

SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR,
INFORMATIKA SZAK



SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM

Állatkert - Navigáció

DIPLOMADOLGOZAT

Témavezető:
Györfi Ágnes,
Tanársegéd
Dr. Kupán Pál,
Docens

Végzős hallgató:
Gál Attila

2023

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,
SPECIALIZAREA INFORMATICĂ



UNIVERSITATEA
SAPIENTIA

Grădina zoologică - Navigatie

LUCRARE DE DIPLOMĂ

Coordonator științific:

Györfi Ágnes,
Asistent Universitar

Dr. Kupán Pál,
Conferențiar Universitar

Absolvent:

Gál Attila

2023

SAPIENTIA HUNGARIAN UNIVERSITY OF
TRANSYLVANIA
FACULTY OF TECHNICAL AND HUMAN SCIENCES
COMPUTER SCIENCE SPECIALIZATION



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA


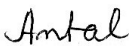



Zoo - Navigation

BACHELOR THESIS

Scientific advisor:
Györfi Ágnes,
Assistant Lecturer
Dr. Kupán Pál,
Associate Professor

Student:
Gál Attila

2023

UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA Facultatea de Științe Tehnice și Umaniste din Târgu Mureș Programul de studii: Informatică		Viza facultății:
LUCRARE DE DIPLOMĂ		
Coordonator științific: dr. Kupán Pál Îndrumător: Györfi Ágnes	Candidat: Gál Attila Anul absolvirii: 2023	
<p>a) Tema lucrării de licență: Grădina zoologică – Navigație</p> <p>b) Problemele principale tratate: Studiul navigației bazat pe GPS și integrarea lui într-o aplicație Documentarea adecvată a stadiilor de proiectare a aplicațiilor.</p> <p>c) Desene obligatorii: Diagrame de proiectare pentru aplicația software realizată.</p> <p>d) Softuri obligatorii: O aplicație de navigare pentru grădina zoologică</p> <p>e) Bibliografia recomandată: NodeJS - https://nodejs.org/en/docs Mapbox - https://docs.mapbox.com/</p>		
<p>f) Termene obligatorii de consultații: săptămânal, preponderent online</p> <p>g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca, Facultatea de Științe Tehnice și Umaniste din Târgu Mureș, sala / laboratorul 414</p> <p>Primit tema la data de: 20.06.2022 Termen de predare: 02.07.2023</p> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>Semnătura Director Departament</p>  <p>Semnătura responsabilului programului de studiu</p>  </div> <div style="width: 45%;"> <p>Semnătura coordonatorului</p>   <p>Semnătura candidatului</p>  </div> </div>		

Declarație

Subsemnatul/a GA'Z ATILA, absolvent(ă) al/a specializării
INFORMATICA, promoția 2020/2023 cunoscând
prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a
Universității Sapienția cu privire la furt intelectual declar pe propria răspundere că prezenta
lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală,
cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de
specialitate sunt citate în mod corespunzător.

Localitatea, Corumca
Data: 2023.06.15

Absolvent
Semnătura.....GAZ.....

Kivonat

A dolgozatom témája egy mobilalkalmazás fejlesztése volt, amely lehetővé teszi a felhasználó számára a könnyed navigációt és az online jegyvásárlást. Mivel napjainkban mindenki siet valahova, ezért fontos, hogy a megfelelő időpontban érkezzen vagy induljon. Ehhez sokban hozzájárulnak a navigációs mobilalkalmazások, amelyekkel meghatározhatjuk a legmegfelelőbb útvonalat a célunk eléréséig.

A tervezett mobilalkalmazás segítséget nyújt a marosvásárhelyi állatkert területén való eligazodásban. Az alkalmazás megoldást biztosít a hosszú sorokban való állás problémájára is. Mivel a felhasználók online előre megvásárolhatják a jegyeket az alkalmazás segítségével. Időt és energiát takaríthatnak meg ezzel a megoldással.

A navigációs funkcióval könnyedén és hatékonyan lehet közlekedni az állatkert területén. Az interaktív térkép segítségével a felhasználók könnyen megtalálhatják a számukra legfontosabb attrakciókat. Továbbá az applikáció segítségével a látogató gyorsan és pontosan eljuthat a kívánt helyekre.

Az online jegyvásárlási funkcióval a felhasználók kényelmesen, akár otthonról is megvásárolhatják a belépőjegyeiket. Ez jelentősen csökkenti a sorban állás időtartamát. A navigáció és online jegyvásárlás funkcionalitások kombinációja hatékonyabb és kellemesebb élményt nyújthat az odalátogatók számára.

Rezumat

Tema lucrării mele de licență este dezvoltarea unei aplicații mobile, care facilitează utilizatorilor navigarea și cumpărarea biletelor online. În zilele noastre toată lumea se grăbește undeva, de aceea punctualitatea este foarte importantă. Din acest punct de vedere aplicațiile de navigare sunt foarte folosite, deoarece determină cea mai corespunzătoare rută spre destinație. Aplicația mobilă propusă ajută în orientarea pe teritoriul Grădinii Zoologice din Târgu Mureș. De asemenea rezolvă problema cozilor la casa de bilete. Utilizatorii pot cumpăra bilete în avans cu ajutorul aplicației. Prin această soluție pot economisi timp și energie. Funcția de navigare a aplicației înlesnește vizita la grădina zoologică. Cu ajutorul hărții interactive utilizatorii pot găsi cu ușurință obiectivele căutate. Prin urmare pot ajunge acolo unde doresc să ajungă. Combinarea funcțiilor de navigare și cumpărare de bilete online asigură o experiență mai eficientă și mai plăcută vizitatorilor Grădinii Zoologice din Târgu Mureș.

Abstract

The theme of my bachelor's thesis is the development of a mobile application which enables the user to easily navigate and buy tickets online. Nowadays everybody is in a hurry, thus it is important to start and arrive on time. Navigation mobile apps are very helpful in this regard, because they determine the most suitable route to our destination. The proposed mobile application helps the user navigate on the territory of the Târgu Mureş Zoo. Furthermore, it solves the problem of standing in line for tickets. Users can buy tickets in advance with the help of the app. This solution will save them time and energy. The navigation function will make visiting easier and more efficient. With the help of the interactive map, users can find the sights that are interesting to them. Consequently, they can get exactly where they want to be. The combination of the navigation and online ticket buying functions offers a more efficient and pleasant experience to the visitors.

Tartalomjegyzék

1. Bevezető	3
2. Dolgozat előzménye	4
3. Felhasználói követelmények	6
3.1. Célkitűzések	6
3.2. Rendszerkövetelmények	7
3.2.1. Funkcionális követelmények	8
3.2.2. Nem funkcionális követelmények	8
4. Piackutatás	9
4.1. Romániában található állatkertek weboldalai	9
5. Felhasznált technológiák	11
5.1. React Native	11
5.2. Mapbox	11
5.3. Express.js	12
5.4. MongoDB	12
5.5. Felhasznált Node.js modulok	13
6. Rendszer leírása	15
6.1. A rendszer architektúrája	15
6.1.1. MERN architektúra	15
6.1.2. Komponens alapú architektúra	16
6.2. Adatbázis felépítése	17
6.3. Use Case diagramm	18
6.4. Szekvencia diagramok	19
6.4.1. Regisztráció	19
6.4.2. Bejelentkezés	20
7. Alkalmazás ismertetése	21
7.1. Kezdőlap	22
7.2. Regisztráció	23
7.3. Bejelentkezés	24
7.4. Térkép	25
7.5. Térkép útvonal kirajzolása	26
7.6. Jegyvásárlás	27

7.7. Megvásárolt jegyek listája	28
7.8. Online fizetés	29
8. Továbbfejlesztési lehetőségek	30
Összefoglaló	31
Köszönetnyilvánítás	32
Irodalomjegyzék	34

1. fejezet

Bevezető

Napjainkban az állatkertek világszerte népszerű látványosságok, amelyek izgalmas és tanulságos élmény nyújtanak az odalátogató gyerekeknek és felnőtteknek is egyaránt. Azonban a hatalmas területeken elterülő állatkertekben néha gondot okoz az eligazodás és a jegyvásárlás gyakran időigényes és bosszantó folyamat lehet. Az emberek hosszú sorokban állnak, hogy megvegyék a belépőjegyüket, és időt pazarolnak azáltal, hogy keresik a kívánt állatokat a területen. Mindazonáltal egy ilyen fajta kikapcsolódás nem szabadna egy sietős időtöltés legyen, azonban néha kevés idő áll rendelkezésére a látogatóknak.

A digitális korban azonban új lehetőségek nyílnak meg az állatkerti élmények terén vagy bármilyen ekkora volumenű létesítményben. Az állatkertnavigációs alkalmazások lehetővé teszik, hogy könnyedén megtaláljuk a kívánt egyed tartózkodási helyét.

Az általam fejlesztett „Zoo - Navigation” nevű alkalmazás célja, hogy megkönnyítse az állatkertbe látogatók számára a navigáció és a jegyvásárlást, ezáltal időt és energiát spórolhatnak meg.

A „Zoo - Navigation” alkalmazás egy könnyen használható felületen keresztül segít az látogatóknak az adott faj pontos helyének a megtalálásában. Az alkalmazás térképe piktogramok segítségével mutatja az állatok pontos tartózkodási helyét. Ezen felül az alkalmazás lehetőséget biztosít az online jegyvásárlásra is, így könnyedén elkerülhetjük a hosszú várakozási sorokat. Ez az alkalmazás a Marosvásárhelyen található állatkert területén való eligazodásban segít.

A dolgozatomban részletesen bemutatom a mobilalkalmazás tervezési folyamatait, beleértve a dolgozat előzményét, a felhasználói felület tervezését, a funkcionális követelményeit, piackutatást, a felhasznált technológiákat, a rendszer leírását és az alkalmazás működését.

Dolgozat előzménye

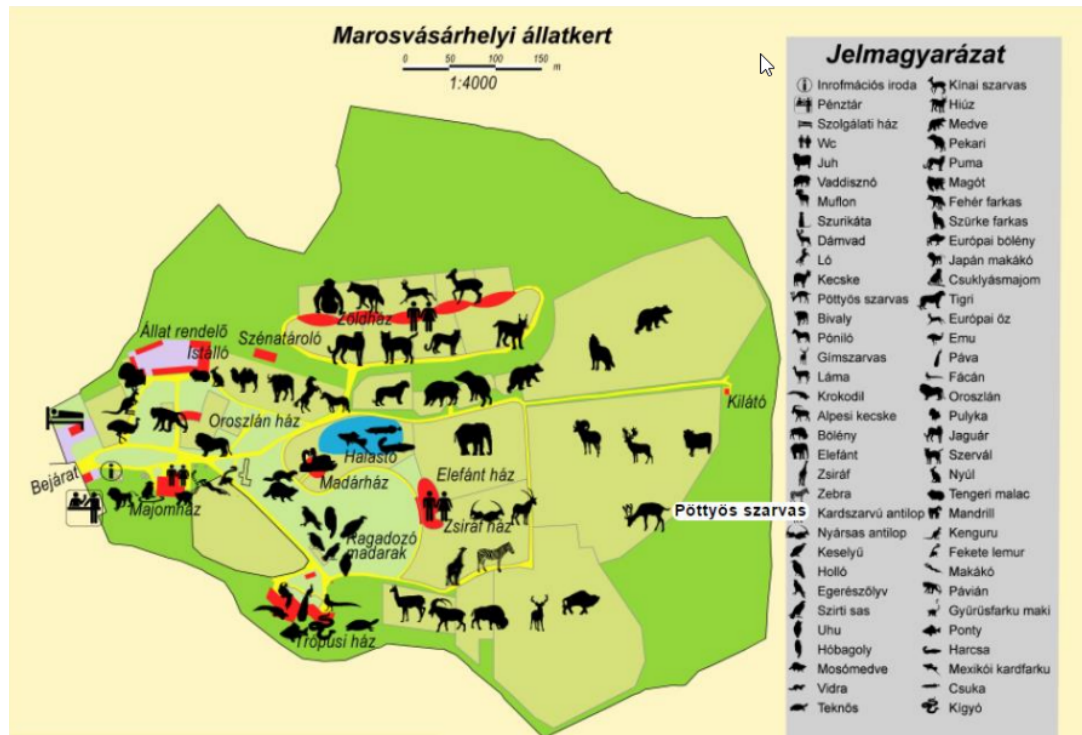
Az ötlet ennek az alkalmazásnak a megvalósításához egy előzőleg megírt államvizsga dolgozatomból származik. A dolgozatom célja a Marosvásárhelyen található állatkert felmérése és egy térkép készítése volt. A felméréshez egy geodéziai GPS-t használtam. Ezek a GPS-ek akár centiméter pontosak is lehetnek megfelelő körülmények között.

A felmérésből összegyűlt adatokból egy térképet szerkesztettem (lásd 2.1 ábra). Ezeket az adatokat egy ZwCAD nevű programban dolgoztam fel. A térkép szerkesztését OCAD szoftver segítségével valósítottam meg, ehhez a munkához társult még egy interaktív statikus térkép is, amelyet egy weblap formájában jelenítettem meg (lásd 2.2 ábra). Mivel akkoriban minimális tudásom volt a weblapfejlesztés területén, ezért csak épp szemléltetésként tettem bele a dolgozatomba.



2.1. ábra. Marosvásárhelyi állatkertről készült papír térkép

A dolgozatomban részleteiben foglalkoztam az állatkertek bemutatásával, történelmével, valamint a régi vagy akkori térképek ismertetésével. Emellett alaposan megvizsgáltam a Romániában található állatkertek webtérképeit. Kiemelten tárgyaltam a térképszerkesztés módszereiről és azok különböző fázisairól is.



2.2. ábra. Marosvásárhelyi állatkertről készült interaktív statikus web térkép

3. fejezet

Felhasználói követelmények

3.1. Célkitűzések

Az általam készített alkalmazás fő céljaként a navigációt emelném ki. Mivel napjainkban nagyon sok helyen használjuk például a Google Maps által nyújtott navigációt, ezért nagyon hozzászoktunk az ilyen fajta alkalmazások használatához. Ennek következtében nagyon sok ember már nem is ismeri a térkép segítségével való tájékozódást és ennek a megkönnyítésére is jött ez az ötlet. Az applikáció rendelkezni fog más funkciókkal is, mint például az online jegyvásárlás. Mindezek mellett egy weboldalt is megszeretnék valósítani, amely az állatkertben dolgozó emberek munkáját könnyítené meg.

Elsődleges célnak, hogy a mobilalkalmazás alapfunkciói működjenek. A következőket szerettem volna megvalósítani:

- Adatbázis létrehozása és csatolása a szoftverhez
- Regisztrálás
- Bejelentkezés
- Jelszó változtatás
- Térkép megjelenítése és szerkesztése
- Navigáció fejlesztése
- Jegyvásárló felület létrehozása
- Online fizetési lehetőség implementálása
- Megvásárolt jegyek tárolása, megjelenítése

Az alkalmazás második céljaként egy webes felület megvalósítását tűztem ki célul, a következő funkciókat az átlagos felhasználók számára tervezem:

- Kommunikáció ugyan azzal az adatbázissal
- Hírportál / Kezdőfelület
- Regisztrálás

- Bejelentkezés
- Jelszó csere
- Jegyvásárlás
- Online fizetési felület
- Jegyek listázása

Az alkalmazás harmadik céljaként a webes felületbe egy olyan platformot szeretnék még létrehozni, ami az ott dolgozók munkájában segíthet:

- Hírportál / Kezdőfelület
- Bejelentkezés
- Regisztráció
- Jelszó csere
- Felhasználók menedzselése
- Statisztikai felület a bevételekről, kiadásokról, odalátogatókról stb.
- Dolgozók adatainak a tárolásáról, fizetéséről stb.
- Az állatok mindenikéről részletes leírás, információ minden egyedről külön-külön
- Eseményrendezői felület, állatkerti séták, túrák tervezése stb.

3.2. Rendszerkövetelmények

A rendszerkövetelmények egy adott alkalmazás, program, játék működéséhez szükséges minimális hardver és szoftverkövetelmények. Ezeket általában közzéteszik az alkalmazások letöltési oldalán. Egy adott alkalmazás rendszerkövetelményei eltérőek lehetnek, ha különböző operációs rendszereken használjuk őket. A követelmények között fel vannak tüntetve az operációs rendszerek verziószámai a memória, tárhelyigény, valamint a processzor típusát és sebességét is feltüntetik. A rendszerkövetelmények két nagy csoportra oszthatóak fel, funkcionális és nem funkcionális követelmények.

3.2.1. Funkcionális követelmények

A funkcionális követelmények közé tartoznak azok a funkciók, amelyek megfelelnek az alkalmazás működési kritériumainak. Azokra a funkcionalitásokra gondolok itt, amelyeket az általam fejlesztett alkalmazás el tud végezni.

- **Regisztráció**

Az alkalmazást bárki használhatja, csupán csak egy regisztrációra van szüksége. A regisztrációhoz a felhasználó meg kell adjon néhány információt, mint például: név, email cím és jelszó.

- **Bejelentkezés**

Az alkalmazásba való bejelentkezést egy „Login” gombbal valósítjuk meg. A gombra kattintva a regisztrált felhasználók email címét és jelszavát kéri az alkalmazás. Miután mindezeket megadta már meg is történt a bejelentkezés.

- **Kezdőoldal**

Amint bejelentkeztünk egy „Home” oldalra kerülünk, ahol két lehetőség közül választhatunk. A térképet jeleníthetjük meg vagy a jegyvásárló felületet.

- **Térkép**

Ahhoz, hogy a térképet megjelenítsük kötelező a regisztráció és a bejelentkezés. A „map” gombra kattintva az alkalmazás az állatkert területére közelít és megmutatja a pontos tartózkodási helyünket. A térképen a piktogramok segítenek az adott faj kiválasztásában. A képre kattintva kirajzolódik egy útvonal az adott faj megközelítéséhez.

- **Jegyek**

A jegyek vásárlását egy „ticket” nevű gombbal lehet elérni. A gombra kattintva megjelenik az összes jegy.

- **Megvásárolt jegyek**

A megvásárolt jegyek listázását „bookings” nevű gombbal lehet elérni. A gombra kattintva megjelenik az összes jegy, amit előtte vásároltunk. .

3.2.2. Nem funkcionális követelmények

Az alkalmazás azon tulajdonságairól beszélünk, amelyek nem a funkcionalitással kapcsolatosak, hanem a rendszer működését, teljesítményét vagy minőségét befolyásolják. Ezeket a követelményeket nem mérhetjük úgy, mint a funkcionális követelményeket, de fontosak a rendszer sikeressége és elfogadása szempontjából.

Például a felhasználói élmény javítása, a rendszer megbízhatósága, a teljesítmény, a biztonság, a skálázhatóság, az elérhetőség vagy akár az alkalmazásra vonatkozó jogi és szabályozási követelmények lehetnek nem funkcionális követelmények.

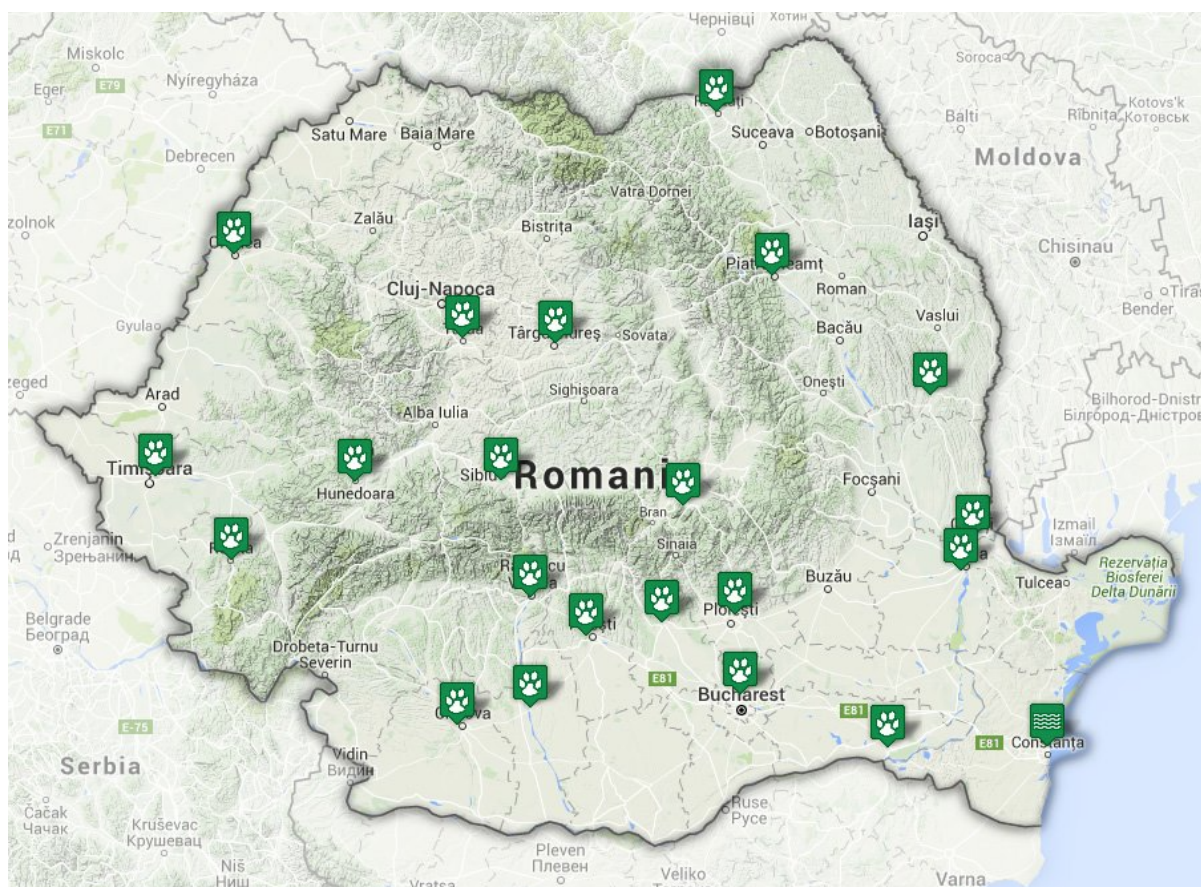
Tehát mindkét fajta követelményre szükségünk van egy szoftver megtervezése és fejlesztése miatt. A sikeres munka miatt minkettőt figyelembe kell venni.

4. fejezet

Piackutatás

4.1. Romániában található állatkertek weboldalai

Az előző dolgozatomban végeztem egy kutatást a Romániában található állatkertek webtérképeiről. A "Federația Grădinilor Zoologice și Acvariilor din România" szövegtség honlapján található interaktív térképet használtam fel, ahol minden egyes létesítmény fel van tüntetve. (lásd 4.1 ábra)



4.1. ábra. Romániában található állatkertek és delfináriumok elhelyezkedése

Ezt a fényképet a jelenlegi Facebook oldalukon találtam. A weboldaluk már nem működik. A térképen összesen 21 állatkert és 1 delfinárium található. A 2017-ben összegyűjtött weboldalak címek közül egyik sem működik, ezért egy új kutatásra volt szükség.

Város	Weboldal	Applikáció
Marvásárhely	https://www.zootirgumures.ro/	nincs
Brassó	https://zoobrasov.ro/	nincs
Nagyvárad	https://zooradea.ro/	nincs
NagySzeben	https://www.zootirgumures.ro/	nincs
Bukarest	https://bucurestizoo.ro/	nincs
Resicabánya	https://zooresita.ro/	nincs
Călărași	https://zoo.primariacalarasi.ro/	nincs
Pitești	-	nincs
Ploiești	https://zooploiesti.ro/	nincs
Râmnicu Vâlcea	-	nincs
Târgoviște	https://www.zootargoviste.ro/	nincs
Galați	-	nincs
Brăila	-	nincs
Rădăuți	-	nincs
Gyulafehérvár	-	nincs
Târgu Neamț	-	nincs
Craiova	-	nincs
Temesvár	-	nincs

4.1. táblázat. Romániai állatkertek weboldalai

Ez uttal az állatkertek weboldalait vizsgáltam és azoknak a működését, továbbá utána néztem a mobilos alkalmazásoknak is. A weboldalak ellenőrzése mellett mobilos alkalmazásokat is kerestem. A nagyszebeni állatkertnek volt egy alkalmazása, de sajnos már nem működik. A nagyobb városok turisztikai alkalmazásaiban van némi információ az ott található állatkertekről, de egyiknek sincs mobilos applikációja.

5. fejezet

Felhasznált technológiák

A projekt kivitelezése során több technológiát is alkalmaztam. A frontend részben a React Native nyílt forráskódú keretrendszert használtam, míg a backend részhez a Node.js Express.js keretrendszerét választottam. A térkép megjelenítéséhez pedig a Mapbox GL JS Javascript könyvtárat alkalmaztam. Az adatbázis terén a MongoDB-t választottam, mivel a Node.js moduljai könnyedén kezelik a JSON adatokat.

5.1. React Native

A React Native-ot mobilalkalmazások fejlesztésére használják. Használata során a fejlesztők Javascript vagy TypeScript készítik az alkalmazást. Számos előnye közül kiemelném azt, hogy az elkészített komponensek újra használhatók iOS és Android felületeken. Gyorsabb fejlesztést és iterációt biztosít, az elemeket könnyedén fel lehet építeni és tesztelni a munka során. Azonnal láthatóak ezek az elemek ez sokban segít a hibakeresésben és a pontosabb fejlesztésben. [\[Fac\]](#)

5.2. Mapbox

A Mapbox egy olyan vállalat, ami térképes hely adatokat biztosít a térképalkalmazás fejlesztők számára. A mapbox segítségével könnyedén létre tudunk hozni saját személyre szabott térképeket az alkalmazásainknak. Nagyon sok API-t és SDK-t nyújt a helyadatok és térképek kezeléséhez. [\[Gun\]](#)

Mapbox GL JS egy olyan Javascript könyvtár, amely webtérképek és webalkalmazások építésére használnak. Ezzel a könyvtárral könnyedén megjeleníthető egy térkép az alkalmazásunkba. Rengeteg különböző elemet tehetünk rá a térképünkre. Számomra a navigációhoz volt szükségem erre a könyvtárra. Több vállalat termékét is megvizsgáltam, de ez volt a legkézenfekvőbb az én applikációm kivitelezéséhez. Mivel nem minden cég adatai között volt meg az állatkert összes útvonala, ezért az is befolyásolta a döntésem. A mapbox több, mint 50,000 kérést is biztosít ingyen. Az én esetemben: „Vector tiles API: 200,000”, „Map Loads for Web: 50,000”, „Direction API 100,000” free request-et biztosít.

5.3. Express.js

Az Express.js egy Node.js webalkalmazás fejlesztéséhez használt keretrendszer. [Hol] Gyors szerveroldali fejlesztést biztosít. Könnyedén és hatékonyan lehet API-kat építeni Node.js környezetben. Használata könnyű és rugalmas. Egyszerű módszert kínál az útvonalak, middlewarek és HTTP kérések és válaszok kezelésére. A middlewarek fontos elemei az Express.js-nek, használatukkor lehetőségünk van a következőkre:

- Végrehajtani a műveleteket az érkező kérés előtt vagy után
- Módosítani a kérés vagy válasz objektumot
- Megszakítani a feldolgozási láncot és visszatérni egy válasszal
- Továbbítani a vezérlést a következő middleware számára „next()” utasítással

A middlewarek nagyon rugalmasak és testre szabható függvények. Több middlewaret is elhelyezhetünk egymás után, így lépésről-lépésre dolgozzuk fel a kéréseket, ez által könnyedén módosíthatjuk az adatokat és adhatunk hozzá új funkciókat. [Schb]

5.4. MongoDB

A MongoDB egy nyílt forráskódú dokumentumalapú NoSQL adatbázisrendszer. [DMR] JSON alapú dokumentumok tárolására és kezelésére szolgál. MongoDB főbb tulajdonságai a következők:

- **Rugalmas adatmodell:** A MongoDB dokumentumokat tárol, amelyek JSON-szerű kulcs-érték párok gyűjteményei.
- **Skálázhatóság:** A MongoDB skálázható adatbázisrendszert biztosít, ami azt jelenti, hogy képes megbirkózni a nagy adatmennyiségekkel és a magas forgalommal
- **Kiemelkedő teljesítmény:** A MongoDB optimalizált adatkezelést és gyors lekérdezéseket nyújt. Indexeket használ a hatékony adatkereséshez, és támogatja az aggregációkat és a lekérdezési optimalizációt a gyors adatelérésekhez.
- **Kifejezetten webes alkalmazásokhoz tervezve:** A MongoDB kiválóan alkalmas webes alkalmazásokhoz, különösen a skálázható és változó adatokkal rendelkező projektekhez. A könnyű adatmodell és az elosztott adatbázis lehetővé teszi a gyors és hatékony adatkezelést.
- **Adatintegritás:** A MongoDB támogatja a tranzakciókat és az adatintegritást, lehetővé téve az adatok biztonságos és megbízható kezelését. Biztosítja az adatok konzisztenciáját és a hibatűrő adatbázis-rendszerét.
- **Kiterjeszthető funkcionalitás:** A MongoDB kiterjeszthető az egyedi igényekhez és funkciókhoz. Számos beépített funkcióval rendelkezik, mint például a geopozíció-alapú lekérdezések, teljes szöveges indexelés, grafikus adatok kezelése stb.

- **Könnyű kezelhetőség:** A MongoDB rendelkezik egy intuitív parancssoros felhasználói felülettel (mongosh), valamint számos nyelvet támogató meghajtókat és könyvtárakat biztosít a fejlesztési környezethez.

5.5. Felhasznált Node.js modulok

Mivel a Node.js nagy moduláris környezettel rendelkezik ezért a fejlesztés során könnyen beimportálhatunk, telepíthetünk olyan külső modulokat, amelyek sokban megkönnyítik a munkánkat. Továbbá még saját modulokat is készíthetünk magunknak. A modulok egyszerű kezelését a "Node Package Manager" teszi lehetővé.

- **Nodemon**

A nodemon szerepe, hogy megkönnyítse a fejlesztők munkáját azzal, hogy automatikusan újraindítja a futó alkalmazást, amikor változást észlelnek a forráskódban. Ez jelentősen megkönnyíti a fejlesztést, mert nem kell minden alkalommal manuálisan újraindítani az alkalmazást. Mivel én a Visual Studio Code szerkesztőben dolgoztam, a nodemon minden mentés után automatikusan újraindította az alkalmazást. Ez nagymértékben segítette a munkámat, mert azonnal visszajelzést kaptam, ha hibát követtem el. A nodemon konfigurálható például arra, hogy milyen fájl típusokat figyeljen meg. Emellett támogatja a "hot-reloading" funkciót is, amely csak az adott változásokat hajtja végre a kódban. [Scha]

- **Bcryptsjs**

Úgyszintén egy JavaScript könyvtárról beszélünk, amelyet azért hoztak létre, hogy biztonságos legyen a jelszavak tárolása és ellenőrzése. Ez a könyvtár a Bcrypt algoritmust implementálja, amely a jelszavak hashelését biztosítja. [Wir]

- **Cookie-parser**

A "Cookie-parser" middleware module a Node.js-hoz. Gyakran használják webalkalmazásokban, hogy kezeljék a http sütiket. A süti adatfájlok, amiket a weboldalak a felhasználók böngészőjében helyeznek el, hogy bizonyos információkat tároljanak. A „cookie-parser”-el könnyen olvashatjuk és írhatjuk a sütiket a Node.js alkalmazásokban. [npma]

- **Dotenv**

Ezt a csomagot a konfigurációs fájlok kezelésére használják. Lehetővé teszi, hogy a konfigurációs változókat „.env” fájlban tároljuk el. Miután elindul az alkalmazás automatikusan beolvassza ezeket a változókat. A fájl kulcs-érték párokat tartalmaz. [npmb]

- **Jsonwebtoken**

Más néven „JWT” úgyszintén egy npm csomag, amely segítségével JSON Web Tokenek-et lehet kezelni Node.js alkalmazásokban. A JSON Web Token (JWT) az adatok biztonságos átvitelére használják a felek között. Ez egy digitális aláírási adatstruktúra. [npmc]

- **Mongoose**

„Object-Document Mapping” könyvtár, amellyel könnyedén tudjuk használni a MongoDB adatbázist. Ennek a segítségével az alkalmazás JavaScript objektumként tudja kezelni és manipulálni a dokumentumokat. A mongoose használatával létrehozott „Schemak” meghatározzák az adatbázisban eltárolt dokumentumok struktúráját. Ezek a sémák meghatározzák a mezők típusát, alapértelmezett értékeit, tulajdonságait.[Kar]

- **Slugify**

A slugify module egy olyan folyamatot hajt végre, amely átalakít úgy egy szöveget, hogy az alkalmas legyen URL-címekben, fájlnevekben vagy más helyeken. Általában kisbetűssé teszi a szöveget, eltávolítja a felesleges szóközöket és kicseréli a speciális karaktereket. Hasznos eszköz a webes alkalmazásokban vagy más helyeken is, ahol egységes és URL-barát szövegeket szeretnénk generálni.[npmd]

- **Stripe**

A Stripe egy online fizetési platform, amely fejlesztők számára nyújt könnyen használható API-t. Az API-t könnyedén be tudják építeni a fejlesztők a weboldalukba vagy alkalmazásaikba. Így egyszerűen kezelhetőek az online fizetések és tranzakciók. [CC]



5.1. ábra. Stripe logó

Az API a következőket nyújtja a fejlesztőknek: Különböző típusú fizetések kezelését, mint például hitelkártya- és banki átutalások, biztosítva a biztonságos és titkosított fizetési folyamatokat.

Ügyfél- és felhasználóadatok tárolását, lehetővé téve az egyszerűbb ügyfélélmény fejlesztését.

Ismétlődő fizetések, mint előfizetések, havi díjak vagy tagsági költségek könnyű beállítását és kezelését, csökkentve az adminisztrációs feladatokat.

Értesítéseket a fizetési tranzakciókról, például sikeres vagy sikertelen tranzakciókról. Webhookok beállításával automatikus értesítések küldése a kiválasztott eseményekről.

Tesztelést a fizetési folyamat lokalizált környezetben, valamint részletes analitikai adatokat a tranzakciók és ügyfélviselkedés nyomon követéséhez.

6. fejezet

Rendszer leírása

A következő fejezetben az alkalmazás működésének és felépítését szeretném ismertetni, melynek a könnyebb megértéséhez ábrákat, diagrammokat is felhasználtam. Ezek az elemek szemléltetni fogják az alkalmazás strukturális és funkcionális tulajdonságait. A fejezetben szó lesz a MERN és komponens alapú architektúráról, adatbázis felépítéséről és a diagrammjáról, Use case és Szekvencia diagrammokról.

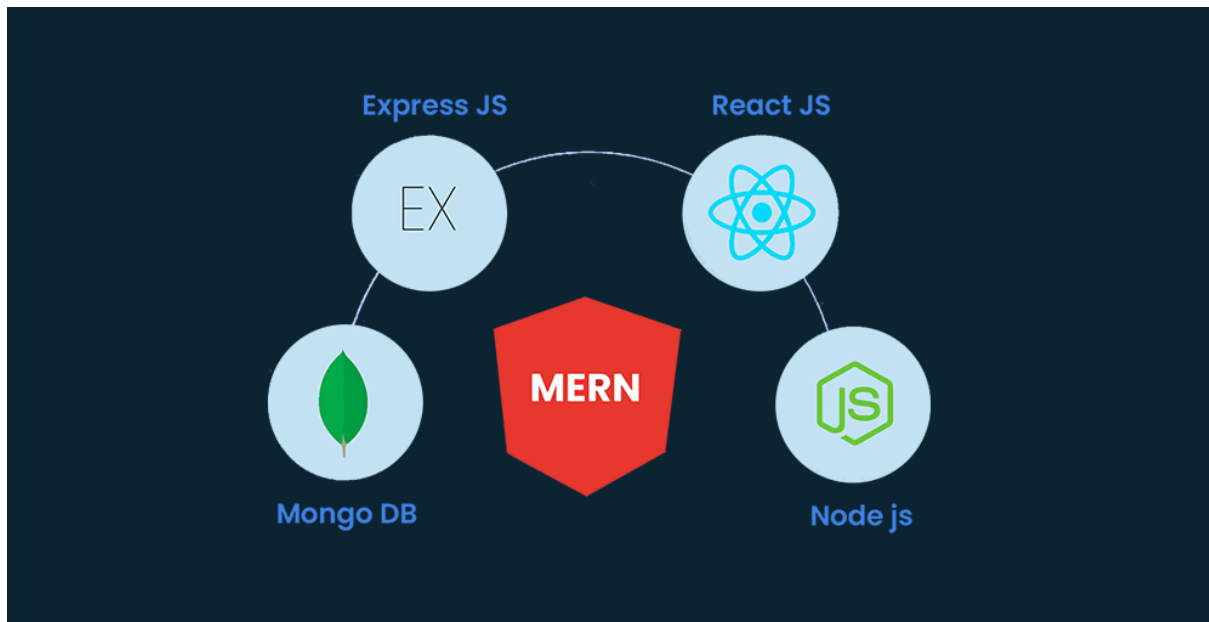
6.1. A rendszer architektúrája

Az alkalmazás két féle architektúrára épül, az egyik a MERN a másik pedig a komponens alapú architektúra.

6.1.1. MERN architektúra

A projektem kliens-szerver oldali architektúrára épül. Ez a fajta architektúra eléggé elterjedt az internetes alkalmazások, mobilalkalmazások körében. Hatékonyabb és skálázhatóbb rendszerek kialakításában segít valamit biztosítja a kliensek és a szerverek elkülönített fejlesztését és karbantartását. Pontosabban a "MERN" stack (MongoDB, Expressjs, React (Native), Node.js) [Nqu] teljes verziójú webalkalmazás architektúra, amely négy főkomponenst kombinál (lásd 6.1 ábra).

A React Native egy keretrendszer, amellyel mobilalkalmazásokat fejleszthetünk React alapú komponensek és technológiák felhasználásával. A MERN stackhez hozzáadva a React Native-ot, ő fogja képviseli a kliensoldali réteget a mobilalkalmazásban. Tehát ezzel a kombinációval fejleszthetünk webalkalmazásokat a böngészőhöz és keresztplatformos mobilalkalmazásokat React Native keretrendszerrel. Ez lehetőséget nyújt arra, hogy az alkalmazásunk üzleti logikáját, adatbázisát és az API-t használhatjuk webes és mobilalkalmazások fejlesztése során, ami egy költséghatékony megoldást eredményezhet.



6.1. ábra. MERN architektúra

6.1.2. Komponens alapú architektúra

A dolgozatomban még használok egy másik fajta architektúrát. React Native és MapboxGL kombinációjából létrehozhatunk egy olyan architektúrát (lásd 6.2 ábra), amely segítséget nyújt a térképekkel kapcsolatos funkciók beépítésében a mobilalkalmazásunkban. Komponens alapú architektúrának nevezik a React Native és MapboxGL kombinálásával létrejött szerkezetet. Nagyon sok képernyő elemet, mint például: térkép, gombok, vektorok, jelölők komponensekként vannak felépítve. Ezeket nagyon könnyen lehet kezelni a kódban, s mindeniket többször is fel lehet használni.

A MapboxGL biztosítja a térképhez kapcsolódó adatok integrálását és megjelenítését, azonban az alkalmazáshoz más adatforrásokat is felhasználhatunk, mint például az API-t vagy adatbázist. A megfelelő komponensekben az adatokat tudjuk tárolni és kezelni.



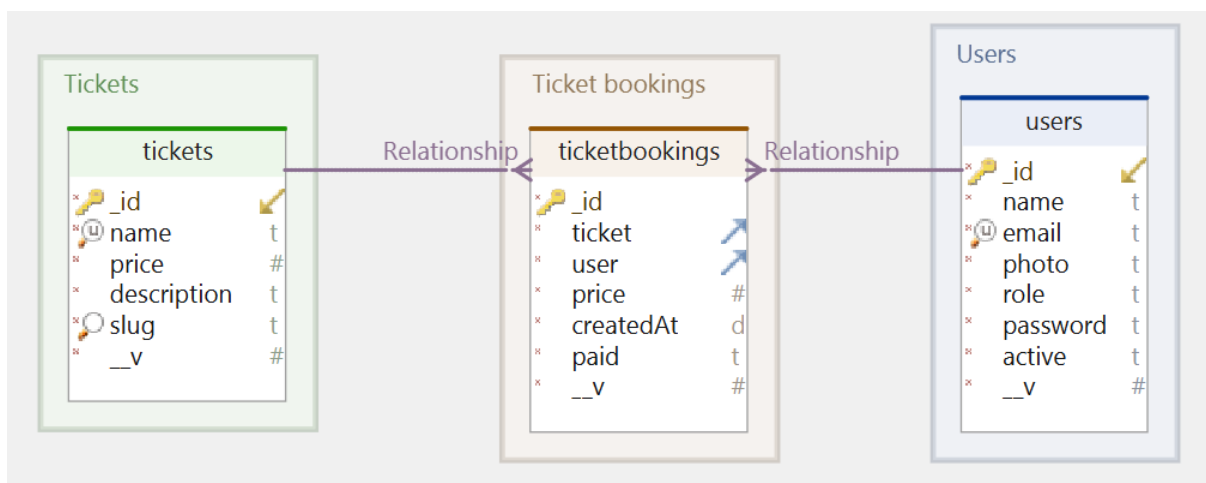
6.2. ábra. React Native és MapBox logók

6.2. Adatbázis felépítése

Az adatbázisom a MongoDB által biztosított MongoDB Atlas felhő adatbázis kezelőjével fejlesztettem. Az Atlas lehetővé teszi az adatbázis hostingot a felhőben, így könnyen elérhető és skálázható adatbázisinfrastruktúrát biztosít a projekt számára. Egyszerű a csatlakozás és kommunikáció az felhőben levő adatbázissal.

Az adatbázisban három gyűjtemény hoztam létre: „User”, „Ticket”, „TicketBooking” (lásd 6.3 ábra).

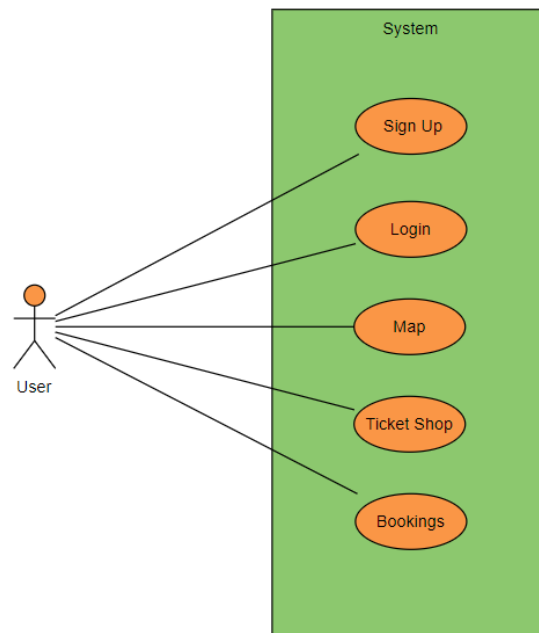
- „User” séma (schema): a felhasználók adatait tárolja. A séma a következő mezőket tartalmazza: név, email, szerepkör, jelszót stb. A jelszavak biztonságos tárolása érdekében bcryptj modult használtam a jelszavak hashelésére.
- A „Ticket” séma a jegyek adatait tárolj, mint például: név, ár és leírást. A séma még tartalmaz egy hook-ot amelyet a mentés előtt generál egy slug értéket a Ticket nevéből, hogy könnyedén használható URL-et lehessen létrehozni.
- A „TicketBooking” sémának a jegyfoglalásban van szerepe. A séma a jegyfoglalás adatait tárolja, mint például a jegy és felhasználóra való hivatkozást, a jegy árát, a jegyvásárlás létrehozásának a dátumát és a fizetési státuszt. A séma tartalmaz egy hook-ot ami a lekérdezés előtt összekapcsolja a „TicketBooking” objektumot a hozzá tartozó Ticket és User objektumokkal.



6.3. ábra. Adatbázis diagram

6.3. Use Case diagramm

A következő fejezetben a Use Case diagramot fogom bemutatni. (lásd 6.4 ábra) Ezen diagramok segítségével szokták ábrázolni a rendszer és a felhasználó közötti interakciókat és funkciókat. Az aktorok és felhasználási esetek jelképezik a résztvevőket és a rendszerben elérhető műveleteket. Ez segít a rendszer funkcionalitásának és viselkedésének áttekintésében, valamint a kommunikációban a projekt résztvevői között



6.4. ábra. Use Case Diagram

Az alkalmazásnak egy típusú felhasználója van. Ez egy egyszerű User-t jelent, vagyis bárki aki regisztrál az normál felhasználói engedélyekkel fog rendelkezni.

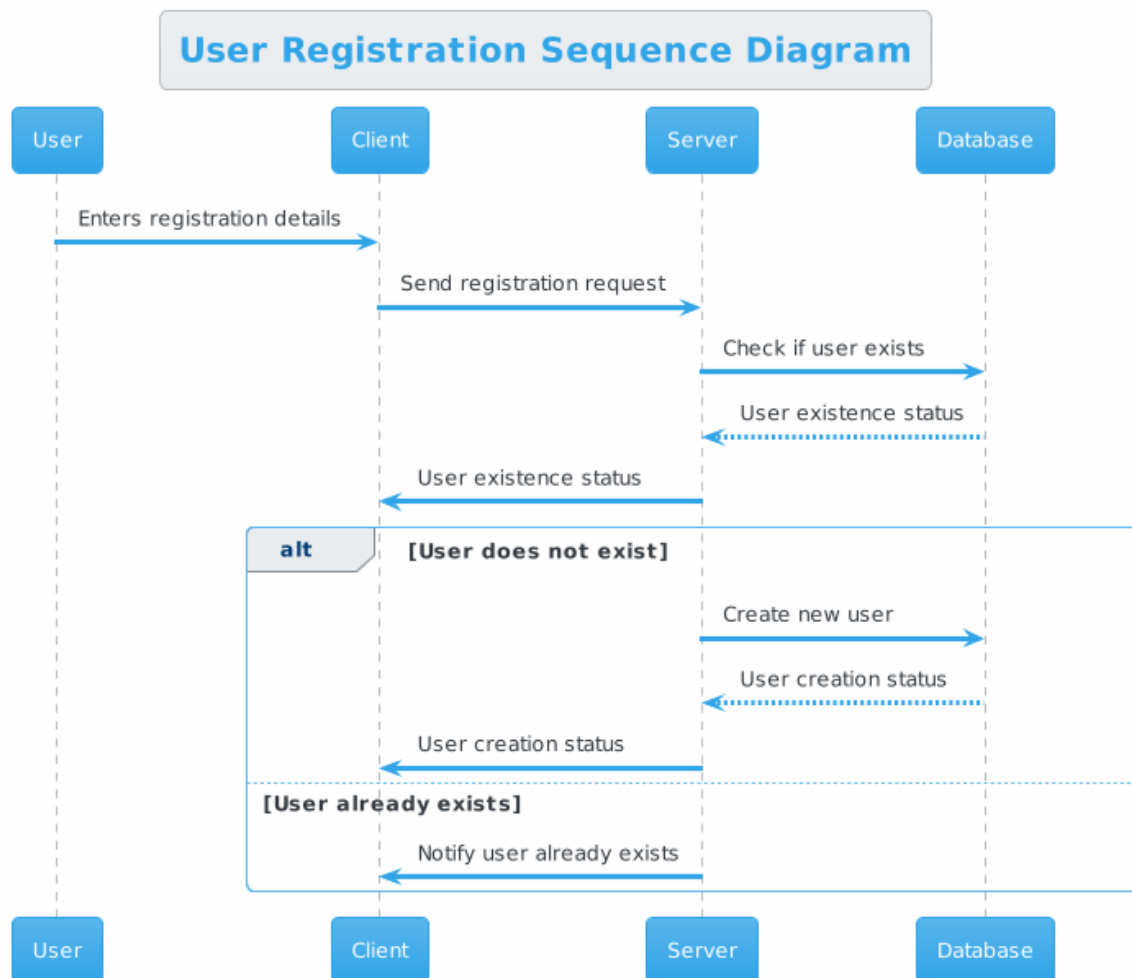
A felhasználó a következő funkcionalitásokat használhatja:

- Regisztráció: a felhasználó létrehozhat egy új fiókot, ha a regisztráció során kért megadott adatok helyesen. A következő adatokat kell kitöltsen regisztrációkor a felhasználó: név, email, jelszó. Nagyon könnyen és egyszerűen regisztrálhat bárki.
- Bejelentkezés: a felhasználó a létrehozott fiók után bejelentkezhet. A bejelentkezés után több új funkciót is elérhet. Láthatóvá válik számára három új menüpont, a „Map”, „Ticket Shop” és a „Bookings”
- Map: amint a felhasználó belépett az elérhetővé vált térképet is rögtön használatba veheti. A térkép az állatkert vonzáskörzetéhez van rögzítve.
- Ticket Shop: a felhasználó könnyen vásárolhat jegyet ebben a menüpontban
- Bookings: a felhasználó ebben a menüpontban tudja igazolni, hogy milyen jegyeket vásárolt

6.4. Szekvencia diagramok

6.4.1. Regisztráció

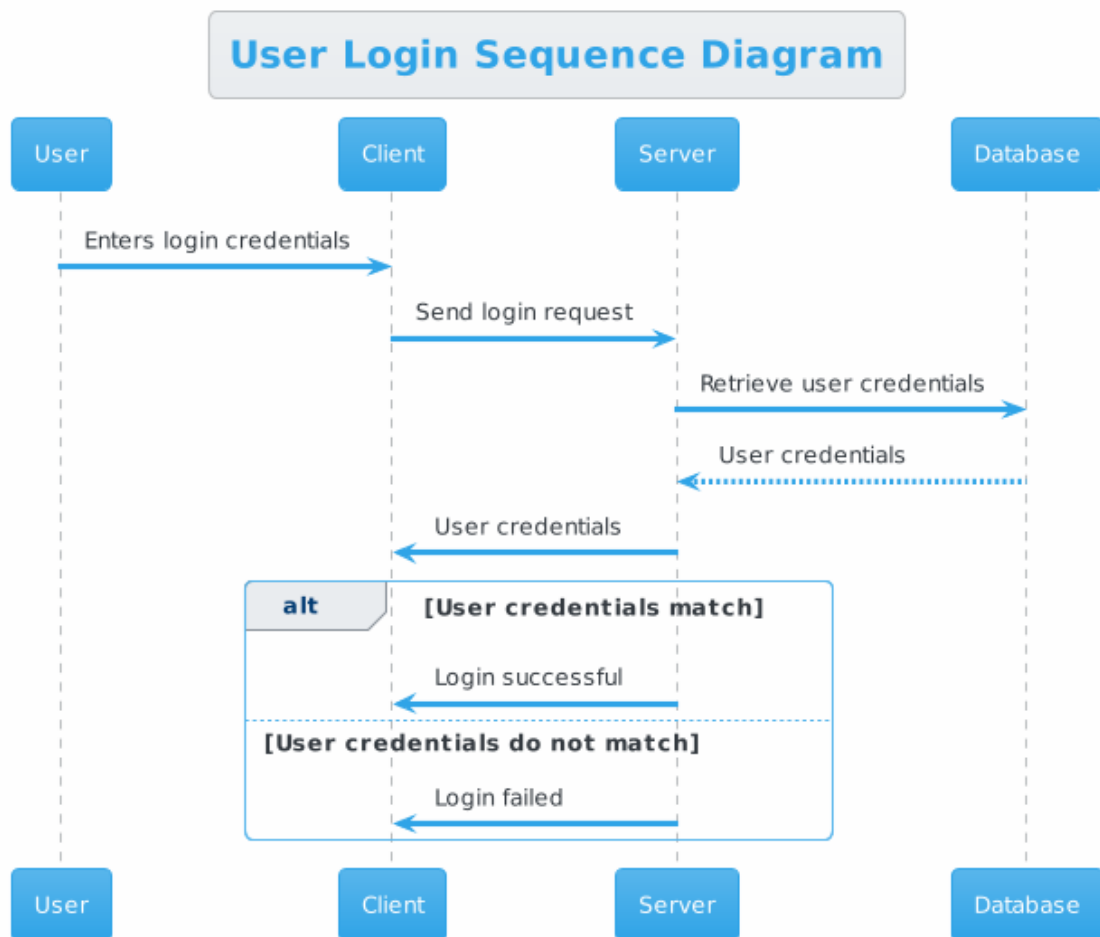
A regisztráció során a felhasználó, az ábrán “User” egy regisztrációs kérést küld az applikáció felé egy gomb lenyomásával, ezután az applikáció bekéri a regisztrációs adatokat, amelyet egy gomb lenyomásával elküld a felhasználó az alkalmazásnak, az applikáció csak akkor engedi meg az elküldést, hogyha kitöltött a felhasználó minden mezőt illetve rendelkezik a megfelelő karakterszámmal és karakter típusokkal. Ezután az applikáció ellenőrzi, hogy az adatbázisban szerepel-e ilyen e-mail című felhasználó. Ha nem szerepel ilyen felhasználó akkor visszaküldi, hogy rendben van, a felhasználó bekerül az adatbázisba, majd az applikáció automatikusan bejelentkeztetni a felhasználót a megadott regisztrációs adatok szerint és átirányítja a főoldalra. Ha van már ilyen felhasználó az adatbázisban akkor az applikáció visszakapja (lásd 6.5 ábra)



6.5. ábra. Regisztráció szekvencia diagram

6.4.2. Bejelentkezés

Bejelentkezés során a felhasználó egy bejelentkezési kérést küld az applikáció felé, a bejelentkezés gomb lenyomásával, amely kéri a felhasználót, hogy adja meg az e-mail címét valamint jelszavát. Az applikáció ez után továbbítja a kérést az API felé, amely ellenőrizni a megadott adatok helyességét. Ha minden helyes akkor az applikáció a szervertől lekéri az aktuális felhasználó adatait, valamint bejelentkezteti, majd átirányítja a főoldalra. Ellenkező esetben jelezve lesz a felhasználónak a sikertelen bejelentkezés (lásd 6.6)



6.6. ábra. Bejelentkezés szekvencia diagram

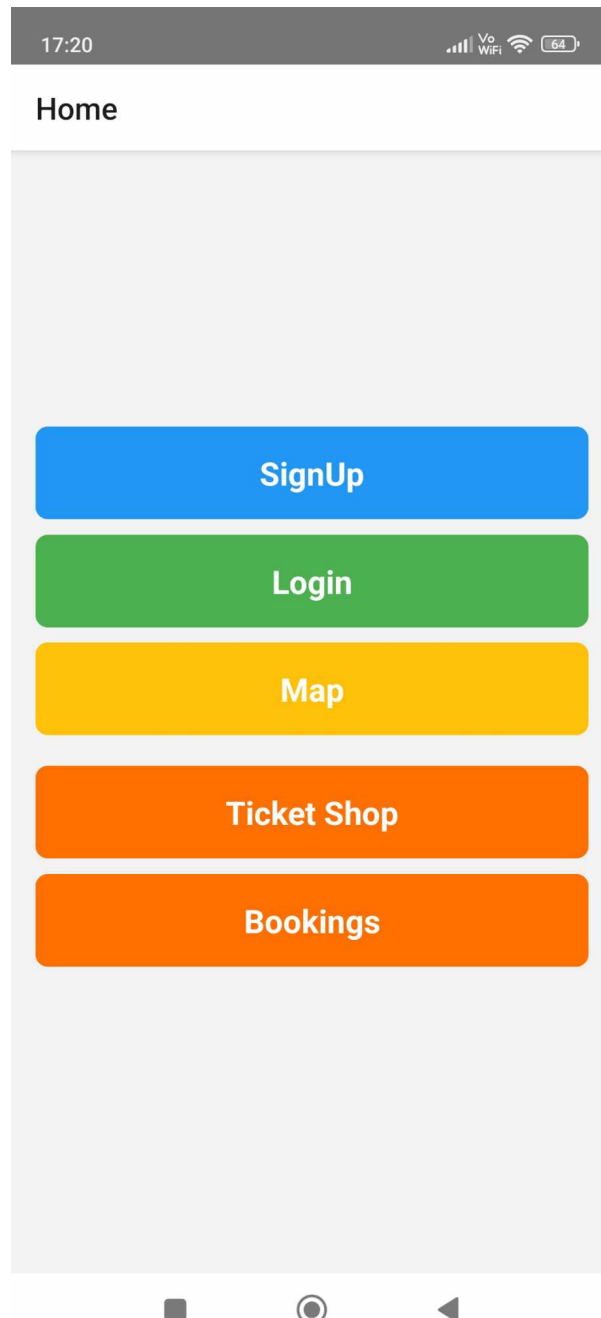
7. fejezet

Alkalmazás ismertetése

A rendszer tervezése során figyelembe kellett vennem, hogy mely dolgokra összpontosítok. Az előző fejezetekben kitűzött célok közül sajnos nem mindeniket tudtam megvalósítani. Mivel inkább a navigáció volt a dolgozat alappillére, ezért a mobilos applikációra fektettem a hangsúlyt. Számítottam arra is, hogy rengeteg problémába ütközök a kivitelezés során, de sajnos még ennek ellenére is volt olyan befolyásoló tényező, amire nem számítottam. Ebben a fejezetben részletesen elmagyarázom az applikáció működését képernyőképek segítségével.

7.1. Kezdőlap

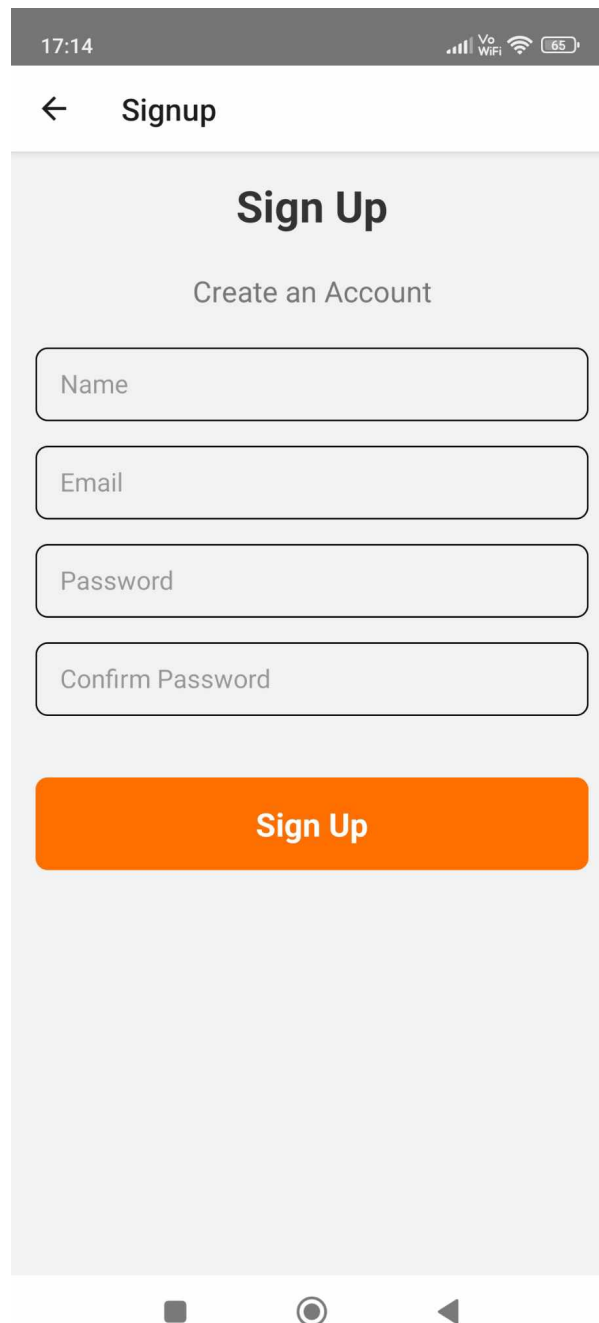
Az alkalmazás elindítása után, megérkezünk a kezdőlapra (Home Page-re), amelyen a következők láthatóak: Sign up, Login, Map, Ticket Shop, Bookings gombok (lásd [7.1 ábra](#)). A Sign up a regisztrációért felelős, a Login a bejelentkezésért, a Map a térkép megjelenítéséért, a Ticket shop a jegyek vásárlásában segít és a Bookings a megvásárolt jegyeket jeleníti meg.



7.1. ábra. Kezdőlap

7.2. Regisztráció

Sign Up gombra kattintva átléphetünk a regisztrációs formra, ahol készíthetünk magunknak egy felhasználót. A felhasználónak mindössze egy nevet, email címet és egy jelszót kell megadnia ahhoz, hogy regisztrálhasson, aminek köszönhetően elérheti a további funkciókat. (lásd [7.2](#) ábra)



The screenshot displays a mobile application interface for user registration. At the top, a status bar shows the time 17:14, signal strength, VoWiFi, and a 65% battery level. Below this is a navigation bar with a back arrow and the text 'Signup'. The main content area has a light gray background and contains the following elements:

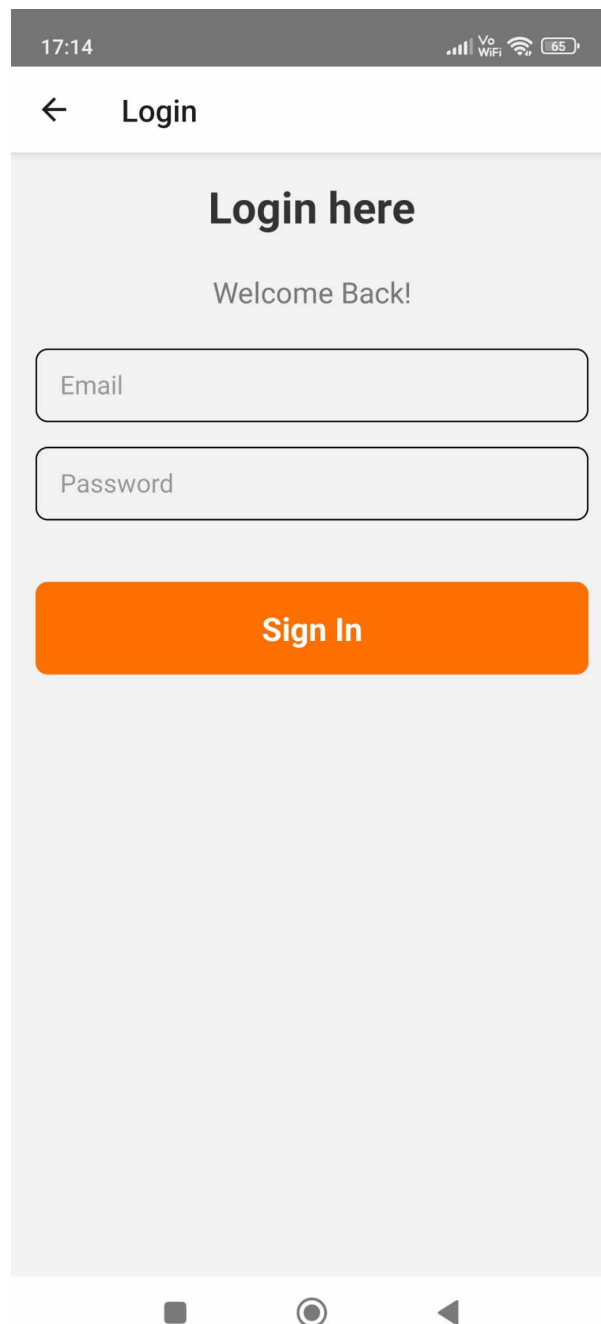
- Sign Up**: Large bold title.
- Create an Account**: Subtitle in a smaller font.
- Name**: Input field.
- Email**: Input field.
- Password**: Input field.
- Confirm Password**: Input field.
- Sign Up**: A prominent orange button with white text.

At the bottom of the screen, there are three standard Android navigation icons: a square, a circle, and a triangle.

7.2. ábra. Regisztráció

7.3. Bejelentkezés

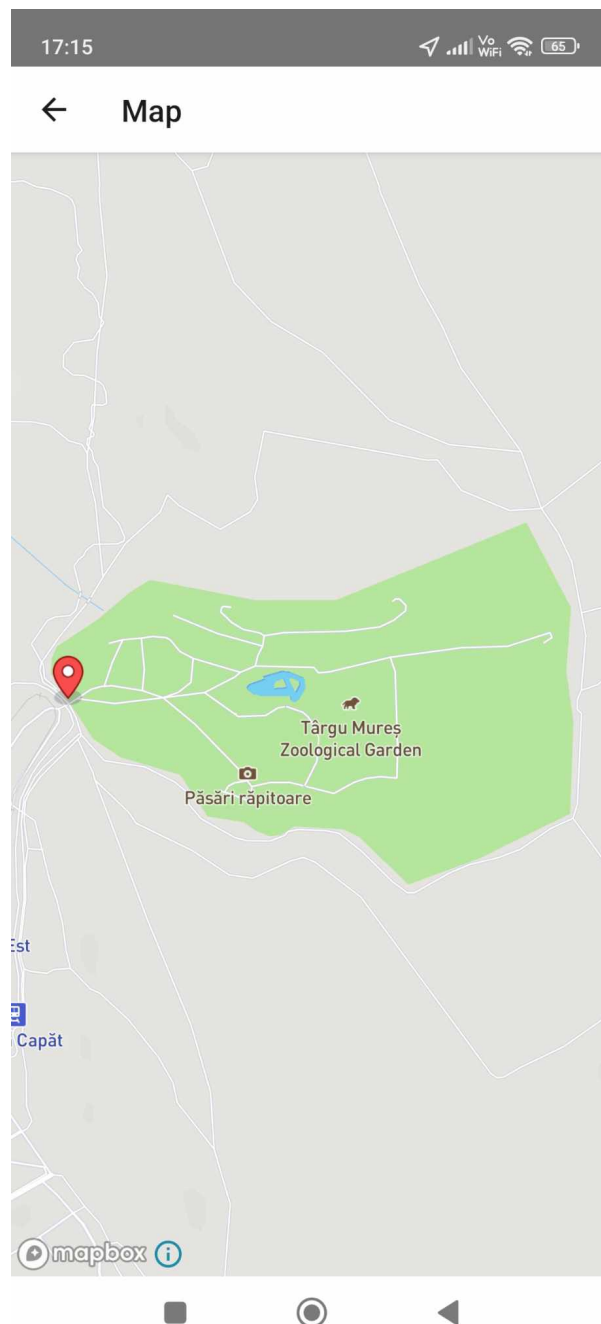
Ahogy megvagyunk a regisztrálással be is jelentkezhetünk, ha sikeres volt a regisztráció. A Login gombra kattintva megjelenik a következő oldal, ahol az adataink beírása után be is tudunk jelentkezni. (lásd 7.3 ábra). Itt csupán csak az email címünket és a jelszót kell megadnunk.

A screenshot of a mobile application's login screen. At the top, a status bar shows the time 17:14, signal strength, VoWiFi, and a 65% battery level. Below the status bar is a navigation bar with a back arrow and the text 'Login'. The main content area has a light gray background. It features the heading 'Login here' in bold, followed by the text 'Welcome Back!'. There are two input fields: 'Email' and 'Password', both with rounded rectangular borders. Below these fields is a large orange button with the text 'Sign In' in white. At the bottom of the screen, there are three Android navigation icons: a square, a circle, and a triangle.

7.3. ábra. Bejelentkezés

7.4. Térkép

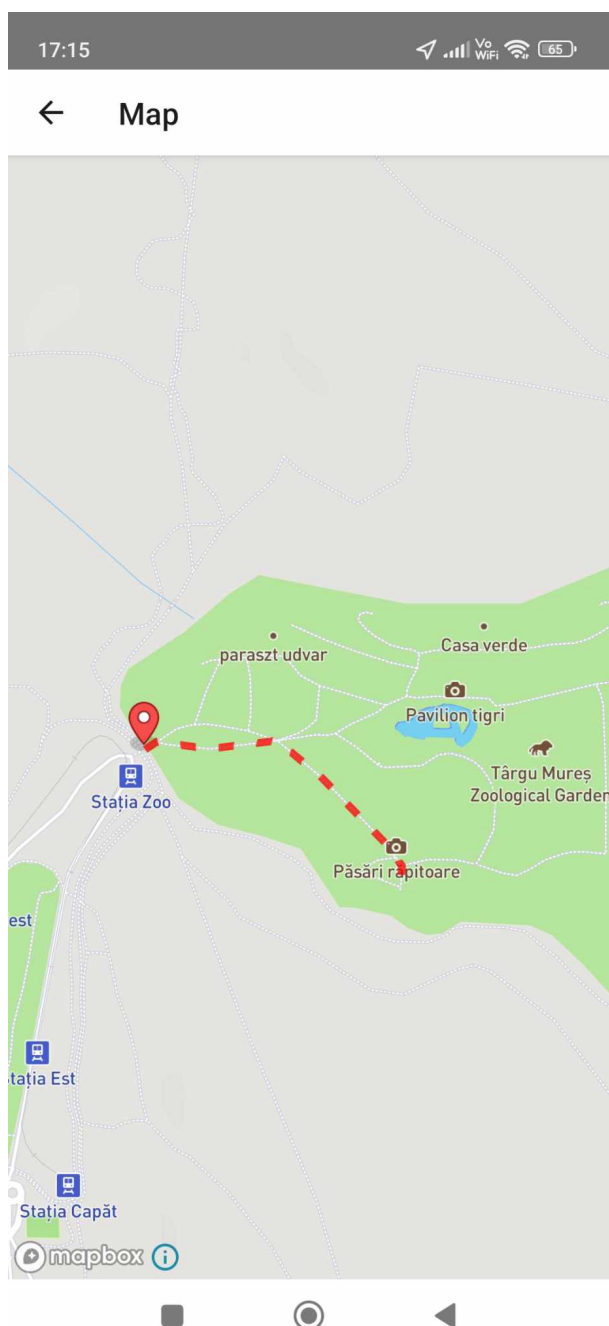
Miután bejelentkeztünk rögtön igénybe is vehetjük az applikáció fő elemének a használatát, a térképes navigációt. A Map gombra kattintva megjelenik egy térkép (lásd 7.4 ábra), ami az állatkertre közelít. A navigáció mindig az aktuális pozíciónktól rajzolja ki a legrövidebb útvonalat a kívánt célig



7.4. ábra. Térkép

7.5. Térkép útvonal kirajzolása

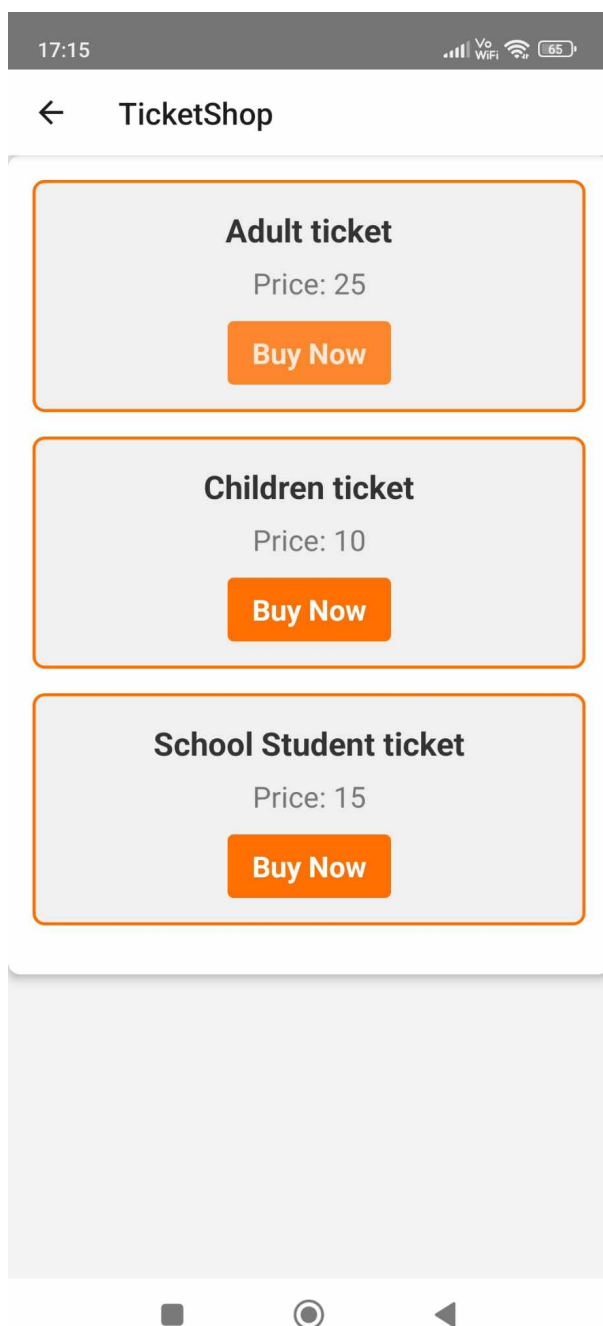
Miután a térkép megjelenik nagyon egyszerűen kirajzólhatjuk a legrövidebb utat. A térképen kiválasztunk egy pontot, hogy hova szeretnénk eljutni és a megérintése után kirajzolódik az útvonal. Ebben az esetben a madárházat választottam ki. (lásd 7.5 ábra). Amint megérkezünk a célunkhoz rögtön kiválaszthatjuk a következő állomást, csupán rá kell bökjünk a térkép egyik pontjára és ismét kirajzolódik a legrövidebb útvonal az aktuális pozíciónktól.



7.5. ábra. Térkép legrövidebb útvonallal

7.6. Jegyvásárlás

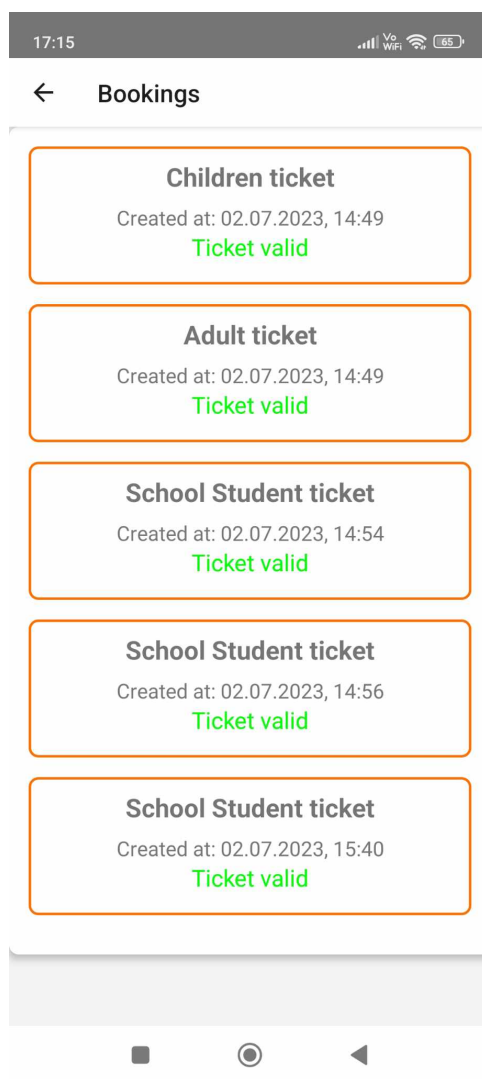
A ticket shop gombra kattintva megjelenik a jegyek listája. A jegyeken látható, hogy milyen típusú és az árak. Egyelőre 3 típust különböztetünk meg, a felnőtt, gyerek és tanuló jegyeket. A jegyeken látható Buy Know gombra kattintva meg is vásárolhatjuk őket. A gombra kattintás után előjön egy fizetési felület, ahol a bankártya adatok kitöltése után, meg is vásárolhatjuk a kívánt jegyeket. (lásd 7.6 ábra)



7.6. ábra. Jegyvásárló

7.7. Megvásárolt jegyek listája

Miután kiválasztottuk a nekünk megfelelő jegyet a BUY NOW gombra kattintva eljutunk egy "stripe checkout" linkre, ahol kifizethetjük a kívánt jegyet. (lásd 7.8 ábra), amint ez megvalósult a Bookings gombra kattintva megérkezünk a megvásárolt jegyek felületére. Ezen a felületen az összes általunk megvásárolt jegy fajtája, vásárlásának az időpontja és státusza is megjelenik. A jegyek általában napos jegyek, ezért másnap már „Ticket expired” fog megjelenni a „Ticket valid” helyett.



7.7. ábra. Megvásárolt jegyek

7.8. Online fizetés

A Stripe developer felülete biztosít nekünk egy „checkout session” linket, ahol az általuk megadott adatokkal online fizetést is szimulálhatunk. (lásd 7.8 ábra) Viszont, ha azt szeretném, hogy a pénzt megérkezzen a számlámra, akkor nincs más dolgom, csak aktiválom kell a felhasználóm és megadnom a bankkártyám adatait.

19:33

checkout.stripe.com

Zoo TEST

Adult ticket jegy
25,00 RON
Ticket for adults

G Pay Fizetés: link

Vagy fizetés kártyával

E-mail-cím admin@gmail.com

Kártyaadatok

1234 1234 1234 1234

HH/ÉÉ CVC

A kártyán szereplő név

Ország vagy régió

Románia

Adataim biztonságos mentése az egykattintásos fizetéshez

Adja meg a telefonszámát, hogy létrehozson egy Link számlát, és gyorsabban fizethessen a(z) Zoo oldalán és bárhol, ahol a Link elfogadott.

0712 034 567 Nem kötelező

link · További tudnivalók

Fizetés

Powered by stripe

7.8. ábra. Online fizetés

8. fejezet

Továbbfejlesztési lehetőségek

A projekt alappillére mivel a navigáció volt, ezért mondhatjuk azt, hogy a fő funkciót sikerült megvalósítani, az alkalmazás használható, de még rengeteg fejlesztési lehetőség van. Ezen a téren lenne egy olyan továbbfejlesztési lehetőség, hogy kiválasztjuk azokat a fajokat, amiket megszeretnénk látogatni s generál egy legrövidebb útvonalat, amivel mindenik pontot érinthetnénk kaputól kapuig, továbbá időt is számolhatna azoknak a meglátogatásához.

A következő pontokban pedig azokra a továbbfejlesztésekre térek ki, amelyekről beszéltem a célkitűzéseknél, de még nem sikerült megvalósítani őket.

Webes felület az átlagos felhasználók számára:

- Továbbfejlesztett kommunikáció az alkalmazással, például az adatok frissítése és szinkronizálása a webes felület és a mobilalkalmazás között.
- Interaktív és felhasználóbarát kezelőfelület a könnyű regisztrációhoz, bejelentkezéshez és jelszócsere lehetőségéhez.
- Kiterjesztett hírportál funkciók, például személyre szabott ajánlatok, programajánlók, látogatói értékelések stb.

Webes felület az állatkert dolgozói számára:

- Részletes statisztikai felület a bevételekről, kiadásokról, látogatói adatokról, dolgozói teljesítményről stb.
- Dolgozói profilok és adminisztrációs felület a felhasználók kezelésére, jogosultságok kezelésére stb.
- Kiterjesztett adatbázis a dolgozók információinak tárolására, beleértve a fizetést, munkaidőt, munkaköröket stb.

Állatok és programok részletes leírása:

- Kiterjesztett információk az állatokról, beleértve fotókat, videókat, életmódot, veszélyeztetettségi állapotot stb.
- Interaktív térkép és időbeosztás a látogatók számára, amelyek részletesen bemutatják az állatok helyét és az aktuális programokat.

Összefoglaló

A dolgozatomban egy állatkerti navigációs alkalmazás létrehozásával foglalkoztam, amelyet több különböző technológia segítségével valósítottam meg. A felhasznált technológiák a következők: React Native, Node.js, Express.js, MongoDB vagyis MERN stack architektúrára épült. A mongoDb Atlas segítségével sikerült egy valós időben működő felhő alapú adatbázist létrehozni. A fejlesztés során úgy érzem, hogy az applikáció alap funkcionálisát sikerült megvalósítani. A projekt megvalósítása alatt úgy gondolom, hogy sikerült betekintést nyerni mindenik technológiába, azonban még biztosan van, ahova fejlődni. A legjobban a mapbox által biztosított API keltette fel a figyelmem, mivel rendelkezek némi térképész tudással, ezért szerintem a jövőben elég sokat fogok foglalkozni ezzel a részével, szeretném magam jobban beleásni magam Mapbox GL által kínált technológiába. A dolgozat kivitelezése során úgy érzem rengeteget fejlődtem, nagyon sok új dolgot megtanultam.

Az alkalmazás frontend és backendjéhez tartozó github repository linkek a következők:

- Frontend: <https://github.com/GAT95/FrontEnd-Zoo-ReactNative-Mapbox.git>
- Backend: <https://github.com/GAT95/BackEnd-Zoo-MERN.git>

Hozzátenném még, hogy a MongoDB, Stripe és Mapbox API tokenkey-et nem tehettem publikussá.

Köszönetnyilvánítás

Köszönettel tartozom elősorban a vezető tanáromnak: Györfi Ágnes tanársegédnek, aki nagyon sok jó tanáccsal látott el, köszönöm neki a türelmet és a megértést, mert véleményem szerint elég nehéz volt velem dolgozni. Sajnos némi egészségügyi probléma miatt, kicsit későn sikerült komolyabban elkezdni a dolgozat megírását és az alkalmazás implementálását. Továbbá köszönettel tartozom az egyetemi kollegáimnak, akik mindig szívesen segítettek, ha elakadtam a munka során.

Ábrák jegyzéke

2.1.	Marosvásárhelyi állatkertről készült papír térkép	4
2.2.	Marosvásárhelyi állatkertről készült interaktív statikus web térkép	5
4.1.	Romániában található állatkertek és delfináriumok elhelyezkedése	9
5.1.	Stripe logó	14
6.1.	MERN architektúra	16
6.2.	React Native és MapBox logók	16
6.3.	Adatbázis diagram	17
6.4.	Use Case Diagram	18
6.5.	Regisztráció szekvencia diagram	19
6.6.	Bejelentkezés szekvencia diagram	20
7.1.	Kezdőlap	22
7.2.	Regisztráció	23
7.3.	Bejelentkezés	24
7.4.	Térkép	25
7.5.	Térkép legrövidebb útvonallal	26
7.6.	Jegyvásárló	27
7.7.	Megvásárolt jegyek	28
7.8.	Online fizetés	29

Irodalomjegyzék

- [CC] Patrick Collison and John Collison.
- [DMR] Eliot Horowitz Dwight Merriman and Kevin Ryan.
- [Fac] Facebook. React native docs, <https://reactnative.dev/docs/getting-started>.
- [Gun] Eric Gundersen. Mapbox, <https://docs.mapbox.com/>.
- [Hol] TJ Holowaychuk. Express.js, <https://expressjs.com/>.
- [Kar] Valeri Karpov. Mongoose js, <https://mongoosejs.com/docs/>.
- [npma] npm. Cookie-parser npm, <https://www.npmjs.com/package/cookie-parser>.
- [npmb] npm. Dotenv npm, <https://www.npmjs.com/package/dotenv>.
- [npmc] npm. Jsonwebtoken npm, <https://www.npmjs.com/package/jsonwebtoken>.
- [npmd] npm. Slugify npm, <https://www.npmjs.com/package/slugify>.
- [Nqu] Calvin Nquyen. Fullstack development with m.e.r.n stack, <https://levelup.gitconnected.com/>.
- [Scha] Isaac Z. Schlueter. Nodemon develop tool, <https://www.npmjs.com/package/nodemon>.
- [Schb] Jonas Schmedtmann. Node.js, express, mongodb more: The complete bootcamp 2023, <https://www.udemy.com/course/nodejs-express-mongodb-bootcamp/>.
- [Wir] Daniel Wirtz. Bcrypt npm, <https://www.npmjs.com/package/bcryptjs>.