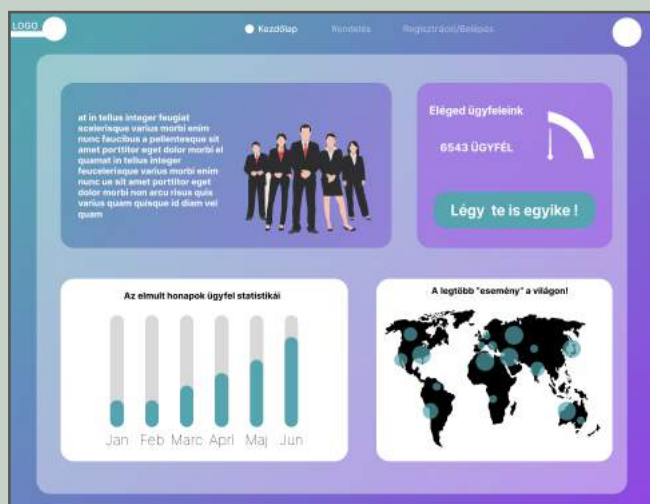
c.) Adatbázis

Mivel az adatbázisunk SQLite típusú adatbázis, ezért nem kell külön szerveren futtatni, hanem a Backend-ünk mellett található és az veszi fel vele a kapcsolatot és kezeli, így semmi dolgunk vele.

VII. Dizájn

A dizájn volt a legelső lépés a projektünkben, ezt figmában készítettük el. A projektünk végleges verziója az alap dizájnhoz képest átdolgozott, de az alapfelvetés, alapgondolat nem változott. A dizájn tervezésénél a következőket tartottuk szem előtt: nem egy átlagos weboldal elrendezést, nézetet szerettünk volna készíteni,

hanem inkább egy alkalmazáshoz hasonlót. A színválasztásnál arra figyeltünk, hogy ne a megszokott mindenki által használt színekombinációk legyenek, hanem kicsit pasztelesebb, noir-ra hajazóbb színvilág. Az átdolgozásra azért került sor, mert a legelső színátmenetes háttér és színpaletta (lásd lejjebb az első képen) összeállítás nem illett egészen a weblap témájához. Az újabb dizájn (lásd lejjebb a második képen) már háttérképet tartalmaz, valamint sokkal átgondoltabb színpalettát.



VIII. Front-end

A Frontend-hez React-et használtunk, mert az előzetesen választott Backend nyelvünk a Python volt és kutatásaink szerint ez illik hozzá a legjobban. Valamint könnyen modularizálhatóságával nagyban megkönnyítette a munkánkat. Ezért az oldal felépítése úgy néz ki, hogy felül található egy dinamikusan változó navigációs sáv, majd ez alá tölti be az oldal a megfelelő komponenseket. Minden egyes oldal egy külön komponens, amely a components mappában található. Ezeket a komponenseket az App.js fájl kezeli le és tölti be a megfelelő időben. Az oldal elemei jsx fájlok, amelyek a React saját fájljai. Ezek közül a profil.jsx fájlt mutatjuk be részletesebben.

```
src > components > Profil.jsx > Profile > orders.map() callback
1 import { useState, useEffect } from 'react'
2 import { Link } from "react-router-dom"
3 import axios from "axios";
4
```

Ezeket az importokat használtuk a legtöbb esetben. Az első az azért felel, hogy egyes változókat vagy komponensek változását figyelje és futtasson megfelelő funkciókat. Valamint ez figyel arra is, hogyha az oldal betöltődött akkor futtasson le egy funkciót.

A következő import az oldalon való linkváltásokban segít.

A legutolsó import a Backend-del való kapcsolódás megkönnyítésére szolgál.

```

13
14 function getData() {
15   axios({
16     method: "POST",
17     url: "http://127.0.0.1:5000/profile",
18     headers: {
19       Authorization: 'Bearer ' + props.token
20     },
21     data: {
22       id: localStorage.getItem('identifier')
23     }
24   })
25   .then((response) => {
26     const res = response.data
27     console.log(res)
28     res.access_token && props.setToken(res.access_token)
29     setUsername(res.userName)
30     setPermission(res.authority)
31     setOrders(JSON.parse(res.orders))
32   }).catch((error) => {
33     if (error.response) {
34       console.log(error.response)
35       console.log(error.response.status)
36       console.log(error.response.headers)
37       console.log(error.response.Authorization)
38     }
39   })

```

A `getData` funkció arra szolgál, hogy adatokat nyerjünk ki a Backend-ből. Az `axios` metódus segítségével ebben a részben megadhatjuk, hogy milyen típusú lekérdezést szeretnénk valamint, hogy honnan és elküldhetjük az azonosításra szolgáló fejegységet. Ezután ha sikeresen felvettük a kapcsolatot akkor betölthetjük a kapott adatokat `response` formájában. A `catch` segítségével elkaphatjuk a Backend által visszajelzett hibákat.

```

79   return (
80     <div className="profile-contener">
81       {permission !== null &&
82         <>
83           <h1>Profil</h1>
84           <h2>Üdvözljük {userName}</h2>
85           {permission === 2 &&
86             <div className='profile-list-contaner'>
87               <h2>Rendelések</h2>
88               {orders.map(orders=>{
89                 <div key={orders.id} className="order-item-contaner">
90                   <p>Dátum: {orders.date}</p>
91                   <p>Ügynök neve: {orders.agentName}</p>
92                   <p>Költsége: {orders.pay} </p>
93                   <p>Célpont neve: {orders.targetName}</p>
94                   <p>Célpont tartózkodási helye: {orders.targetLocation}</p>
95                   <p>Célpont leírás: {orders.targetDescription}</p>
96                   <p>Likvidálás oka: {orders.reason}</p>
97                   {orders.status === 0 &&
98                     <p className='fakeButon-green'>Elvégezve</p>
99                   }
100                  {orders.status === 1 &&
101                    <p className='fakeButon-yellow'>Folyamatban</p>
102                  }
103                  {orders.status === 2 &&
104                    <button index={orders.agentId} index2={orders.id} onClick={openRating}>Értékelésre vár</button>
105                  }
106                </div>
107              )}}
108            </div>

```

A Return mezőben adjuk meg azt, amit megjelenít az oldal. Itt nézi meg azt az oldal, hogy a felhasználó be van-e jelentkezve, valamint, hogy milyen jogkörrel rendelkezik és ennek megfelelően tölt be adatokat. Erre a permission változó figyelésével kerül sor. Nézzük meg azt, hogyha ez a változó kettes értéken van. A map funkció segítségével végigmegyünk az orders listán és megfelelő formában kiíratjuk az elemeit. Az oldalon a legtöbb szöveg <p> szintű. De például a gombok azok külön <button> tagban vannak, ez a tag több funkcióval látja el, például onClick. Ebben a részben megadhatunk olyan funkciót, amely a gombra kattintás után fut le.

IX. Back-end

A Backend-hez Python-t, azon belül is a flask csomagot használtuk, mivel ebben már voltak előzetes ismereteink. Mivel már előzetesen kiválasztottuk az SQLite adatbázist és a tapasztalatok szerint a Python kezeli a legegyszerűbben és leghatékonyabban. Valamint a jogkörök kiosztását is nagyon sok csomag segíti. Az autentikációt flask_jwt_extended csomag segítségével valósítottuk meg. Itt meg tudtuk adni azt, hogy egy autentikációs tokennek mennyi legyen

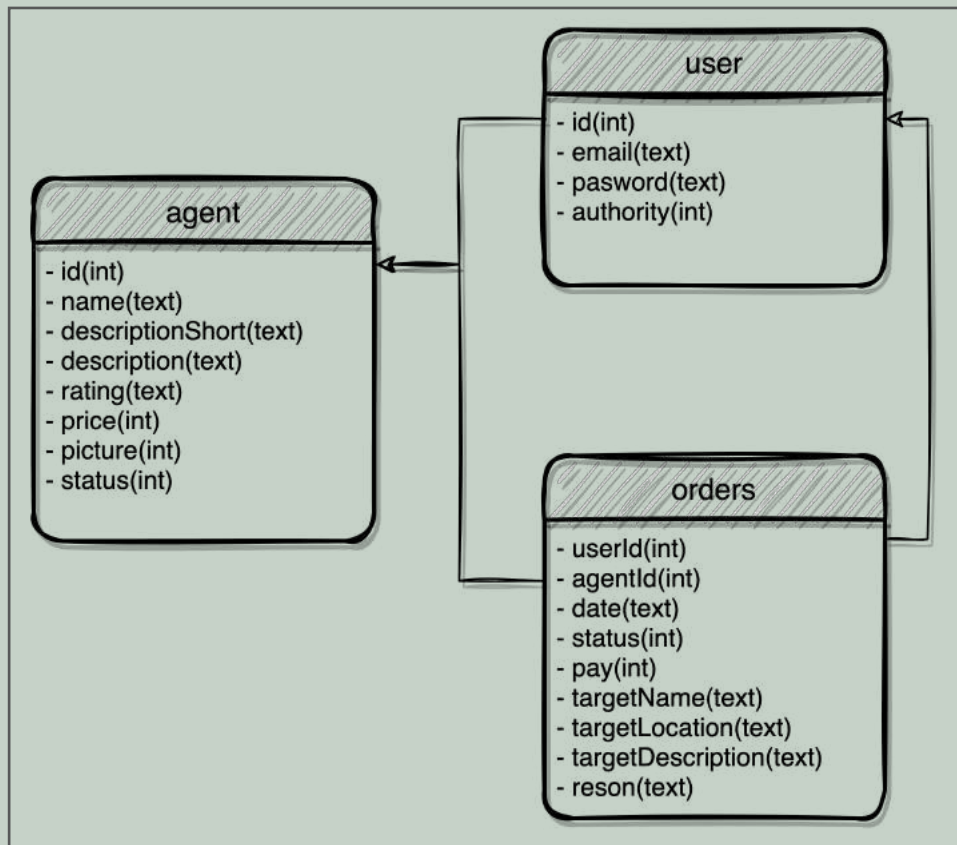
az élettartama, esetünkben ez egy óra. A Flask segítségével úgy nevezett route-okat hoztunk létre. A Backend működését az egyik ilyen routerrel mutatjuk be, amelynek neve agentView.

```

220 @app.route('/agentView', methods=["POST"])
221 @jwt_required()
222 def agent_view():
223     id = request.json.get("id", None)
224     name = ""
225     description = ""
226     rating = ""
227     price = -1
228     picture = -1
229
230     with sqlite3.connect("datas.db") as con:
231         try:
232             cur = con.cursor()
233             cur.execute(f"SELECT * FROM agent WHERE id='{id}'")
234             data = cur.fetchone()
235             if data is None:
236                 return {"msg": "Agent doesn't find!"}
237             else:
238
239         except:
240             print("Server> Agent view problem")
241
242     response_body = {
243         "id": id,
244         "name": name,
245         "description": description,
246         "rating": rating,
247         "price": price,
248     }

```

Az elején a @app.route segítségével megadjuk, hogy ez egy route. Itt megadhatjuk a nevét, valamint, hogy ez milyen típusú ág, jelen esetben ez egy post. Ezután a @jwt_required segítségével megadjuk, hogy erre az ágra csak olyan lekérdező kérdezhet le, akinek megvan a jogosultsága rá. Ezután definiálunk egy funkciót. A funkción belül deklarálunk néhány változót, valamint inicializálunk egyes változókat, például az id-t, ennek request-ből származó adatot adjuk át. Ezután megnyitjuk az adatbázist a megfelelő helyről, a megfelelő néven. Ezen belül try és catch ágak találhatóak, ezekre azért van szükség, hogyha rossz adatot kapnánk akkor ne omoljon össze a szerver. A try ágon belül megírjuk a megfelelő query-t, jelen esetünkben azt, hogy keresse meg azokat az ügynököket akiknek az id-je a megadott. Ezután a kapott értékeket betöltjük a megfelelő változóba. Ha ez nem sikerülne, akkor visszajelzést küld a szerver, hogy valami hiba történt. Végül a response body-ban átadjuk a Frontend-nek a kért, megfelelő adatokat és változókat.



X. Adatbázis

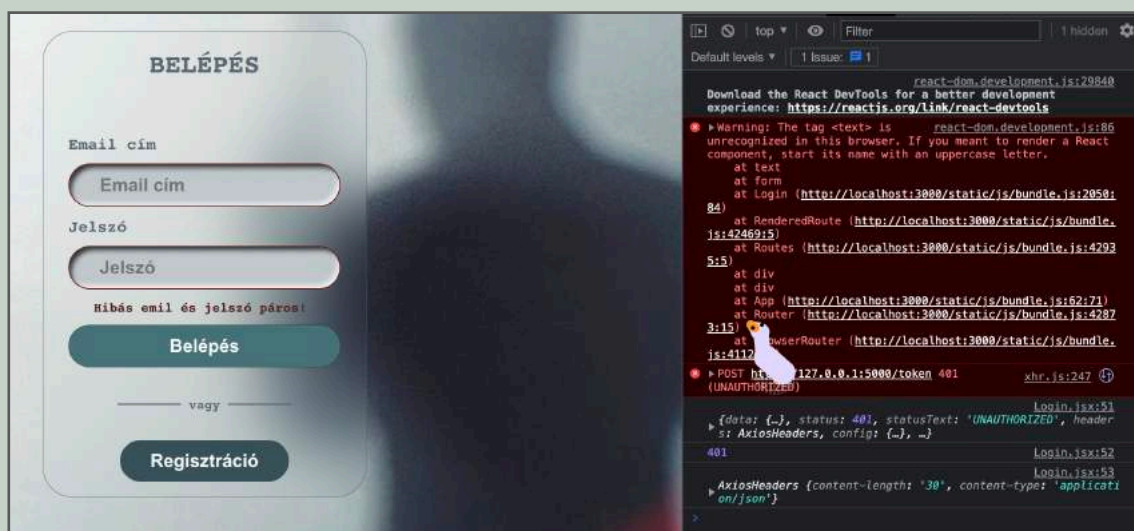
Az adatbázisunkhoz SQLite-ot használtunk azért, mert nem akartunk egy külön serveren futó adatbázist használni. Valamint, azért választottunk SQL típusú adatbázist, mivel a weboldal témájához nem szükséges az adatbázis folytonos bővítése, mert várhatólag nem lesz sok felhasználója (meghatározott réteget érint).

Az adatbázisunknak három táblája van és ezek össze vannak kapcsolva egymással, az egyszerűbb lekérdezések miatt. A legtöbb adat amit tárolunk az vagy szöveg formátum (text), vagy egész szám (int). A user tábla tartalmazza a felhasználókat. Itt egy speciális sor az id különbözteti meg őket egymástól, amely olyan funkcióval van ellátva, hogy nem lehet két egyforma. Az orders tábla csatlakozik a users-hez, minden egyes felhasználónak van egy megrendelések táblája, ebben mentődnek a hozzá tartozó megrendelési adatok. Az agent táblában tárolódnak a kibérelhető ügynökök, minden egyes ügynöknek van egy felhasználója, valamint hozzá tartozó megrendelések/munkák.

XI. Tesztelés

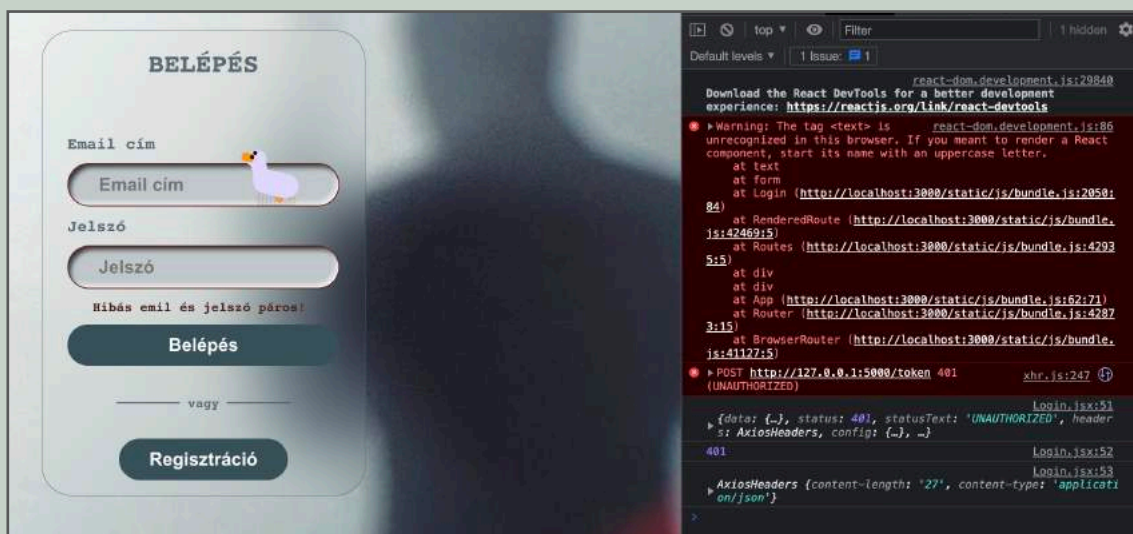
A fejlesztés során kiemelt figyelmet fordítottunk a tesztelésre. Minden egyes komponenst teszteltünk, hogy a megadott működésnek megfelelően funkcionáljon. Ezáltal biztosítottuk a megfelelő működést, valamint a felhasználóélményt is nagyban javítottuk. Az alkalmazás részeit kis csapatunk független tagjai külön is tesztelték, hogy objektíven feltárjuk az esetleges hibákat. A felmerülő hibákról és annak előidézéséről részletes tájékoztatót adtunk a fejlesztést végző csapattagnak. A tesztelés folyamán az alapvető tesztekkel kezdtünk, majd fokozatosan bővültek a tesztek, amelyek a teljes rendszer tesztelését célozták meg. Azért fordítottunk kiemelt figyelmet a fejlesztés során a tesztelés folyamatára, hiszen ez az a rész, amelyek keretén belül megbizonyosodhatunk arról, hogy a szoftverünk megbízhatóan és funkciójának megfelelően működik.

A következőkben néhány képpel szemléltetjük, hogy a tesztelésünk hogyan működik. A legegyszerűbben a belépési felületen tudjuk ezt szemléltetni, a következőkben ezt mutatjuk be. Vettünk három bemenetet a.) rossz felhasználói azonosító és jelszó, b.) jó felhasználói azonosító de a jelszó helytelen, c.) minden bemeneti mezőt helyesen töltöttünk ki.



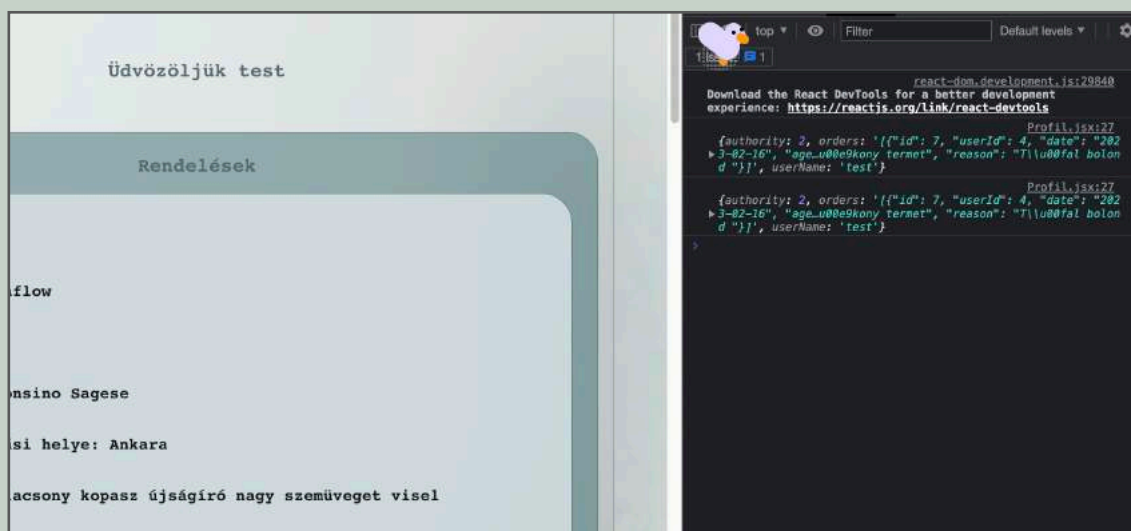
a.) Teszt eset I.

Az első esetben azt szemléltjük, hogy mi van akkor, ha a felhasználó elrontja az azonosítót valamint, a jelszót is. Miután beírta a felhasználó a hibás adatokat és elküldte a szervernek a “Belépés” gombbal. A szerver visszajelzett egy 401 hibakóddal vagyis hibás és/vagy rossz autentikációs kérelem. Ezen a ponton a Frontend is visszajelez a felhasználónak azzal, hogy pirosra váltanak az input mezők keretei. Valamint egy hibaüzenet jelenik meg, a belépés gomb felett.



b.) Teszt eset II.

A második esetben azt szemléltjük, hogy mi történik akkor, ha a felhasználó csak a jelszavát rontja el. Miután beírta a felhasználó a hibás adatot valamint, az egy jót és elküldte a szervernek a “Belépés” gombbal. A szerver visszajelzett egy 401 hibakóddal vagyis hibás és / vagy rossz autentikációs kérelem. Ezen a ponton a Frontend is visszajelez a felhasználónak azzal, hogy pirosra váltanak az input mezők keretei. Valamint, egy hibaüzenet jelenik meg a belépés gomb felett. Vagyis a szerver nem tesz különbséget az I. és a II. Teszt eset között.



c.) Test este III.

Az utolsó esetben vagyis a harmadikban, azt nézzük meg, mi történik akkor, ha a felhasználó mindent jól ír be. Itt a felhasználó miután beírta a helyes jelszó azonosító párost és elküldte azt a szervernek a "Belépés" gombbal. Az rögtön bejelentkezteti a felhasználót, kiosztja neki a megfelelő Tokent és betölti a profil oldalt. Ahol már autentikálja magát és megkapja a megfelelő adatokat.

XII. Hardver követelmények

Mivel az applikációnk egy online felületen működik, ezért nincsen nagy hardver igénye. Minden olyan eszközön elfut, amely stabilan képes futtatni Windows-t, Linux-ot vagy Mac operációs rendszert. Valamint ezek legfrissebb verzióját.

XIII. Szoftver követelmények

Mivel az applikációnak nincs beépített felhasználó felülete, hanem egy böngészőt használ, így elengedhetetlen egy böngésző. Az operációs rendszerben alaphoz beépített is megfelelő. Valamint, mivel a szerver Python alapú így elengedhetetlen a Python a gépre. Az ajánlott verzió a 3.9 .

XIV. SWOT

S

Egyedi
Jártasság a témát illetően

W

Magyarázat szorulhat az
ötlet
Kevés ismerettség a
webfejlesztésben

O

Továbbfejlesztési lehetőség

I

Nem a legnépszerűbb
Backend, ezért nincs sok
tárhely szolgáltató
Mivel egyedi, nincs sablon

XV. Felhasznált szoftverek

Figma - dizájn

Krita - képszerkesztés

Gimp - képszerkesztés

Google drive diák - prezentáció

Visual studio code - frontend

Pycharm - backend

Db browser - adatbázis

Git - verzió kezelés

Pages - dokumentáció szerkesztése

Meta messenger - meetingek, egyeztetés

XVI. Források

Az oldal háttérképe:

<https://www.behance.net/gallery/33816156/HITMAN-World-of-Assassination>

2022.12.20

Ügynök férfi:

<https://www.freepik.com/search?format=search&query=suit%20&selection=1&type=vector>

2022.11.05

Ügynök nő:

<https://www.freepik.com/search?format=search&query=suit%20woman&selection=1&type=vector>

2023.02.20