UNIVERSITATEA „SAPIENTIA” DIN CLUJ-NAPOCA

FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,

TÎRGU-MUREȘ

SPECIALIZAREA CALCULATOARE

Aplicații de urmărire a contactelor cu telefoane Android

PROIECT DE DIPLOMĂ

Coordonator științific:

Ș.l. dr. ing. Vajda Tamás

Absolvent:

Solyom Botond-Őrs

2021

UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA **Viza facultății:**
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș
Specializarea: **Calculatoare**

LUCRARE DE DIPLOMĂ

Coordonator științific:	Candidat: **Solyom Botond-Őrs**
ș.l. dr. ing. Vajda Tamás	Anul absolvirii: **2021**

a) Tema lucrării de licență:
Aplicații de urmărire a contactelor cu telefoane Android

b) Problemele principale tratate:
- Studiu bibliografic privind sistemele de urmărire a contactelor
- Realizarea unei aplicații pentru urmărire a contactelor

c) Desene obligatorii:
- Schema bloc al aplicației
- Diagrame UML privind software-ul realizat.

d) Softuri obligatorii:
- Aplicații de urmărire a contactelor cu telefoane Android

e) Bibliografia recomandată:

[1] Mbunge, E. (2020). Integrating emerging technologies into COVID-19 contact tracing: Opportunities, challenges and pitfalls. *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, *14*(6), 1631-1636.

[2] Hernández-Orallo, E., Manzoni, P., Calafate, C. T., & Cano, J. C. (2020). Evaluating how smartphone contact tracing technology can reduce the spread of infectious diseases: the case of COVID-19. *IEEE Access*, *8*, 99083-99097.

[3] FreeCodeCamp website, “An awesome guide on how to build RESTful APIs with ASP.NET Core,” [Online]. Available: https://www.freecodecamp.org/news/

[4] GitHub website, “GitHub,” [Online]. Available: https://github.com/mitchtabian/

f) Termene obligatorii de consultații: săptămânal
g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca,
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș
Primit tema la data de: 31.03.2020
Termen de predare: 27.06.2021

Semnătura Director Departament

Semnătura coordonatorului

Semnătura responsabilului programului de studiu

Semnătura candidatului

Declarație

Subsemnatul Solyom Botond-Őrs, absolvent al/a specializării Calculatoare, promoţia 2021 cunoscând prevederile Legii Educaţiei Naţionale 1/2011 şi a Codului de etică şi deontologie profesională a Universităţii Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licenţă/proiect de diplomă/disertaţie se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informaţiile şi datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Localitatea, Târgu Mureș

Data: 06.07.2021

Absolvent

Semnătura 

Aplicații de urmărire a contactelor cu telefoane Android

Extras

Scopul lucrării mele este realizarea unei aplicații care este capabilă de gestionarea contactelor mele cu alte persoane, pe o perioadă de timp determinată. În zilele noastre noțiunea de siguranță este din ce în ce mai importantă din toate punctele de vedere, de aceea păstrarea ei devine prioritară.

Accelerarea vieții de zi cu zi din jurul nostru atrage după sine întâlnirea directă între oameni, creșterea contactelor directe. Creștere exponentială a acestor contacte rezultate, a devenit un aspect hotărâtor al vieții noastre. Din păcate în unele situații acest lucru poate reprezenta un aspect negativ, din motivul că poate influența siguranța personală. Întâlnirile frecvente, înmulțirea contactelor noastre directe pot pune în pericol siguranța noastră personală sau materială.

Pe parcursul realizării lucrării, am avut ca scop crearea unei aplicații, care cu ajutorul tehnologiei Bluetooth, este capabilă a urmării cu cine și unde ne-am întâlnit, perioada de timp petrecută în prezența unei anumite persoane. În același timp, aplicația poate atenționa utilizatorii de potențiale întâlniri cu persoane care pot reprezenta un pericol pentru siguranța lor.

Aplicația este realizată cu ajutorul mediului de dezvoltare Android, cu ajutorul tehnologiei Bluetooth care este pare integrantă a telefoanelor inteligente.

Lucrarea are ca rezultat o aplicație mobilă care rulează pe sistemul de operare Android. Ea poate fi utilă în contextul unei situații epidemiologice, mai exact în orice situație care necesită procesarea de contacte directe, dar și în scopuri personale în care dorim gestionarea proprii mișcări sau întâlniri cu alte persoane.

Cuvinte cheie: urmărire a contactelor, Bluetooth, Android, siguranță

SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR
SZÁMÍTÁSTECHNIKA SZAK

Androidos kontakt alkalmazás

DIPLOMADOLGOZAT

Témavezető:
Dr. Vajda Tamás
egyetemi adjunktus

Végzős hallgató:
Solyom Botond-Őrs

2021

Kivonat

Dolgozatom célja egy olyan alkalmazás megvalósítása, amely képes nyomon követni, és elmenteni a más személyekkel való találkozásokat egy adott időkorlátra visszamenőleg. Napjainkban egyre fontosabbá vált a biztonság fogalma életünk minden területén, ezért ennek megőrzése elsőbbséget élvez emberi létünk megtartása céljából.

A körülöttünk lévő világ felgyorsulása maga után vonja az emberek közti találkozások, kontaktusok gyarapodását, rendszerességének kialakulását. A találkozásokból adódó kontaktusok számának exponenciális növekedése felgyorsult világunk meghatározó motívumává vált. Sajnos bizonyos helyzetekben ez negatív hatást kelthet életünkben, ugyanis a biztonság megőrzésének lehetőségét akadályozhatja meg. A gyakori találkozások, emberek közti kontaktusok sokasága esetenként veszélyt jelenthetnek egészségünk, akár tárgyi értékeink biztonságának megőrzése szempontjából.

Dolgozatom megalkotása során célom egy olyan alkalmazás megvalósítása, amely a Bluetooth technológia segítségével képes követni azt, hogy kivel és hol találkozunk, illetve figyelni egy bizonyos időkorlátot, amit egy adott személy társaságában eltöltünk, amennyiben szükséges, figyelmeztetni a felhasználókat egy potenciálisan, biztonságukra veszélyt jelentő kontaktusról.

Az alkalmazást az Android fejlesztő környezetben sikerült kivitelezni, az okostelefonokba integrált Bluetooth technológia segítségével.

Eredményként egy Android operációs rendszeren futó telefonos alkalmazást kapunk, mely használható járványhelyzet esetén, illetve bármely olyan esetben amikor igényelt a kontaktkutatás, de akár személyes célokra, saját mozgásunk, találkozásaink nyomon követése céljából is alkalmazható.

Kulcsszavak: kontaktkutatás, Bluetooth, Android, biztonság

Abstract

My license thesis aim is to realize an application, which is able to save my appointments, back to a specific time limit. In our days the concept of security it becomes more and more important in every area of our life, therefore the preservation of this, takes precedence keeping our human life.

The acceleration of the world, involves the increase of meetings and contacts between people and the regular formation of this. An exponential increase of these contacts resulting from these meetings becomes a defining motive of our accelerated world. Unfortunately, in some cases it creates a negative effect in our lives, it can prevent the opportunity of preserving the security. The frequently encounters, the multitude of contacts between people, in some cases can be a threat to our heath and the safety of our material assets.

During the creation of my license thesis, my goal was the implementation of an application which, with the help of Bluetooth technology, can follow where and with whom we meet and pursues a certain time limit, which we are spending with a certain person. If it is necessary warns the users about a potential contact that could pose a danger of their safety.

The application was successfully implemented in the Android development environment, with the help of Bluetooth technology integrated in the smartphones.

As a result, we get a phone application running on an Android operating system which can be used in case of pandemic, or in any case when is required the contact research, but it can also be used even for personal reason for the tracking of our own movements and meetings.

Keywords: contact researcher, Bluetooth, Android, safety

Tartalomjegyzék

Ábrák jegyzéke

1. Bevezető

Dolgozatom célja egy olyan alkalmazás megvalósítása, amely képes nyomon követni, és elmenteni a más személyekkel való találkozásokat egy adott időkorlátra visszamenőleg, felhasználva az Android okostelefonokban integrált Bluetooth technológia által nyújtott lehetőségeket. A körülöttünk lévő világ természetességét bizonyítja az a tény, hogy az idő teltével ahogy emberi életünk, úgy a világunk is változik. Mint emberi életünk szakaszaiban, úgy a világban is jelen van a változás fogalma, ami több szempontból is értelmezhető, legyen az éghajlati, gazdasági vagy esetleg technológiai szempont. Értelmezhető szempont lehet a világ változását tekintve, a bennünket körülvevő világ fejlődésének tulajdonítható felgyorsulás is, mely napjaink jellegzetességévé vált. Világunk felgyorsulása maga után vonja az emberek közti találkozások, illetve az ezekből adódó kontaktusok kialakulásának gyarapodását és rendszerességét. Ennek tulajdonítható az a tény, hogy a találkozásokból adódó korrelációk számának nagymértékben történő növekedése felgyorsult világunk meghatározó motívumává vált. Sajnos bizonyos helyzetekben ez negatív hatással lehet életünkre, megakadályozhatják a biztonság megőrzésének a lehetőségét. Az emberek közti kapcsolatok sokasága esetenként veszély forrását jelenthetik egymás számára, úgy egészségügyi, mint más szempontokból is.

Ennek értelmében fogalmazódott meg dolgozatom témája, mely a kontaktkutatást tárgyalja, és ennek megvalósítására, fejlesztésére és egyszerűsítésére keres megoldásokat, egy olyan szoftver megalkotása révén, mely könnyedén elősegítheti különböző szervezetek, illetve egyének számára a találkozásokból származó kontaktusok felderítését egy visszamenő időszakra tekintve. Azt a tényt figyelembe véve, hogy napjainkban egyre fontosabbá vált a biztonság fogalma életünk minden területén, ezért ennek megőrzése elsőbbségi célt jelent az emberiség számára. Léteznek bizonyos helyzetek, melyek életünk részét képezhetik, és melyek a találkozások gyakoriságából adódóan megnehezíthetik biztonságunk megőrzésének lehetőségét, legyen szó egészségügyi biztonságunkról, vagy esetlegesen tárgyi értékeink biztonságának a kérdéséről. Dolgozatom célja egy olyan alkalmazás megvalósítása, amely képes elmenteni találkozásaimat visszamenőleg egy adott időszakra, meghatározni azt, hogy kivel találkoztam az elmúlt hetekben, ezek közül pedig mely kontaktus volt az, ami esetlegesen veszélyeztethette biztonságomat, egészségügyi állapotomat. A cél egy olyan alkalmazás tervezése volt, melyet egyaránt fel lehet használni állami szervezetek által, esetlegesen járványhelyzetek esetében, ezek megelőzése ellen, vagy megakadályozása céljából, vagy akár egyénileg saját biztonságunk szempontjából, továbbá

egyéni mozgásaink nyomon követése tekintetében. Mivel a biztonság fogalma életünk minden területén kiemelkedő értéket képvisel, ezért fontosnak tartottuk ennek megőrzését a felhasználók tekintetében is, így a rendszer célja úgy szolgálni a közösséget, hogy a felhasználók személyes adatait biztonságban megőrizze, különböző azonosítókat felhasználva ennek érdekében, illetve a felhasználókat abban az esetben értesítve, amennyiben ez szükséges. A kontaktkutatás által lehetőség nyílik úgy az egyén, mint a közösség biztonságát megőrizni, amennyiben azt a helyzet megköveteli, ennek megvalósítására pedig kimondottan előnyt jelent a napjainkban igen nagymértékű használatnak örvendő okostelefonok rendszeres használata az emberek körében, ami manapság szinte kivétel nélkül szerves részét képezi minden egyén életének. Ennek köszönhetően az okostelefonok által támogatott különböző technológiák által már bárki számára ismert lehet a telefon alapú nyomon követés, illetve a helymeghatározás fogalma, azonban ezen technológiák felhasználásának fontosságát és hasznosságát biztosítja az, amikor az egyén, illetve a közösség biztonságának a megőrzése a cél. Ezért kihasználva a technológiai fejlődés lehetőségeit dolgozatomban az okostelefonokba integrált Bluetooth technológiát felhasználva sikerült elérni, hogy az alkalmazás elérje a kitűzött célokat.

Felgyorsult világunkban, úgy gondolom az egyén és a közösség biztonsága egyaránt fontos értékeket képviselnek, ezért bizonyos, az élet által hozott helyzetekben elengedhetetlen a kontaktkutatás fontossága, ennek értelmében a továbbiakban az okostelefonok előnyeit és a Bluetooth technológiát felhasználva kerül bemutatásra a kontaktusok felderítését szolgáló alkalmazás megvalósítása.

2. Elméleti megalapozás és szakirodalmi tanulmány

A tervezett projekt megvalósítása különböző szakaszokból tevődik össze. Az első lépést követően, a pontos téma, illetve a motiváció megfogalmazása után az elméleti megalapozás és szakirodalmi tanulmány következik, tehát a dokumentálódás az adott témakörben. Ebben a fejezetben kerülnek bemutatásra az általam tanulmányozott szakirodalmi tanulmányok, melyek útmutatásként szolgálnak a terv kivitelezése szempontjából, továbbá bizonyos elméleti alapok, melyek megalapozzák az adott területben felhasznált technológiák ismeretét, valamint azon irányelveket, melyek meghatározzák a szoftver biztonságos használatát, illetve az ismert hasonló alkalmazások bemutatása, melyek irányvonalat biztosítottak a projekt kivitelezésében, viszonyítási alapként szolgálva. Végül pedig az általam felhasznált technológiák bemutatásával

zárul a fejezet, melyek az alkalmazás megvalósítását tették lehetővé, és annak építő köveiként szolgálnak.

2.1. Szakirodalmi tanulmány

Ebben a részben a témakörben az általam tanulmányozott szakirodalom kerül bemutatásra, mely részben összefüggésbe hozható az általunk megvalósított alkalmazással. Az alábbiakban olyan tanulmányokról lesz szó, melyek az alkalmazásunkban is felhasznált technológiák általi megvalósításokról szólnak. Az alkalmazás megvalósítása során számos kérdés fogalmazódott meg, melyek közül a legjelentősebb az, hogy milyen technológiák segítségével érhető el a tervezett cél. Kérdésemre a válasz egy angol nyelvű, friss tanulmány keretein belül fogalmazódott meg, melyet Elliot Mbunge az Eswatini Egyetem Számítástechnika szakjának egyetemi adjunktusa publikált a 2020–as év augusztusában, amikor számos kutató csoportot és szakembert foglalkoztatott a kontaktkutatás megvalósításának kérdése a technológia által. A tanulmány, mint ahogy címe is mutatja a feltörekvő technológiák kontaktkutatásban való integrálásának lehetőségeire, kihívásaira és buktatóira keresi a választ, azon technológiák felhasználására, melyek szinte kivétel nélkül megtalálhatóak napjaink okostelefonjaiban. A közzétett kutatás számos tudományos környezetben az „Integrating emerging technologies into COVID-19 contact tracing: Opportunities, challenges and pitfalls” címen vált elérhetővé az olvasók számára, utat mutatva a technológia által nyújtott lehetőségek hasznos célra való felhasználására [1]. Kezdetben a kontaktkutatás, kontaktusok, találkozások feltárásának fogalmából kiindulva szükséges értelmezni azt, hogy tulajdonképpen mit is értünk e fogalom alatt.

Tudományos megfogalmazásokból inspirálódva, a közegészségügyben az úgynevezett kontaktusok követése, feltárása alatt azt a folyamatot értik, amelynek során azokat a személyeket azonosítják, akik kapcsolatba kerülhettek, egy számukra, vagy biztonságukra veszélyt jelentő személlyel, személyekkel. Az általam tanulmányozott kutatások mellett, számos tanulmány egyhangúan egyetért abban, hogy az emberek közötti találkozások felfedezésére, nyomon követésére a legalkalmasabb egy olyan eszköz által nyújtott lehetőségek kihasználása, ami rendszeresen az emberek tulajdonában van, bárhol is tartózkodnának. Ez az eszköz, pedig egyértelműen az okostelefon, mely az utóbbi időben felgyorsult világunk szerves részévé vált. A kontaktusok felderítése az adott személyek helyzetének meghatározására enged következtetni, ugyanis egyértelmű az a tény, hogy amennyiben két személy azonos helyszínen tartózkodnak, nagy valószínűséggel értelmezhető közöttük a találkozás fogalma. Az eredmények pontossága érdekében fontos azonban odafigyelni arra, hogy milyen szabályrendszer határozza meg a

találkozás fogalmát, tehát hogy mennyire közel helyezkednek el egymáshoz az adott személyek, illetve, hogy mennyi idő telt el a találkozás beazonosításától, a kontaktus megszűnéséig. Az okostelefonokba integrált technológiák általi helymeghatározás fogalma napjainkban már bárki számára ismert fogalom, ugyanis számos alkalmazás működésének alapfeltételét képezi a felhasználó helyzetének meghatározása, legyen szó bármely térkép, illetve útvonal tervező alkalmazásról, vagy akár időjárást meghatározó szoftverről. Ezen alkalmazások rendszerint, a jól ismert, illetve jól bevált technológiát, a Globális Helymeghatározó Rendszer, angol nevén a Global Positioning System, rövidítve, ismertebb megnevezésként a GPS technológia által nyújtott lehetőségeket használják a személy, pontosabban a személy által használt eszköz helyzetének meghatározására. A GPS használatát számos szoftver igényli a helymeghatározás céljából, többnyire fejlettségének és méteres nagyságrendű pontosságának köszönhetően. Amennyiben azonban kontaktkutatásról van szó, bár a GPS használat is elősegítheti célunk elérését, nem feltétlenül szükséges arra a kérdésre választ kapjunk, hogy a figyelt személyek hol találkoztak, hanem fontosabb lehet annak a kérdésnek a megválaszolása, hogy kivel találkoztak. Ennek felderítésére segítségül szolgálhatnak a Wi-Fi, illetve a Bluetooth technológiákban rejlő lehetőségek is, melyeket felhasználva választ kaphatunk arra, hogy kivel kerültünk kontaktusba, elhanyagolva azt a tényt, hogy hol, elősegítve ezáltal a személyes adatok biztonságának megőrzését. Napjainkban számos kutatás, illetve tanulmány készült arról, hogy az előzőekben említett technológiák közül, melyek bizonyulnak hasznosabbnak a tervezett cél elérésének szempontjából.

Egy általam tanulmányozott kutatás, mely szintén a 2020–as évben került bemutatásra, az a Politécnica de Valéncia Egyetem Hálózatépítési Kutatócsoport – DISCA alkotta Számítástechnika szakjának képviselői által lett publikálva az „Evaluating How Smartphone Contact Tracing Technology Can Reduce the Spread of Infectious Diseases: The Case of COVID-19” cím alatt, mely szintén a technológia hasznosságára hivatott rá világítani a kapcsolatok nyomon követésének témakörében [3]. A kutatócsoport mindhárom technológiát figyelembe véve arra a kérdésre kereste a választ tanulmányában, hogy ezek közül melyik bizonyosul a legjobb szövetségesnek a kontaktusok felderítése céljából. Az IEEE Access nevű folyóiratban is közzétett tanulmány azt a következtetést fogalmazta meg, hogy a Bluetooth technológia bizonyul a legjobb lehetőségnek az érintkezések nyomon követésének szempontjából. A legfontosabb előny, mely megfogalmazódott a Bluetooth használata mellett, az a technológia jelerősségének mértékéből származó nagy pontosság, amely jóval magasabbnak minősült a GPS, illetve a Wi-Fi

technológiákkal szemben mért pontossággal. Bár mindhárom technológia pontossága kitűnő eredményeket bizonyított a kutatások során, azonban „a Bluetooth a legalkalmasabb technológia, mert lehetővé teszi a nyomkövetők számára, hogy 2-3 méteres távolságon belül észleljék az érintkezéseket”, vélekedett a kutatócsoport egyik tagja és egyben a tanulmány egyik szerzője Enrique Hernández Orallo. A kutatás szerint az ezen a ható távolságon belül észlelt érintkezéseket tekintik az epidemiológiai modellek egy esetleges fertőzés átadására képes kapcsolatoknak, tehát az ennek megfelelő kapcsolatok számítanak valós kontaktusoknak, az ezen kívül eső észlelt kontaktusok hamis kapcsolatokként tarthatók számon. A kutatócsoport rá világít az okostelefonokra fejlesztett kontaktkutatást elősegítő alkalmazások fontosságára is, ugyanis ezek „rendkívül hasznosnak” tekinthetőek a jövőre nézve esetleges újabb járványok kialakulásának elkerülése érdekében, ugyanis „az okostelefonokon alapuló érintkezés nyomon követése rendkívül hasznos, még akkor is, ha csak a lakosság egy része, kevesebb, mint 60% -a hajlandó használni” magyarázta a kutató [2].

Dokumentálódásom során újabb ismeretekkel gazdagodhattam, tanulmányozva a Massachusetts Institute of Technology magánegyetem kutatói által vezetett és sok más intézmény szakértőiből álló csoport kutatásán alapuló rendszer fejlesztésének alapjait, melyet a kontaktkutatás céljából fejlesztettek. A kutatócsoport célja az volt, hogy úgy valósítsák meg a kapcsolatok feltárását a különböző személyek között, hogy közben megőrizzék a felhasználók személyes adataink biztonságát. Céljaik elérése érdekében használták fel az okostelefonokba beépített Bluetooth technológiát, és az általa kibocsátott jeleket és azok erősségét, valamint a felhasználók személyes adatainak védelme érdekében véletlenszerűen generált azonosítókat alkalmaztak, ugyanis azt az alapelvet igyekeztek követni, hogy maximális biztonságot jelentsen bárki számára az alkalmazás használata [4].

2.2. Elméleti alapok

A dokumentálódás során sikerült útmutatást szerezni arról, hogy mely technológiák segíthetik elő az alkalmazás helyes működését. A tanulmányok kutatását követően az új ismeretek elméleti megalapozása következik, melyek az alábbiakban fogalmazódnak meg.

2.2.1. Bluetooth technológia

A Bluetooth technológia sikerének egyik legfontosabb tényezője a két rádió opciót kínáló technológia sokoldalú megoldásai, melyek rugalmasságot nyújtanak a fejlesztők számára terveik megvalósítása szempontjából. A technológia két rádió opciót kínál a vezeték nélküli kapcsolatok

egyre növekvő igényeinek kielégítése céljából. Ez alapján beszélhetünk a klasszikus, úgynevezett Bluetooth Classic opcióról, ami alacsony fogyasztású rádióhullámok segítségével 79 csatornán keresztül továbbítja az adatokat a 2,4 GHz –es, tudományos, ipari és orvosi frekvenciasávban. A pont-pont közötti kommunikációt biztosító Bluetooth Classic főként a vezeték nélküli audio streaming engedélyezésére szolgál, illetve a vezeték nélküli fejhallgatók, hangszórók, esetlegesen az autókban is használt rendszerek mögött álló rádió protokollként feleltethető meg. Egy a technológia által kínált másik opciónak a Bluetooth Low Energy-t (LE) említhetjük, amelyet a fejlesztők célszerűen nagyon alacsony fogyasztású működésre terveztek. Ez esetben az adatátvitel 40 csatornán keresztül a 2,4 GHz –es frekvenciasávban történik. A Bluetooth Low Energy társával szemben már többféle kommunikációs topológiát támogat, a pontról-pontra topológiától kezdve a hálózat alapú topológiáig, lehetőséget biztosítva ezáltal a nagyszabású eszközhálózatok létrehozására. Bár a technológia kezdetben eszközkommunikációs képességeiről vált ismertté, a Bluetooth Low Energy–t ma már előszeretettel használják eszközök meghatározására is, kielégítve ezáltal a beltéri nagy pontosságú helymeghatározó szolgáltatások irányába növekvő igényt is [5]. A Bluetooth, mint vezetéknélküli rádiótechnológia rövid hatótávolságú jelek felhasználásával teszi lehetővé a kommunikációt, illetve az adatcserét a számítástechnikai eszközök között. Úgy létesít kapcsolatot az egyes eszközökkel, hogy lehetővé teszi a kábelek használata nélküli jelek átvitelét a készülékek között. A technológia a kábelek felhasználásának elkerülése mellett számos pozitív tulajdonságot hordoz magával, mint például az alacsony energiafogyasztásának előnyéből származó lehetőséget, mely szerint alkalmas hordozható eszközök számára is [6].

Napjainkban ezen technológia használata egyre elterjedtebb a mobil eszközök széles körében, ugyanis megtalálható a nagy gyakorisággal használt számítástechnikai eszközeinkben, legyen szó telefonokról, hordozható számítógépekről, akár nyomtatókról, vagy bármely más eszközről, melyek mindennapi, akár szórakozási tevékenységünket, segítik elő. A különböző rendszerekbe integrált technológia főként a használat könnyítését hivatott segíteni, ugyanis olyan módon valósítja meg a kapcsolatot az egyes eszközök között, hogy elkerülhető legyen kábelek használata. A használat megkönnyítése mellet pedig akár a biztonságot is elősegítheti bizonyos értelemben, mint például az autókba integrált rendszerek működésével, vagy akár egy kontaktkutatást végző alkalmazás működésének biztosításával.

2.2.2. GPS technológia

A GPS megnevezés a különböző szakirodalmi tanulmányokból is ismert Global Positioning System angol nyelvű szó rövidített változatából ered, melyet magyarul Globális

Helymeghatározó Rendszer–nek nevezünk. Mint ahogy nevéből is ered, egy olyan helymeghatározó rendszer, amely lehetővé teszi a háromdimenziós helymeghatározásának lehetőségét a világ bármely pontján, a földfelszínen, vízen, de levegőben egyaránt. Pontosságát illetően méteres nagyságrendű pontosságot érhetünk el használatával, azonban elérhető az akár milliméteres pontosság is valós időben. A fejlett helymeghatározó rendszer előnye, hogy a kívánt helyzet meghatározását keringő műholdak segítségével képes megvalósítani, folyamatos adatelérést biztosítva a nap 24 órájában. Mint sok más technológia, így a GPS fejlesztésének hátterében is katonai célok álltak, azonban napjainkban már hétköznapi használatra is elterjedt rendszerré nőtte ki magát. Számos rendszerben megtalálható ezen technológia, így az okostelefonok szolgáltatásainak is szerves részévé vált. Kiemelkedő előnyt jelent a telefonokban is megtalálható GPS technológia, mely manapság számos alkalmazás működésének alapját képezi, továbbá a biztonság megőrzését is elősegítő rendszerként működik [7].

2.2.3. Wi-Fi

A Wi-Fi a vezeték nélküli helyi hálózatok (WLAN) IEEE 802.11 –es szabványa, melynek célja úgy megvalósítani az adatátvitelhez szükséges kapcsolatot az egyes számítástechnikai eszközök között, hogy elhanyagolható legyen a kábelek használata. A Bluetooth technológiához hasonlóan a Wireless Fidelity, melyet a köztudatban helytelenül a Wi-Fi megfelelőjeként tartanak számon, rádióhullámok segítségével képes megteremteni a kapcsolatot a Wi-Fi hozzáférési pontok és egyes számítástechnikai eszközök között. A Bluetooth technológiához hasonlóan a Wireless technológiának is megfeleltethető a jelerősség fogalma, mely nagyban befolyásolhatja a technológia működését, illetve a kapcsolatot biztosító jelek minőségét [8].

2.2.4. GDPR irányelvek

A többnyire használt GDPR rövidítés az angol eredetű General Data Protection jelentésből származik, melynek magyar megfelelője az Általános Adatvédelmi Rendelet lenne. A GDPR, mint adatvédelmi rendelet, a világ legszigorúbb adatvédelmi és biztonsági törvényeként számontartott rendelet, melyet az Európai Unió (EU) dolgozott ki és fogadott el, melynek célja az EU-ban élő emberek személyes adatainak védelmét biztosítani. A GDPR szigorú bírságokat fogalmaz meg azokra a szervezetekre, illetve személyekre, akik megsértik az adatvédelmi és biztonsági normákat, jelezvén ezáltal egy határozott álláspontot a személyes adatok biztonsága terén. A rendelet világosan megfogalmazott pontokban tárgyalja törvényeit, így ennek betartása különös figyelmet követel bármely kis- és középvállalkozások számára. Fontos megjegyezni továbbá azt,

hogy amennyiben uniós polgárok számára kínálunk termékeket, vagy szolgáltatásokat, tartózkodástól függetlenül szükséges figyelembe venni a GDPR irányelveit, melyben hosszasan megfogalmazódnak a betartandó jogi kifejezések, mint például a személyes adatra vonatkozó törvénykezések [9].

A rendelet érthetően meghatározza a személyes adatnak a fogalmát, hogy pontosan mit értenek ezen meghatározás alatt, illetve, hogy mely szabályozásokat kell figyelemmel követni az esetleges felhasználás következtében. Ez alapján személyes adatnak tekinthető minden olyan személyre vonatkozó információ, amely közvetett vagy közvetlen módon egy azonosítható személyre vonatkozik, ilyen adatok lehetnek például egy személy vezeték, illetve keresztneve, lakcíme, személyazonosító igazolványának száma, de hasonló adatoknak számítanak a helymeghatározó adatok is, melyek egy okostelefon helymeghatározási funkcióját veszik igénybe, továbbá IP-címek, illetve a vezeték és utóneveket tartalmazó e-mail címek is. Megjegyzendő az is, hogy léteznek bizonyos esetek, amikor a helymeghatározó adatokra egy speciálisan megfogalmazott jogszabály vonatkozik. Nem számítanak személyes adatoknak például a cégjegyzékszámok, bizonyos e-mail címek, melyek nem tartalmaznak személyes információkat tulajdonosáról, mint például keresztnevét, vagy utónevét, illetve az anonimizált adatok, ugyanis a rendelet alátámasztja azt a tényt, hogy „az olyan személyes adatok, amelyeket olyan módon anonimizáltak, amelynek következtében az érintett nem vagy többé nem azonosítható, nem tekinthetőek többé személyes adatnak” [10].

2.2.5. Álnevesítés és anonimizálás a GDPR alapján

A személyes adatok védelmének érdekében szükségszerű különbséget tenni az álnevesítés és az anonimizálás fogalmak között, ugyanis a GDPR rendelet határozatai alapján, csak az anonimizált adatok nem tekinthetők személyes adatnak, az álnevesített adatok viszont továbbra is a személyes adatokra vonatkozó szabályok hatókörében maradnak.

Anonimizált adatnak tekinthetők az olyan adatok, melyek visszafordíthatatlanok, vagyis nem visszafejthetők személyes adattá.

Álnevesített adatnak tekinthetők az olyan titkosított adatok, amelyekről már nem határozható meg, hogy a személyes adat mely konkrét személyre vonatkozik, azonban ezek újra felhasználhatóvá válhatnak esetleges újabb beazonosítás érdekében [10].

2.3. Ismert hasonló alkalmazások

Az elméleti megalapozás után hasonló célt szolgáló, illetve hasonló elven működő alkalmazások kutatása következik, mely folyamat során előtérbe helyezhető az alkalmazások megvalósításához alkalmazott technológiák vizsgálata, illetve az alkalmazások működési és megvalósítási szempontból való elemzése.

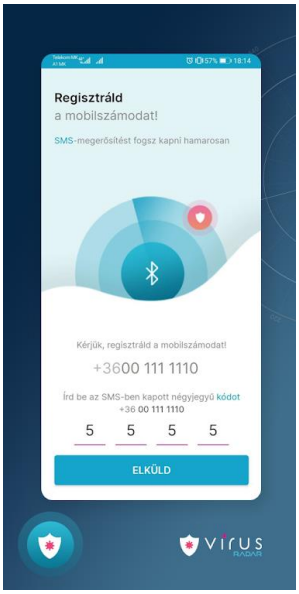
2.3.1. VírusRadar

A VírusRadar nevezetű alkalmazás egyike azon alkalmazásoknak, melyet különböző országok szakemberei fejlesztettek ki az egészségügyi biztonság megőrzésének céljából. Az alkalmazást Magyarország lakosságának számára fejlesztették, melynek célja elősegíteni az ország egészségügyi központjának a kontaktkutatását.

A VírusRadar alkalmazás működésének hátterében a Bluetooth technológia áll, ugyanis ennek segítségével valósul meg a kommunikáció a felhasználók okostelefonjai között. A Bluetooth alapján tehát a rendszer rögzíti az adatokat és 14 napig tárolja ezeket titkosított formában. Amennyiben egy felhasználó megfertőződne, abban az esetben megoszthatja az alkalmazás által tárolt adatait a kontaktkutatást végző szakemberekkel, akiknek így lehetőségük adódik figyelmeztetni az adott személlyel kontaktusban lévő személyeket. Figyelemre méltó az értesítések küldésének megvalósítása, ugyanis a rendszer a felhasználók értesítésekor nem közöl más felhasználókkal személyes adatokat, mindössze arról figyelmeztet, hogy biztonságára veszélyes személlyel került kontaktusba az elmúlt időszakban. Az alkalmazás tehát teljes biztonsággal kezeli a felhasználói adatokat, mindössze egyetlen személyes adatot gyűjt a felhasználókról, a telefonszámukat, amit kizárólag csak a felhasználó hozzájárulásával és esetleges megfertőződése esetén használnak fel, továbbá ezt egy biztonságos szerveren tárolják [11]. Az alábbiakban egy rövid ismertető keretében bemutatásra kerül az alkalmazás működése, így kezdetben egy az alkalmazás elindítását követő ábra kerül szemléltetésre.

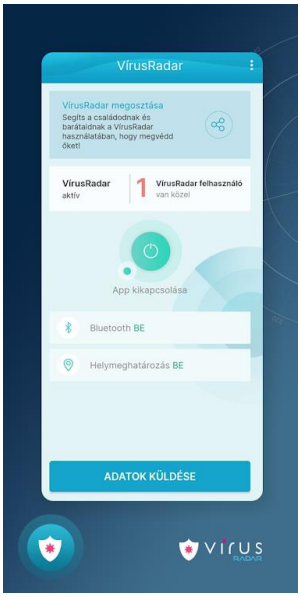
2.1. ábra: VírusRadar alkalmazás kezdőképernyője [12]

Kezdetben az alkalmazás indítását követően, egy regisztrációhoz szükséges oldal jelenik meg a felhasználók számára, ahol az alkalmazás begyűjti az egyetlen személyes információt a felhasználóról, pontosabban a telefonszámát. Ekkor egy véletlenszerű kódot generál az alkalmazás, a felhasználó azonosítása céljából, majd ezt a kódot hozzárendelik az adott telefonszámhoz, majd eltárolják az alkalmazás biztonságos szerverein, vigyázva, hogy ezt soha senkivel sem közöljék. Az alábbi ábrán az alkalmazás regisztrációs oldala látható, mely egy általános azonosítást végez a felhasználóval [11].



2.2.ábra: VírusRadar alkalmazás regisztrációs oldala [12]

Működés szempontból az alkalmazás a Bluetooth technológia segítségével, és annak jelerősségének felhasználásának köszönhetően becsüli fel az észlelhető távolságot az eszközök között, megőrizve ezáltal a felhasználók tartózkodási helyzetéről szóló adatokat, ugyanis az alkalmazás nem gyűjt információkat az egyén mozgásáról. A szoftver úgy van megvalósítva, hogy az alkalmazást futtató telefonok Bluetooth segítségével kódot cserélnek, melyek a saját telefonon vannak tárolva, s melyeket kizárólag csak a kontaktkutatást végző szakemberek tudnak visszafejteni. Amennyiben egy felhasználónál a megfertőzés bekövetkezését észlelték, a felhasználónak egyénileg joga van dönteni arról, hogy az alkalmazás által gyűjtött és így az eszközén tárolt adatokat hajlandó-e beküldeni a kutatást végző szakemberek számára. A Bluetooth által kibocsátott jelek erősségének meghatározására a rendszer az úgynevezett RSSI értékeket használja fel, illetve a kontaktus időtartamát az alkalmazás rögzíti. Az alábbiakban az alkalmazás kezelőfelületének képernyője kerül szemléltetésre [11].

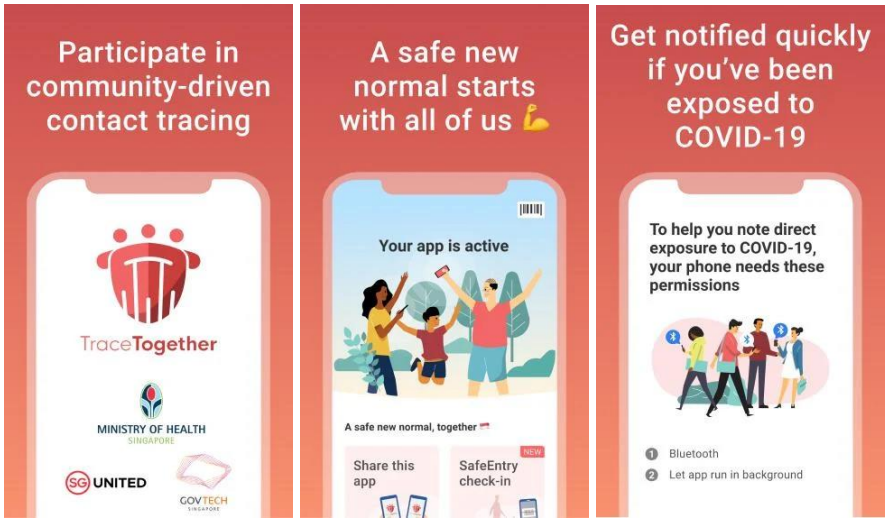


2.3.ábra: VírusRadar alkalmazás kezelőfelületének képernyője [12]

Az alkalmazás megvalósításának célja az országban zajló kontaktkutatásnak az elősegítése, továbbá a lakosság egészségügyi biztonságának a megőrzése. Ezen cél elérésére az alkalmazás akadálymentes működése mellett, nagymértékben szükség van a felhasználók hozzájárulására is, illetve az alkalmazáshasználat fontosságának köztudatban való elterjedésének is.

2.3.2. TraceTogether

Egy másik, az előzőekben bemutatott alkalmazás alapelveire épülő hasonló célt szolgáló szoftver a Szingapúr lakossági körében kontaktkutatást végző szakemberek által kifejlesztett alkalmazás a TraceTogether, melynek célja a Szingapúr lakosainak biztonságának elősegítése, illetve egy esetleges fertőzési lánc megszakításának lehetőségének elérése a kontaktkutatás megvalósítása során. A cél elérését szolgáló gördülékeny működésnek a hátterében szintén a Bluetooth technológia áll, melynek segítségével beazonosíthatók a környéken lévő, szintén az alkalmazást futtató eszközök. Az alkalmazás a Bluetooth technológia által begyűjtött adatokat biztonságosan tárolja a telefonon, illetve ezeket csak abban az esetben osztja meg a kutatást végző szakemberek csoportjával, amennyiben az adott felhasználónál a fertőzést észlelték. Abban az esetben, ha nincs szükség az adatok továbbítására a szervezet számára, 25 nap elteltével az adatok automatikusan törlődnek a telefon memóriájából. Az applikáció hozzáférést igényel a rendszer kamerájának használatához, QR kód leolvasása érdekében, a helyadatok eléréséhez, a Bluetooth beállítások eléréséhez, illetve az eszköz memóriájának tartalmának olvasásához. Az előző alkalmazás tekintetében a QR kód olvasásának lehetősége újdonságként értelmezhető, célja megvalósítani annak az ellenőrzésének a lehetőségét, hogy a felhasználó járt-e olyan helyen, ahol esetleges fertőzöttségi veszélynek volt kitéve [13]. Az alábbiakban látható a TraceTogether alkalmazás illusztrált képernyőfelvételei.

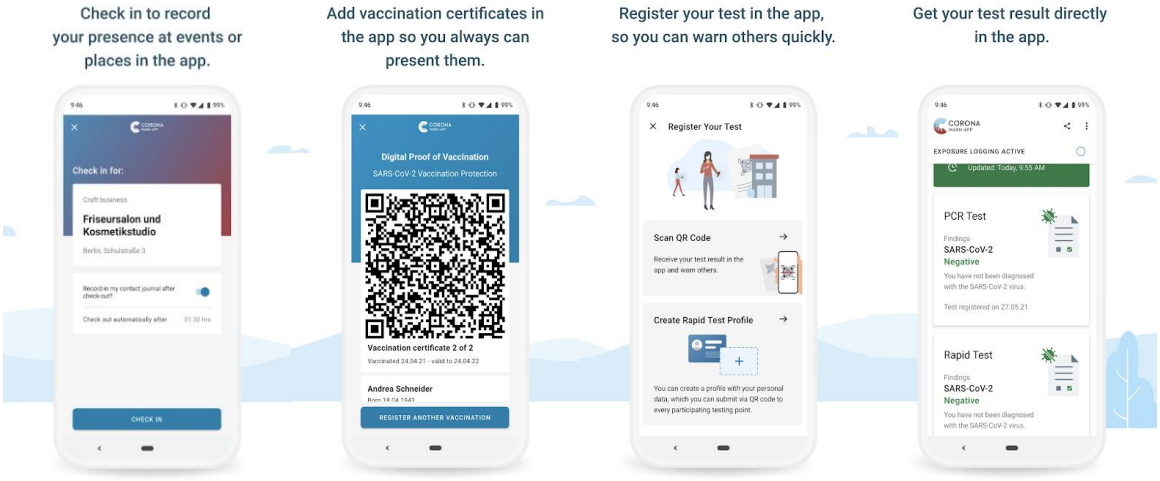


2.4.ábra: TraceTogether alkalmazás illusztrált képernyőfelvételei [14]

Az alkalmazás tehát hasonló alapelven működik, az előzőekben tárgyalt alkalmazáshoz, céljuk is azonos, az emberek biztonságának megőrzése a kontaktkutatás révén.

2.3.3. Corona-Warn-App

Kutatásom során tanulmányozott alkalmazások részét képezte a Németországban fejlesztett Corona-Warn-App néven ismert szintén kontaktkutatás céljából készített alkalmazás. Az előzőekben bemutatott két alkalmazás által is használt technológiát, a Bluetooth-ot alkalmazták az alkalmazás fejlesztésekor, továbbá pedig az Apple és Google Exposure Notification API-kat. Működésében nem tapasztalunk újdonságot, a lényege, hogy az alkalmazásban aktiválva legyen egy úgynevezett naplózás funkció, ami ha engedélyezve van, akkor lehetőséget biztosít a közelben tartózkodó, szintén az alkalmazást használó eszközök Bluetooth alapú beazonosítására, majd az alkalmazások véletlenszerű azonosítójának cseréjére. Ezen azonosítok kizárólag a találkozás időtartamáról és távolságáról szolgáltatnak információkat, fontos, hogy a rendszer nem gyűjt adatokat a felhasználók tartózkodási helyéről. Az értesítések esetenkénti megvalósítása sem változik az előzőekben tárgyaltaktól, amennyiben egy adott személynél fertőzést észleltek, a felhasználó maga dönthet arról, hogy hozzájárul az esetleges fertőzöttségi lánc kialakulásának megtöréséhez, így amennyiben beleegyezik, megoszthatja a telefonja által tárolt adatokat a szakértőkkel, akik ennek tudatában értesíthetik az érintett felhasználókat. Amennyiben erre 14 napon belül nem kerül sor, az addig tárolt adatokat az alkalmazás automatikusan eltávolítja a készülékről [15]. Az alábbiakban látható a Corona-Warn-App alkalmazás illusztrált képernyőfelvételei.



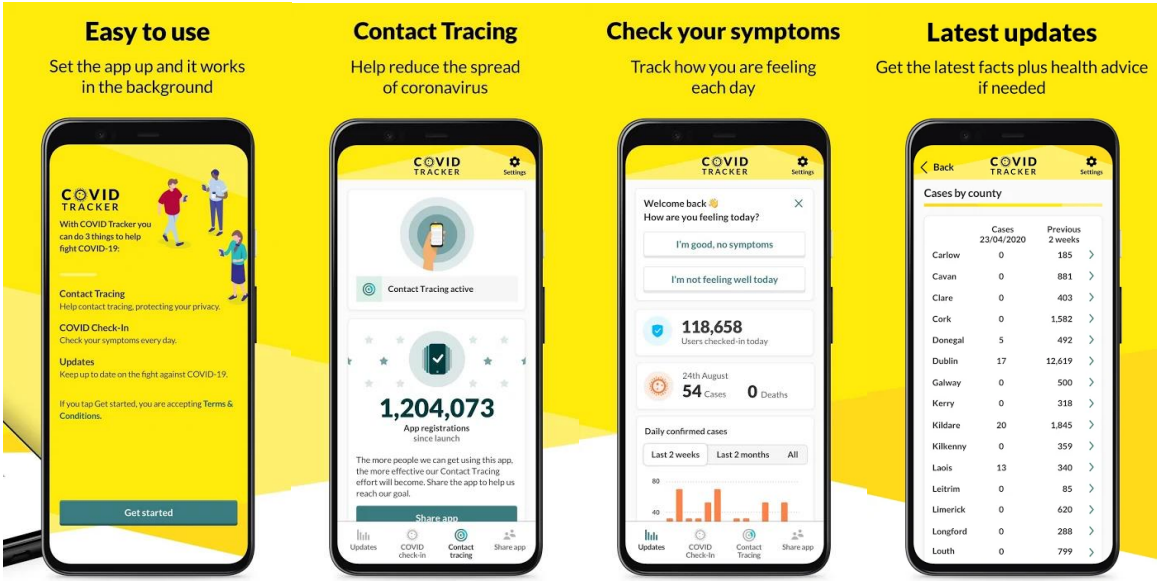
2.5.ábra: Corona-Warn-App alkalmazás illusztrált képernyőfelvételei [15]

Mint ahogy a *2.5.ábrán* is látható, az alkalmazás állandó tovább fejlesztésének köszönhetően, újabb funkciókkal bővült a szoftver, így már lehetővé teszi a felhasználók számára,

hogy az alkalmazáson keresztül értesüljenek teszt eredményeikről, továbbá bizonyíthatják az alkalmazás által esetleges védettségüket egyaránt.

2.3.4. COVID Tracker Ireland

A hasonló alkalmazások témakörében elmélyülve, kutatásom részét képezte egy az előzőekhez hasonló elven működő alkalmazás tanulmányozása, melyet Írországban fejlesztettek a szakemberek, s melynek tárgyát szintén a kontaktkutatás megvalósítása képezte. Mint az előzőekben említett alkalmazások, így a COVID Tracker Ireland alkalmazás is elérhető a Google Play áruházban, a jelenlegi tapasztalatok alapján pedig nagymértékben elősegítette a szakemberek kontaktkutatásban végzett munkáját. Az alkalmazás szintén a Bluetooth technológiát előtérbe helyezve segíti elő a találkozások beazonosítását, illetve véletlenszerű azonosítókat generál a felhasználók számára. A rendszer 2 óránként letölt egy listát, ami véletlenszerű azonosítókat tartalmaz, azon személyek azonosítóit, akiknél az utóbbi időszakban megállapították a fertőzést. Az alkalmazás lényege, hogy amennyiben egy adott felhasználó tudatában van, hogy a fertőzés veszélyét észlelték esetében, akkor, saját döntése alapján megoszthatja, az alkalmazás által generált véletlenszerű azonosítóját a kontaktkutatást végző szakemberekkel, így felkerül az azonosítója, a más felhasználókat értesítő listára. Fontos megjegyezni azt, hogy az alkalmazást úgy tervezték meg, hogy a felhasználók értesítése esetében is a személyes információk teljes biztonságban maradjanak. Az alábbiakban látható az alkalmazásról készült képernyőfelvételeinek illusztrációi [16].



2.6.ábra: COVID Tracker Ireland alkalmazás illusztrált képernyőfelvételei [17]

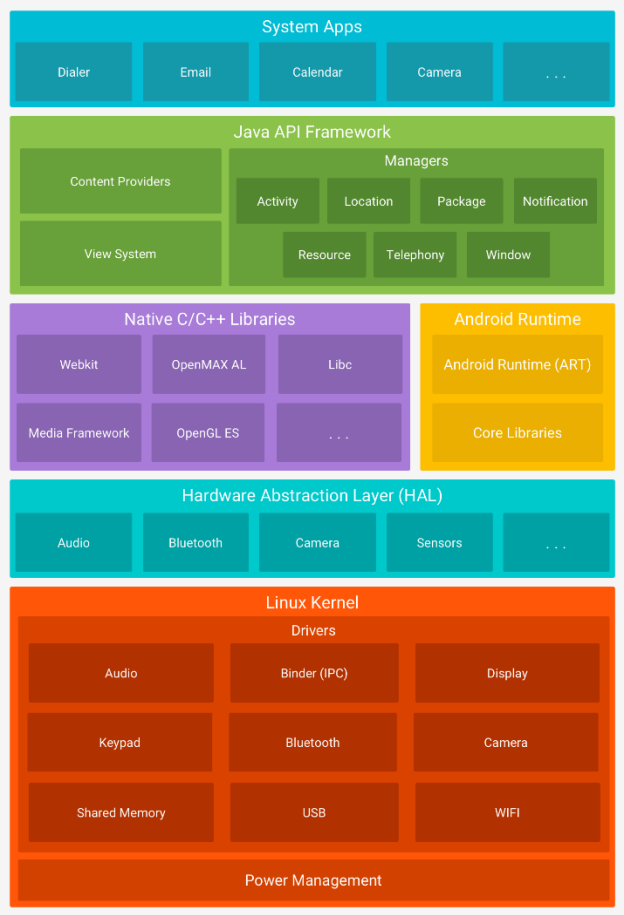
A kutatásom során tanulmányozott alkalmazások alapvető bizonyítékai voltak az általam értelmezett szakirodalmi tanulmányoknak, melyek alkalmazásának tapasztalatai alátámasztották a kontaktkutatás megvalósításának lehetőségeit a telefonokba integrált technológiák felhasználásával, főként a Bluetooth technológia nyújtotta alkalmazási lehetőségeivel.

2.4. Felhasznált technológiák

Az elméleti megalapozás és bibliográfiai tanulmány fejezetének záró, befejező alfejezetét a felhasznált technológiák ismertetése képezi, melyben megfogalmazódnak, az alkalmazás kivitelezését elősegítő technológiák elemzései. Az alábbiakban bemutatásra kerülnek a szoftver megalkotása során alkalmazott technológiák elméleti ismertetései.

2.4.1. Android technológiák

Az Android egy napjainkban igen ismert operációs rendszerként számontartott, Linux kernel alapú, nyílt forráskódú mobil operációs rendszer, mely többnyire, az esetek 80%-ában okostelefonokon működnek, de számos más számítástechnikai rendszer operációs rendszereként ismeretes, így megtalálható táblagépek, televíziók, autókba integrálva egyaránt. Az Android a Google által fejlesztett operációs rendszer és programozási platform, melyet alkalmazásfejlesztési célokra gyakran használnak a szakemberek, annak érdekében, hogy elérjék a mobileszközök felhasználói többségét. Továbbá az Android érintőképernyős felhasználói felületet (UI felületet) szolgáltat az alkalmazásokkal való interakciókhoz a felhasználók számára. Az operációs rendszer és az alkalmazások fejlesztése érdekében ismert az Android SDK, teljes nevén az Android Software Development Kit, magyarul szoftverfejlesztő készlet használata, mely előre megírt kódókat, könyvtárakat, hibakeresőt, eszközemulátort, dokumentációt, mintakódókat és oktató anyagokat tartalmaz. Az SDK segítségével olyan mobil eszközre tervezett alkalmazásokat hozhatunk létre, amelyek képesek kihasználni az Android operációs rendszert futtató eszközök hardver által nyújtott képességeit [18]. Az alábbi *2.7.ábrán* az Android operációs rendszer architektúrája látható, az ábrát pedig annak magyarázata követi.



2.7.ábra: Az Android operációs rendszer architektúrája [19]

Tudva, hogy az Android egy nyílt forráskódú, Linux alapú rendszer, mely számos eszköz számára készült, így az architektúrája, egy szoftver köteget képvisel, melynek az ábrán látható rétegeit különböztetjük meg. Kezdetben, lentről felfele haladva a rétegek között, az első, és egyben az alapréteget a Linux kernel képviseli, amely az eszközök hardverének különböző részeit kezeli. A kernel tartalmaz egy úgynevezett Binder-t, amely a futó folyamatok közötti kommunikáció kezeléséért felelős. A következő, a Linux kernelt követő réteget az úgynevezett Hardware Abstraction Layer, rövidített nevén a HAL képezi. Magyar nevén a Hardver absztrakciós réteg interfészeket biztosít, amelyek az eszköz hardver képességeit a magasabb szintű réteg felé, vagyis a Java API Framework felé tárják. Pontosabban egy hétköznapi eseményt példázva a HAL réteg úgy működik, mint egy fordító, tolmács. Ugyanis tegyük fel, hogy egy számunkra több ismeretlen nyelvet beszélő emberek csoportjában vagyunk és nem ismerjük az összes nyelvet. Lehetőségünk van arra, hogy megtanuljuk az összes nyelvet, így akadálymentesen boldogulnánk a csoportban, azonban egyszerűbb és gyorsabb megoldásnak tekinthető, ha megkérünk egy tolmácsot, hogy fordítson nekünk. Hasonlóképpen működik a Hardver absztrakciós réteg is, ami az úgynevezett

Android Runtime-nak fordít, a különböző hardverek szerint, ugyanis az Android Runtime, rövidítve az ART több különböző hardveren fut, de nincs bármelyik hardver számára személyre szabva. Tovább haladva, a következő réteget két részre osztva elemezhetjük, az egyik része az Android Runtime, a másik a Natív C/C++ könyvtárak. Az Android Runtime egy úgynevezett műhelyként működve lehetővé teszi a Java és Kotlin kódok futtatását az eszközön. A Natív C/C++ könyvtárak pedig számos kódsablonokat tartalmaznak, melyek C, illetve C++ nyelven íródtak. A következő réteget a Java API Framework képezi, melyet olyan Java nyelven írt API-k alkotnak, melyek az Android alkalmazások létrehozásához szükséges építőelemként szolgálnak. Végül pedig a legfelső réteg az alkalmazásokat foglalja magába, ezzel a réteggel kerülnek kontaktusba a fejlesztők és a felhasználók egyaránt [20].

2.4.2. Retrofit

A segítő könyvtárként ismert Retrofit, egy REST kliens könyvtárként számon tartott, a fejlesztők által gyakran használt technológia, melyet HTTP kérelmek létrehozására, illetve válaszok feldolgozására használnak. A Square által fejlesztett könyvtár egy hatékony keretként szolgál az Application Programming Interface, rövidített nevén API-k hitelesítéséhez és a közöttük létrejövő interakciókhoz, valamint hálózati kérések küldéséhez. Ezen könyvtár lehetőséget biztosít a JSON vagy XML, az utólagos verziókban már a SimpleXML és a Jackson típusú adatstruktúrákon alapuló adatok letöltésére is egy webes API-ból.

A Retrofit esetében beszélhetünk a REST Client-ről és a REST API-ról. Az általam tárgyalt esetben a REST Client, annak a Retrofit könyvtárnak a szerepét tölti be, amelyet az úgynevezett kliens oldalon, esetünkben az Android oldalon használnak HTTP kérelmek benyújtására a REST API irányába. Ezzel szemben a REST API olyan lehetőségeket biztosít a fejlesztők számára, hogy azok kéréseket nyújthatnak be, illetve válaszokat fogadhatnak a HTTP, kérés-válasz alapú protokollon keresztül, felhasználva bizonyos metódusokat, amelyet a HTTP protokoll definiál, mint például a gyakran használt GET, POST, PUT és DELETE metódusok [21].

2.4.3. .NET technológiák

A .NET keretrendszer egy olyan új programozási felületet biztosító fejlesztési keretrendszer, mely a Windows nyújtotta szolgáltatásokhoz és API-khoz biztosít programozási felületet, integrálva számos olyan technológiát amit a Microsoft bontakoztatott ki. A .NET felület négy elkülöníthető termék csoportot foglal magába. Az első ilyen csoportot az úgynevezett Development tools and libraries, magyarul a Fejlesztési eszközök és könyvtárak néven

ismerhetjük. Ez a csoport egy átfogó osztály könyvtárként jellemezhető, webes szolgáltatások, Windows alkalmazások létrehozására. Egy másik csoportot az angolul Web services, avagy Webes szolgáltatások csoportja képezi, mely webes szolgáltatások lehetőségét nyújtja a fejlesztők számára. A harmadik csoportot a Specialized servers jelenti, ami speciális szerverek csoportját definiálja, mint például SQL Server, Exchange Server, BizTalk Server. A negyedik és egyben az utolsó csoportot a Devices, az Eszközök csoportja képezi, ami új, .NET kompatibilis nem-PC eszközök csoportját képezi, a mobil telefonoktól a játék vezérlőpultokig [22].

2.4.4. Microsoft SQL

A Standard Query Language kifejezés rövidítéseként elnevezett SQL egy strukturált, szabványos lekérdezési nyelv, mely információk lekérdezését teszi lehetővé az adatbázisból, eleget téve az ANSI szabványok előírásainak [23].

A Microsoft SQL Server a Microsoft által fejlesztett relációs adatbázis-kezelő rendszer, amelyet olyan szoftvertermékként ismerhetünk meg, amelynek elsődleges funkciója az adatok tárolása, vagy azok visszakeresése más szoftveralkalmazások elvárásai szerint [24].

3. A rendszer specifikációi és architektúrája

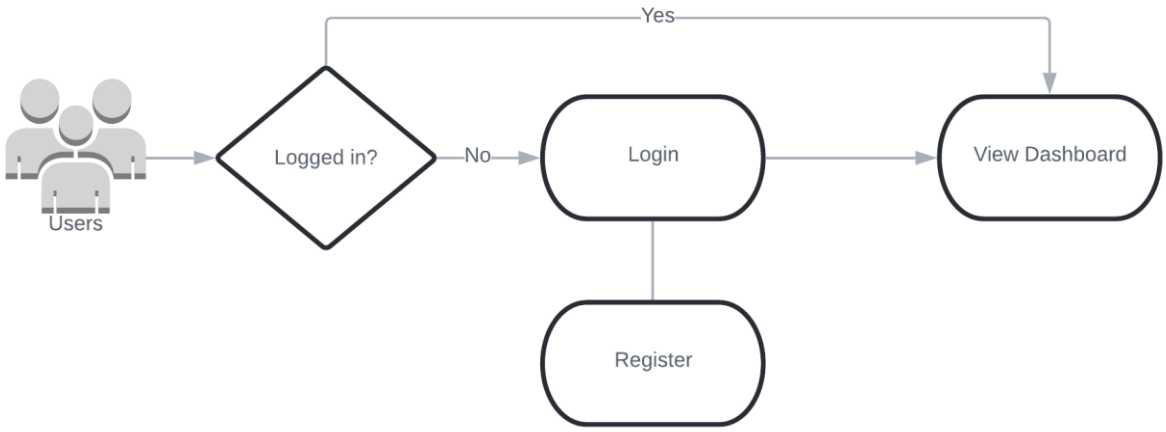
A kutatásom során tanulmányozott szakirodalmi tanulmányok elemzése során, illetve az elméleti megalapozásnak köszönhetően egy pontosan körülhatárolható képet alkothatunk a választott téma megvalósításának lehetőségeiről a technológia által megalapozott lehetőségek felhasználásával. A témában való elmélyülést követően, a rendszer specifikációinak és architektúrájának a bemutatására kerül sor, amelyet ezen fejezet keretén belül tárgyalunk.

A rendszer specifikációinak elemzése során célunk az, hogy érthetően és részletesen értelmezzük és definiáljuk a tárgyalt rendszer működését, a rendszerrel szemben elvárt szolgáltatások meghatározásával, illetve a rendszer üzemeltetésének és fejlesztésének megszorításainak azonosításával. Abból a tényből kiindulva, hogy a követelmények tervezése a szoftvertervezési folyamatnak egy kritikus szakasza, fontos figyelni az ebben a szakaszban véthető hibák elkerülésére, ugyanis ezek elősegíthetik a későbbi rendszertervezési fázisban, illetve az implementációban felmerülő hibák gyarapodásának esélyét. A fejezet az alábbiakban a követelmény dokumentáció megvalósítását, illetve a rendszer architektúrájának elemzését tárgyalja a rendszer specifikációjának eredményeként.

3.1. Felhasználói követelmények

Az úgynevezett követelmény dokumentáció megvalósításának szempontjából kezdetben a felhasználói követelmények, mint magas szintű absztrakt követelmények megfogalmazása kerül bemutatásra, melyet ezen alfejezetben részletesen tárgyalunk.

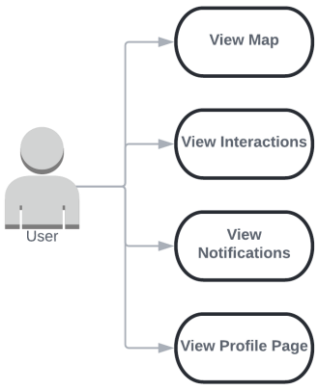
Az alkalmazás használatának érdekében, elsősorban szükséges, hogy a felhasználónak be kell tudnia jelentkeznie, annak érdekében, hogy számára elérhetővé váljanak az alkalmazás által nyújtotta szolgáltatások. A bejelentkezés fontosságát az határozza meg, hogy eszköz váltás esetén is a saját felhasználó fiókját felhasználva segítse elő a felhasználó a kontaktkutatás lehetőségét, elősegítve ezáltal is az így tanulmányozott adatok pontosságának mértékét. A bejelentkezést megelőzően az alkalmazás lehetőséget ad az úgynevezett „Remember Me” opció használatára, ami az előzőekben említett szöveg melletti négyzet bejelölésével válik elérhetővé. Ennek az opciónak a lényege elősegíteni a bejelentkezés gördülékenységét és gyorsaságát, így ennek használatával az alkalmazás megjegyzi a bejelentkezéshez szükséges adatokat, így azt nem kell megadnia a felhasználónak az elkövetkezendő bejelentkezések során. A bejelentkezésnek a folyamatát az alábbi ábra szemlélteti.



3.1.ábra: Felhasználók bejelentkezését ábrázoló diagram

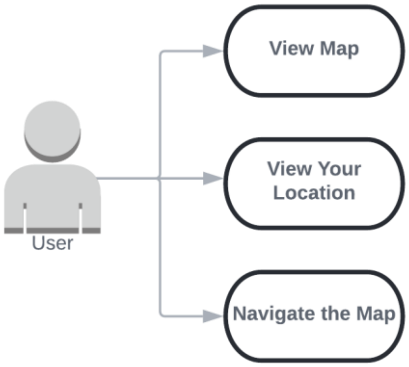
Amennyiben a felhasználó nincs regisztrálva a rendszerben, ezt megteheti a „Don’t have an account? Register here!” szövegre lépve, ami a regisztrációs oldalra navigálja a felhasználót. A regisztrációhoz szükséges egy e-mail cím, jelszó, illetve felhasználónév megadása. Sikeres regisztrációt követően, újra a bejelentkezési oldalra kerül a felhasználó, ahol végre tudja hajtani a bejelentkezést. A bejelentkezést követően a felhasználó számára elérhetővé válik a főoldal, amelyen egy térkép válik láthatóvá, illetve a felhasználónak a jelenlegi pozíciója, tartózkodási

helye. A főoldal megjelenésével elérhetővé válik a felhasználók számára egy navigációs menü, melynek használatával az alkalmazás különböző oldalaira navigálhat a felhasználó. Az alábbi ábra a bejelentkezést követő navigációs menüpont általi kezelőfelületet és az általa nyújtott további lehetőségeket ábrázolja.



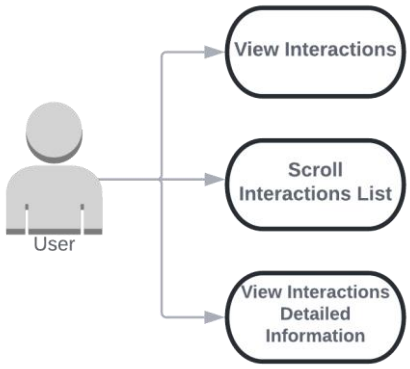
3.2.ábra: A felhasználó által használt navigációs felület által nyújtott lehetőségek

A *3.2.ábrán* egy bejelentkezett felhasználónak az alkalmazásban adatott lehetőségei figyelhetők meg. A navigációs felület használatával a felhasználó az előzőekben szemléltetett oldalakon szabadon navigálhat, mindezt négy ponton, így az első menüponton, amely egyben a kezdeti oldalt képezi a bejelentkezést követően egy térkép válik láthatóvá a felhasználó számára, melyen saját tartózkodási helyzetét követheti nyomon az alkalmazás felhasználója. A kezdeti oldal által nyújtott lehetőségeket a *3.3.ábra* szemlélteti.



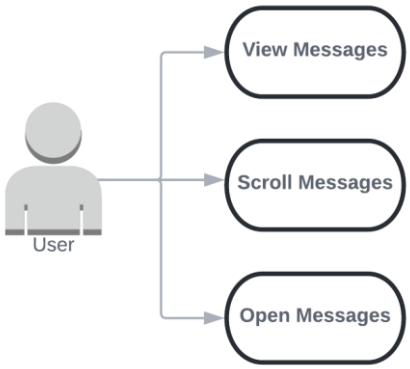
3.3.ábra: Az Elhelyezkedés menüpont által nyújtotta lehetőségek a felhasználó számára

A második menüponton az alkalmazás által érzékelt kontaktusok, és azok időpontjai követhetőek. Ennek a megjelenítésnek a célja mindössze egy általános képet alkotni arról, hogy az adott személy milyen gyakran kerül kontaktusba más személyekkel. Ezen a menüponton a felhasználónak lehetősége van megnézni, görgetni a találkozásokat alkotó listát, illetve egy kontaktusra lépve, megtekintheti annak részletesebb információit is, ami esetünkben az időponttal egészül ki. A kontaktusokat listázó menüpontnak a felhasználó számára biztosított lehetőségeit az alábbi ábra foglalja össze.



3.4.ábra: Az Érintkezések menüpont által nyújtotta lehetőségek a felhasználó számára

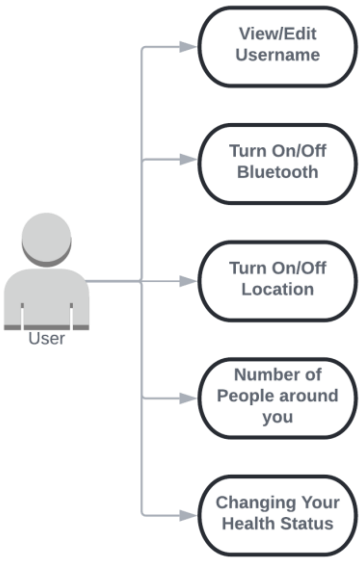
A navigációs menü harmadik pontját a rendszer által küldött értesítések, illetve esetenként figyelmeztetések alkotják. Az alábbi ábrán az értesítéseket tartalmazó oldal által nyújtott lehetőségek láthatóak.



3.5.ábra: Az Értesítések menüpont által nyújtotta lehetőségek a felhasználó számára

A *3.5.ábra* alapján a felhasználónak lehetősége van megnézni és görgetni az értesítéseinek a listáját, illetve megnyitni az adott értesítést, ezáltal részletesen, teljes formájában elolvasni azt.

Végül pedig a negyedik menüpontot a profil oldal képezi, ahol hasznos információk, illetve fontos az alkalmazás működését elősegítő funkciók válnak elérhetővé, mint például a Bluetooth, illetve a GPS aktiválása, illetve amennyiben az adott felhasználónak változik az egészségügyi állapota, azt jelezheti az alkalmazás működését megvalósító szervernek, amely ennek következtében értesíteni tudja az adott felhasználókat. Az alábbiakban a *3.6.ábrán* követhetőek azok a lehetőségek, melyeket a profil oldal nyújt a felhasználók számára.



3.6.ábra: A Profil oldal által nyújtott lehetőségek a felhasználók számára

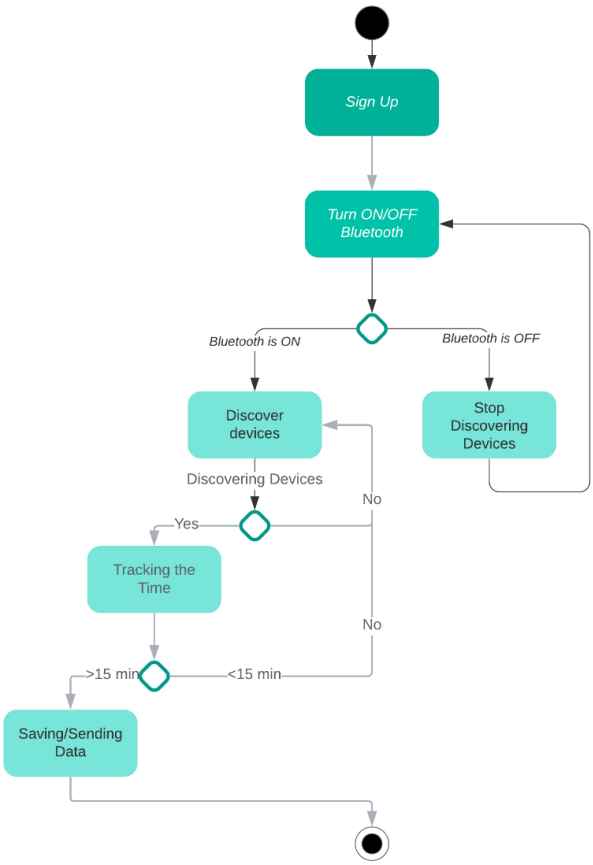
Az alkalmazás működését elősegítő funkciók mellett, a felhasználónak lehetősége van személyre szabott, saját felhasználónevet beállítani számára. Mindez nem kötelező, amennyiben ezt nem kívánja megtenni, akkor a rendszer alapvetően az általa használt eszköznek a Bluetooth azonosítóját állítja be felhasználónévnek. Továbbá a felhasználónak lehetősége adódik információt szerezni arról, hogy hány, szintén az alkalmazást használó személy tartózkodik a közelében. Fontos megjegyezni azt, hogy az alkalmazás nem igényli a felhasználótól, hogy felhasználónevet adjon meg, ezért már alapból a Bluetooth azonosítóját állítja be alapértelmezett felhasználónévnek. A rendszer mindössze lehetőséget nyújt a felhasználó számára, hogy amennyiben szeretné saját személyre szabott felhasználónevet állíthasson be magának. Hasonlóképpen az alkalmazás működése érdekében elvárt funkciók használatára sem kötelezi a rendszer a felhasználót, amennyiben a felhasználó ezeket mégis igénybe venné, akkor automatikusan engedélyt kér az alkalmazás a kiválasztott funkciók bekapcsolására, illetve használatára.

3.2. Rendszerkövetelmények

A követelmény dokumentum, mely egyben a rendszer specifikációját képezi, egyik legfontosabb alkotóeleme a felhasználói követelmények mellett a rendszerkövetelmények, melyet ezen alfejezetben értelmezünk. A rendszerkövetelmények keretén belül olyan követelmény leírásokat említünk meg, melyek a rendszer működési megszorításaira, funkcióira és szolgáltatásaira fektetik a hangsúlyt. Ebben a fejezetben tárgyalt rendszerkövetelményeket két csoportra oszthatjuk, így külön elemzésre kerülnek a funkcionális követelmények, melyek célja összefoglalni azt, hogy pontosan mit képes teljesíteni a rendszer, illetve részletesen bemutatásra kerülnek a nem funkcionális követelmények, megszorítások, melyek célja a termékkel kapcsolatos követelmények megfogalmazása.

3.2.1. Funkcionális követelmények

A rendszer funkcionális követelményei alatt azon követelmények fogalmazódnak meg, melyek előtérbe helyezik a rendszertől elvárt funkciókat, illetve azok elvárt működését. Az előzőekben megfogalmazottakra alapozva az alkalmazás funkcionális követelményeinek egyik jellemzője, az, hogy a rendszer helyes működésének érdekében a felhasználóknak lehetőségük legyen a rendszer alapját képező Bluetooth technológia elindítására, az alkalmazás általi használatának engedélyezésére. A Bluetooth technológia használatának köszönhetően a rendszernek képesnek kell lennie az alkalmazással szemben elvárt egyik legfontosabb, alap funkcionalitást megvalósítani, mégpedig a kontaktok felderítését és azok elmentését, annak érdekében, hogy amennyiben szükséges bizonyos felhasználókat a rendszer értesíteni tudjon. Az alkalmazás egyik legfontosabb funkcióját, a kontaktok felderítését a rendszer működésének segítségével, az alábbi ábra részletesen szemlélteti.



3.7.ábra: A kontaktusok felderítésének folyamata a Bluetooth technológia segítségével

Amint azt a *3.7.ábra* is szemlélteti a bejelentkezést követő első lépés a Bluetooth technológia aktiválása, illetve annak felhasználásának engedélyezése az alkalmazásban. Amennyiben az aktiválás sikeresen megtörtént a rendszer jelenleg manuálisan végzi az eszköz hatókörében levő más mobil rendszerek felderítését. Abban az esetben, ha a felhasználó által használt rendszer más eszközök jelenlétét érzékeli, szükséges, hogy a rendszer figyelje az érzékelés kezdetének az időpontját, illetve az érzékelés megszűnésének időpontját, egy úgynevezett kezdeti időt és egy befejezési időt. Ebből a két értékből a rendszer eldöntheti azt, hogy a találkozást szükséges elmentenie, vagy az nem tett eleget egy kontaktus kialakulásának a feltételeinek. Amennyiben a két időegységből számolt érték meghaladja a 15 percet, akkor a rendszer ezt az adatot egy potenciális kontaktus kialakulásának tekintheti, illetve elmentheti, valamint elküldi azt a szervernek. Ha a számolt érték nem éri el a 15 perces időkorlátot, abban az esetben az adatok nem mentődnek el, ugyanis a rendszer úgy ítéli meg, hogy az nem számított kontaktusnak, így tovább folytatja a folyamatos felderítést. Az alkalmazás jelenlegi verziójában a

rendszer eltekintve a 15 perces időkorláttól, figyelembe veszi és elmenti az eszköz által érzékelt kontaktusokat.

Az érintkezések felderítését a rendszernek a Bluetooth technológia segítségével kell megvalósítania, így állandóan figyelve a közelben levő felhasználók Bluetooth azonosítóit. Tudva azt, hogy a Bluetooth alapján az eszköz egy olyan távolságú hatókörben észleli a többi hatókörében elhelyezkedő készüléket, hogy az már megfelelhet egy potenciális kontaktusnak, így a rendszernek már csak arra kell figyelnie, hogy számon tudja tartani azt az időegységet, ameddig az eszköz észlelte egy másik felhasználónak a jelenlétét, illetve, hogy csak abban az esetben kerüljön elmentésre az adott eset, amennyiben az ténylegesen megfelel a kontaktus értelmezésének, tehát minimum 15 percnek. A rendszernek továbbá képesnek kell lennie a találkozásokból származó, kontaktusoknak számító adatoknak az elmentésére, illetve azok megjelenítésére, az adott felhasználó Bluetooth azonosítója alapján, annak érdekében, hogy a felhasználó követni tudja kontaktusainak számát és időpontját. Fontos, hogy a felhasználók számára ne kerüljenek nyilvánosságra azon személyek személyes adatai, akikkel kontaktusban voltak, ezt a célt szolgálja az is, hogy a megjelenített információk közt a felhasználók Bluetooth azonosítója szerepel mindössze, illetve a találkozásnak az időpontja. Ennek a funkciónak a célja egy általános képet nyújtani a felhasználók számára, hogy találkozásaikból adódóan, milyen gyakorisággal kerülhettek esetleges veszélynek számító kontaktusba más személyekkel, mindezt úgy hozva a felhasználók tudatára, hogy biztosítva legyenek más felhasználók személyes adatainak a biztonsága. A rendszertől elvárt funkcióinak egyike az, hogy amennyiben a felhasználó egészségügyi állapotának megváltoztatásával hozzájárul a kontaktkutatás lehetőségéhez, a rendszernek képesnek kell lennie az esetlegesen érintett, veszélyeztetett felhasználók számára figyelmeztető értesítést küldeni, tudatva azt, hogy egészségügyi állapotára nézve veszélyeztetve volt egy bizonyos találkozás következtében. Megalapozva azt a követelményt, hogy a felhasználó saját pozícióját a nagyobb felhasználói élmény érdekében egy térképen követni tudja, a rendszernek mindenekelőtt lehetőséget kell nyújtania, az ennek a funkciónak a megvalósítására felhasznált technológia engedélyezésére az alkalmazásban, így megvalósítva az engedélykérés alapú helyadatok használatát, a GPS technológiát. Ezen technológia alkalmazásának segítségével a rendszernek képesnek kell lennie folyamatosan valós adatok alapján megjelölni a térképen a felhasználó pillanatnyi tartózkodási helyét.

3.2.2. Nem funkcionális követelmények

Az előző fejezetben a funkcionális követelmények keretén belül bemutatásra kerültek a rendszer által biztosított specifikus funkciók, melyek alapelvárásként szolgáltak a szoftverrel szemben. Az alábbi fejezetben, az előzővel ellentétben az úgynevezett nem funkcionális követelmények fogalmazódnak meg. A nem funkcionális követelmények elemzésekor a rendszertulajdonságok bemutatására fektetjük a hangsúlyt, így főképp az adott rendszer megbízhatóságára, tárfoglalására, válasz idejére, illetve annak működési szempontból értelmezett rugalmasságára reflektálunk.

Kezdetben a termékre vonatkozó tulajdonságok kerülnek bemutatásra, illetve a rájuk vonatkozó követelmények, melyeknek célja egy összetett képet ábrázolni a termék viselkedéséről. Az alkalmazás hibátlan működése érdekében mindenekelőtt szükségünk van egy okostelefonra, amelyen Android operációs rendszer fut, ugyanis a szoftver kimondottan az Android-os eszközök területét célozta meg, így a rendszer minden funkciója az ezen operációs rendszert futtató eszközök tulajdonságait tartja szem előtt. Az alkalmazás zavartalan és egyben gördülékeny működése érdekében az adott készülék minimum 7.0-ás Nougat, Level API 23–as támogatottságú Android operációs rendszerrel kell rendelkezzen. Az alkalmazás tervezésekor fontos szempont volt, hogy egy a régi rendszerekkel szemben újabb verziójú rendszer legyen az alapkövetelmény, azonban úgy kiválasztva ezt, hogy a napjainkban használt okostelefonoknak nagyobb részén az alkalmazás működőképes legyen, ugyanis célunk egy nagyobb felhasználócsoport elérése. Így ez a jelenlegi verzió az Android eszközök körülbelül, mintegy 73.7% -án zavartalanul fut. Bár napjainkban szinte bármely eszközben megtalálható a Bluetooth, illetve a GPS technológia implementálva, fontos, hogy az általunk használt eszköz támogassa ezen technológiák használatát, tehát, hogy képes legyen a közelben levő, szintén Bluetooth technológiát használó eszközök felderítésére, illetve a GPS alapú helymeghatározásra. Elengedhetetlen továbbá a rendszer általi internethasználat támogatottsága, Wi-Fi, illetve Mobiladat-használat által egyaránt, ugyanis az adatok elmentésének, illetve az értesítések küldésének és fogadásának zavartalan megvalósítása szempontjából az alkalmazás igényli az állandó internet elérhetőséget. A jelenlegi verzió telepítéséhez szükséges körülbelül 25MB szabad tárhely, illetve az eszköznek rendelkeznie kell minimum 1 GB RAM memóriával az applikáció zökkenőmentes futása érdekében.

Amennyiben a cél eszköz az alkalmazás elvárt igényeit teljesíteni tudja, a rendszer egy állandóan frissített adatok feldolgozását képes megvalósítani, illetve azokat elmenteni, amennyiben szükséges. Az adatok frissítését tehát a rendszer pillanatok alatt képes elvégezni,

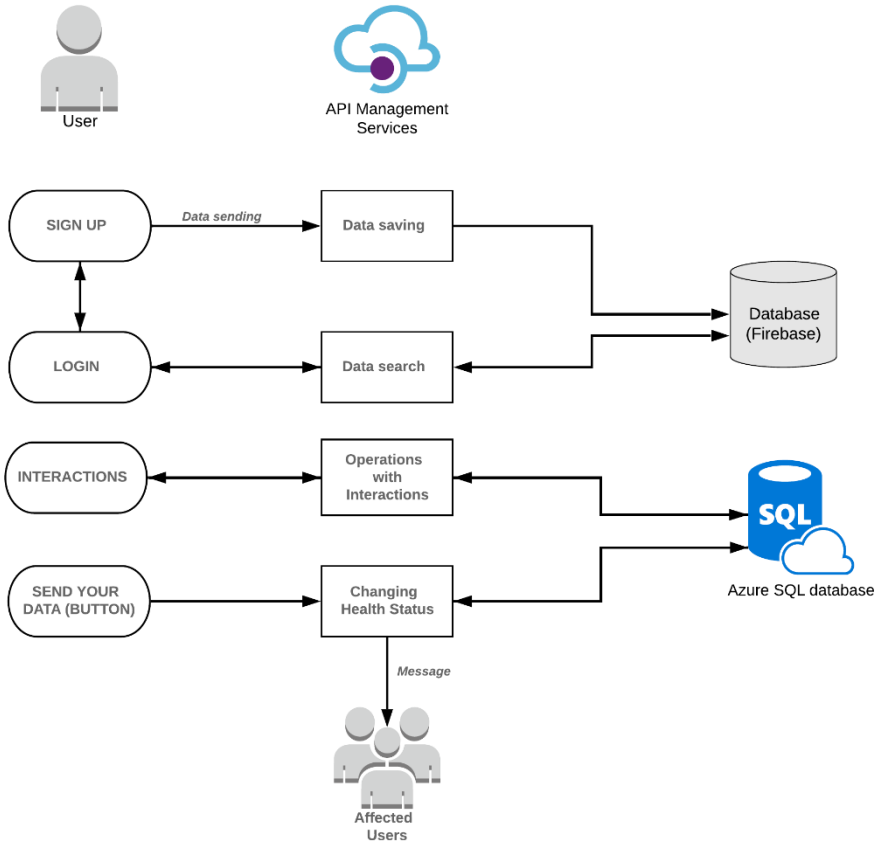
abban az esetben pedig, ha a felhasználónak megváltozott egészségügyi állapota, rövid időn belül képes megkeresni az érintett felhasználókat, a Bluetooth azonosítójuk alapján, majd ezután értesíteni őket, tudatva velük, hogy veszélyeztetve voltak.

Az alkalmazás megvalósításakor folyamatosan előtérben voltak bizonyos külső követelmények, melyekre esetünkben elengedhetetlen volt odafigyelni. Ilyen külső követelménynek feleltek meg a személyes adatok védelmére vonatkozó, úgynevezett GDPR követelmények. Célunk egy olyan alkalmazás megvalósítása volt, amely minimális személyi adatot vár el a felhasználóktól, hogy ezáltal is növelhető legyen a felhasználók és a rendszer közötti megbízhatóságnak a mértéke. Ennek érdekében sikerült úgy megvalósítani a kontaktusok feltárását, hogy a szoftver csak az eszközök Bluetooth azonosítója alapján képes legyen azonosítani a találkozásokat. Az alkalmazás célja ugyanis azonosítani az esetlegesen felmerülő veszélyt, elkerülve a személyek helyzetének azonosítását, illetve azt, hogy kivel találkozott, és hogy kinek a jelenléte képezhette esetlegesen a veszély forrását. Ennek érdekében az alkalmazás semmilyen személyes adatot nem közöl a felhasználóval, így az érzékelt kontaktusokat is csak Bluetooth azonosítókkal jeleníti meg, annak érdekében, hogy ne lehessen következtetni a felhasználók személyére. Az alkalmazás által nyújtott helymeghatározási funkciónak, nem szükségszerű a használata, ugyanis az alkalmazás gondtalanul működik annak aktivitása nélkül is, célja csak a felhasználói élmény növelésében van, így minden felhasználó egyénileg dönthet annak használatáról. Amennyiben a felhasználó élni szeretne az alkalmazás ezen lehetőségével, a térképen figyelemmel követheti jelenlegi pozícióját. Ennek a funkciónak a kivitelezésére a rendszernek bár szüksége van a személy helyzetét megjelölő adatokra, azonban ezeket a rendszer nem használja fel más célokra, mindössze a pozíció megjelenítésére, biztosítva ezáltal is a felhasználók adataink biztonságát. Bár a kontaktusok felderítése érdekében elegendő a rendszer számára a Bluetooth azonosítókat ismerni, azonban az értesítések szempontjából szükséges a felhasználóknak egy e-mail címet is megadniuk, amelyen keresztül értesítve lesznek a felmerülő veszélyekről. Így egyetlen személyes adat, melyet elvár az alkalmazás, az egy e-mail cím, azonban ennek biztonságára is odafigyel a rendszer, ugyanis ezt sehol nem jeleníti meg az alkalmazáson belül más felhasználók számára, mindössze arra használja fel, hogy ezen keresztül értesítéseket küldjön azoknak a személyeknek, akiknek szükséges. Az alkalmazás alapból a felhasználó által megadott felhasználónevet jeleníti meg, továbbá lehetőséget nyújt ennek megváltoztatására egy az alkalmazást használó személy által megadott felhasználónévre. Amennyiben a név megváltoztatása bekövetkezik, a rendszer mindössze a felhasználó számára jeleníti azt meg, nem

használja fel más célokra, illetve nem teszi láthatóvá azt más felhasználók számára sem, biztonságban tartva a személyes adatokat.

3.3. A rendszer architektúrája

Az előzőekben két alfejezetre lebontva bemutatásra kerültek a rendszer specifikációi, melyeket a felhasználói követelmények, illetve a rendszerkövetelmények alfejezetekben részletesen tárgyaltunk. A továbbiakban pedig a rendszer architektúrája kerül bemutatásra, amelynek értelmezése kiemelkedő fontosságú, ugyanis ez képezi a rendszer alap vázát, melynek alapján elkészült és üzembe helyezhetővé vált a kontaktkutatás megvalósítására tervezett alkalmazás. A rendszer architektúráját és funkcionalitásainak működését a *3.8.ábra* részletesen szemlélteti.



3.8.ábra: A rendszer architektúrája

Mint látható a rendszer architektúrájának fő komponensei a felhasználók által használt kliensoldali alkalmazás, az API menedzsment szolgáltatásai, illetve az adatok tárolására használt adatbázis, melyből az alkalmazás jelenlegi verziója kettőt használ. Az egyik adatbázist a Firebase [25] nevű

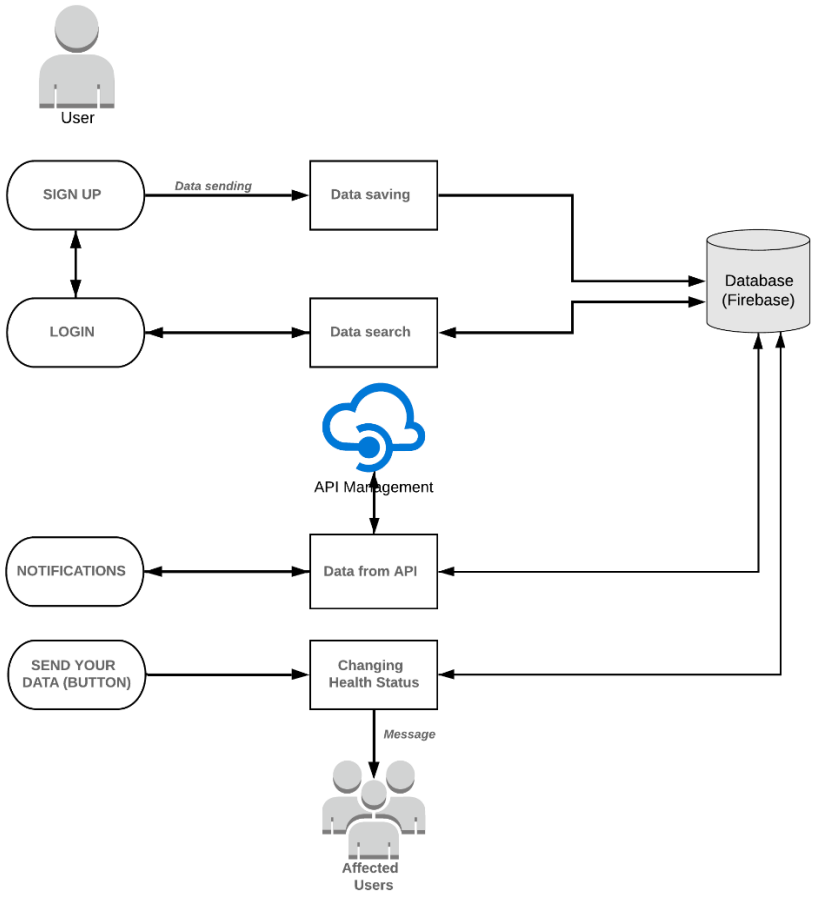
adatbázis képviseli, melyben a felhasználókra vonatkozó adatok tárolása történik. A második adatbázist az Azure [31] webszerver által felhasznált felhő alapú SQL adatbázis képviseli, melyben a rendszer által észlelt kontaktusok elmentése történik.

4. Részletes tervezés

A rendszer specifikációja és architektúrájának bemutatása során, más szóval élve a követelménytervezés fázisában részletesen bemutatásra került a rendszer működése és architektúrájának elemzése. Ennek eredményeként a követelmény dokumentum megalkotására került sor, amely tulajdonképpen a rendszer specifikációját jelenti.
Az alábbi fejezetben a célkitűzések általi rendszer részletes tervezése, illetve megvalósításának lépései kerülnek bemutatásra, külön tervezési fázisokra felosztva.

4.1. Első tervezési fázis

Az első tervezési fázisban már egyértelművé vált számomra, hogy a rendszer egészét több rendszerkomponens fogja alkotni, melyeknek különálló feladati lesznek, illetve bármely komponens helyes működése elengedhetetlen a rendszer egészének megalkotása érdekében. Kezdetben nyilvánvalóvá vált az a tény, hogy a tervezett rendszernek két fő komponensből fog kelleni felépülnie, így két különálló részre lesz osztható. Beszélhetünk egy úgynevezett Frontend részről, ami a tulajdonképpeni felhasználói felületet képezi, és amely megvalósítja a rendszer és a felhasználó közötti interakciót. Ez lesz az a része a rendszernek, amely elérhetővé teszi a felhasználók számára a másik rész, az úgynevezett Backend rész által elvégzett munka eredményét. Végül pedig abból a tényből kiindulva, hogy a rendszertől elvárjuk, hogy folyamatosan adatok feldolgozását tegye lehetővé, ezért szükség lesz egy adatbázisra is, ahol az általunk feldolgozott adatokat a rendszerünk tárolni tudja, és szükség esetén azokat felhasználni. Megalapozva a tervezési folyamatnak az első fázisát, létrehoztam egy a rendszer architektúráját értelmező ábrát, mely a rendszer alapját képezte. A rendszer architektúrájáról készült első ábrát az alábbiakban szemléltetem.



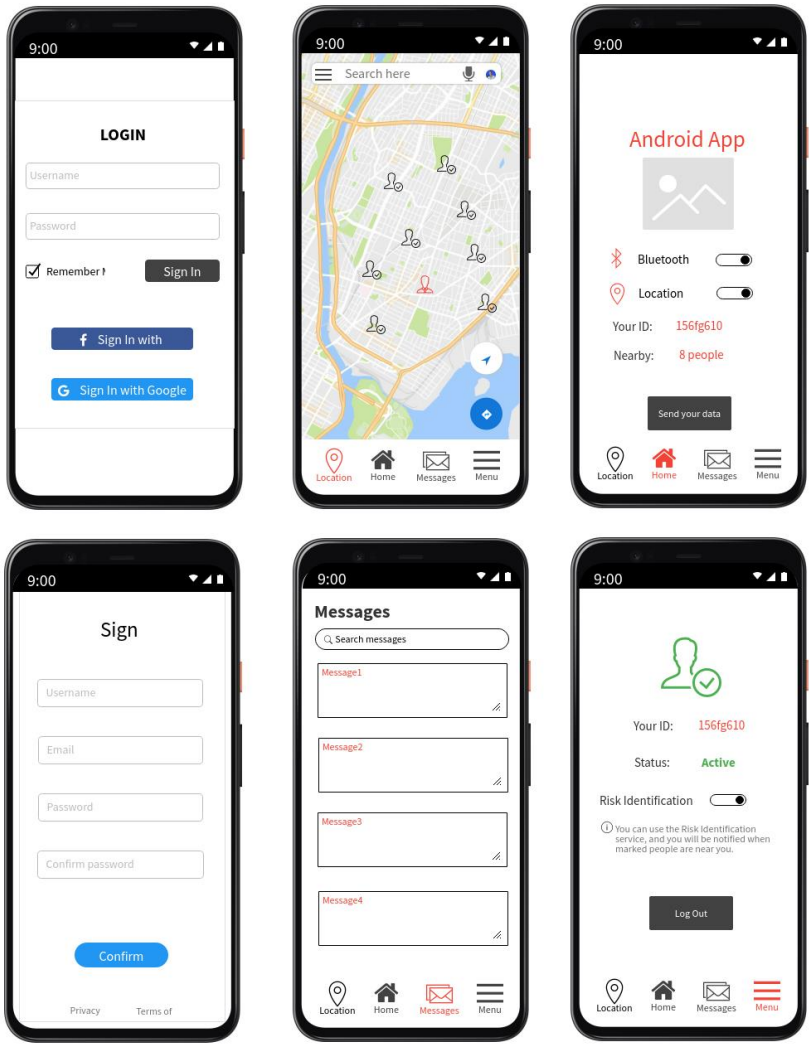
4.1.ábra: A rendszer első tervezési fázisában készült architektúrája

Az első architektúrának az ábrázolása során fény derült arra, hogy három különböző komponens működésére kell odafigyelni a rendszer megalkotása során. Ezen három rendszerkomponens pedig a felhasználói felület, az alkalmazás szervere, illetve az adatbázis.

Kezdetben a működést megvalósító rész a Backend bemutatására kerül sor, melyet az első tervezési fázisban felhasználtam, illetve melynek aktív résztvevői az alkalmazás szervere, illetve az adatbázis. Az első lépések során egyértelművé vált az adatbázis, illetve a felhasználói felület feladatköre. A cél az volt, hogy a felhasználói felület az adatbázisban tárolt adatokat, illetve azok feldolgozásával különböző funkciókat tegyen elérhetővé a felhasználó számára. Azonban ezen két komponens között szükséges elhelyezkednie egy harmadik rendszerkomponensnek is, amely vezérli a felhasználói felület, illetve az adatbázis közti kommunikációt. A saját szerver megtervezése és megalkotása előtt egy bárki számára elérhető API-t használtam fel, annak érdekében, hogy megvalósítsam a kezdeti lépéseket egy már meglévő szerverként üzemelő API és az Android-os alkalmazás között. Az általam felhasznált API, az a Coronavirus COVID19 API

[26], amely egy általános adatokat tartalmazó, bárki számára elérhető API, mely lehetővé tette a HTTP kérés-válasz alapú protokoll alapján a kommunikáció megvalósítását az API és az alkalmazás között, a protokoll metódusainak köszönhetően. Az alkalmazásban a GET metódust felhasználva az API által különböző adatokat sikerült megjeleníteni. Ennek megvalósítása érdekében használtam az úgynevezett Retrofit Service könyvtárat, mely segítségül szolgált a kommunikáció megvalósításában az API és a kliensoldali alkalmazás között, mindezt a HTTP kérelmek benyújtásának lehetőségével biztosítva a REST API irányába [21].

Az adatbázis és a felhasználói felület közti kommunikáció megalapozása érdekében kezdetben a Firebase [25] nevű valós idejű adatbázis által nyújtotta lehetőségeket használtam fel. Abból kiindulva, hogy a kontaktusokat ebben a tervezési fázisban még nem kellett a rendszernek elmentenie, hanem az API-nak köszönhetően elérhetővé vált, ezért az adatbázist mindössze a felhasználóra vonatkozó adatok elmentésére alkalmaztam. A felhasználónak a regisztrálási adatai, illetve az egészségügyi státusza került elmentésre. Az egészségügyi állapotát egy felhasználónak az alkalmazás alapértelmezetten egészségesre állítja. Később pedig sikerült megvalósítani, a gombnyomás általi egészégügyi státusznak a megváltoztatását, már nem csak alkalmazás szinten, hanem adatbázis szinten is. Ennek köszönhetően, sikerült megvalósítani egy Android-os alkalmazás és egy valós idejű adatbázis közti kommunikációnak a lehetőségét. Megtapasztalva egy Android-os felhasználói felület és egy API, valamint egy adatbázis közötti kapcsolat létrehozását, illetve a köztük levő kommunikációnak a megvalósítását, a következőkben az alkalmazás felhasználói felületének a részletes tervezése következett, melyet többnyire a végleges alkalmazás is szemléltet. Az első tervezési fázisban elkészített felhasználói felület tervrajza meghatározta a későbbi tervezési fázisok folyamatát is. Az alábbiakban a felhasználói felület tervrajzának ábrája látható, majd azt követően annak magyarázata olvasható.



4.2.ábra: Az első tervezési fázisban készített felhasználói felület tervrajzának ábrája

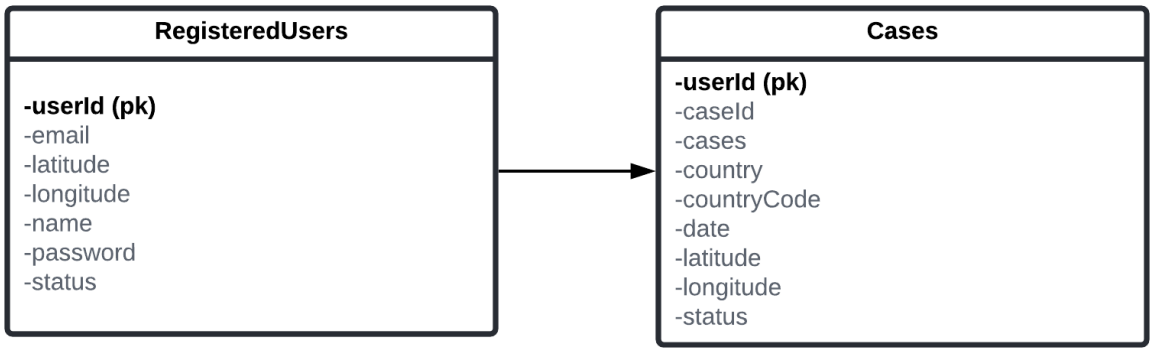
Az *4.2.ábra* a kezdetben elképzelt, felhasználói felületet hivatott ábrázolni, amely útmutatóként szolgált a tervezési, illetve megvalósítási folyamatok során egyaránt. A kezdeti elképzelés alapján a felhasználónak regisztrálnia kell az alkalmazásban, illetve ennek következtében be is kell tudnia jelentkezni a saját profiljára. A bejelentkezés megkönnyítésére egy olyan funkciót képzeltünk el, mely lokálisan képes elmenteni a felhasználó bejelentkezési adatait, megkönnyítve ezáltal a gyakori bejelentkezéseket, ugyanis ezen funkciónak az igénybe vételével, a felhasználónak elegendő csupán egyszer kitöltenie az adatait, továbbá egyszerűen a bejelentkezési gombra lépve, megteheti a bejelentkezést. A bejelentkezést követően egy navigációs menüsort képzeltem el, mely megkönnyítheti a felhasználónak az alkalmazás különböző oldalaira navigálását. A fő oldalnak egy térkép megjelenítése képezte az alap tervet, melynek köszönhetően a felhasználó figyelemmel

követheti pillanatnyi tartózkodási helyét, illetve a környékén levő felhasználókat. A második oldalnak egy általános oldal terve látható, melyen az volt a cél, hogy olyan funkciókat helyezzünk el, mint például a Bluetooth, illetve a GPS technológiák aktiválására szolgáló gombnyomásnak a lehetőségét, abból a tényből kiindulva, hogy ezen technológiák képeznék az alkalmazás működésének fő aspektusát. Terv szerint ugyancsak ezen az oldalon helyezkedne el a felhasználó azonosítója, illetve egy információ arról is, hogy hány más felhasználó tartózkodik pillanatnyilag a közelünkben. Ezt az oldalt zárná végül pedig egy gomb, melynek azt a funkciót kell megvalósítania, hogy amennyiben a felhasználó rákattint, cserélje meg az egészségügyi státuszát, egészségesről, fertőzöttre. Fontos cél volt az is, hogy mindezt az adatbázisban is sikerüljön cserélni, ne csak az alkalmazásban, pillanatnyilag. Az elképzelések szerint ennek a funkciónak a lényege az lenne, hogy amennyiben az adatbázisban megváltozik egy felhasználó állapota egészségesről, fertőzöttre, akkor a szerver megvalósítsa az adott személlyel kontaktusban lévő személyek értesítését az alkalmazáson keresztül. Ennek érdekében lett megtervezve a következő menüpontot képviselő oldal is, amely az értesítéseket tartalmazza. A tervek alapján a felhasználó itt olvashatja el a számára a rendszertől küldött értesítéseket, azonban a viszont válaszolás, illetve üzenet megfogalmazás lehetőségét a rendszer nem támogatja. Végül pedig a záró oldalt egy információkat tartalmazó oldalnak képzelhetünk el, amelyen a felhasználó azonosítója, egészségügyi állapota található, illetve egy olyan gombnyomásra aktiválható funkció, mely aktiválása esetén figyelmezteti a felhasználót a környezetében lévő, számára veszélyt jelenthető felhasználók jelenlétére. Az oldalt a terv szerint egy az alkalmazásból való kijelentkezést megvalósító gomb zárja.

4.1.1. Az első tervezési fázisban megvalósított alkalmazás

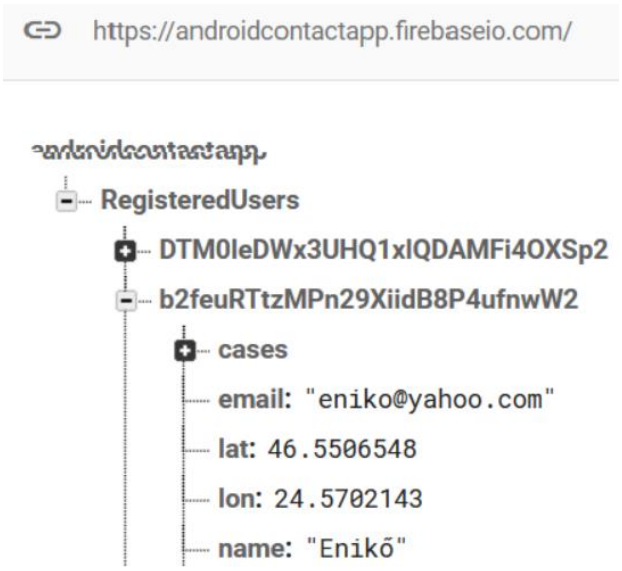
Az első tervezési fázisban elkészített architektúrát, illetve a felhasználói felület tervrajzát követve egy működőképes alkalmazást sikerült megvalósítani, mely ettől a ponttól kezdve a projektem egy alap alkalmazásává vált, ami a tervek szerint erre alapozva fejleszthető tovább, a különböző funkciók működésének kiegészítésével, illetve a már meg lévő funkciók tovább fejlesztésével. Ebben a fejezetben az első tervezési fázisban elkészített alkalmazás bemutatására kerül sor. Amint azt az előzőekben is említettem az alkalmazás kezdetben a Firebase [25] adatbázis szolgáltatásait vette igénybe, ebben az adatbázisban kerültek elmentésre, úgy az alkalmazásban regisztrált felhasználók, mint a hozzájuk köthető esetek. Az adatbázis felépítésének alapötlete az volt, hogy minden regisztrált, az adatbázisba bekerülő felhasználóhoz, eseteket tudjunk hozzárendelni. Ezen eseteket, az általam felhasznált API-ból listázta az alkalmazás, illetve egy

gombra kattintva lehetősége volt jelezni a felhasználónak manuálisan, hogy az adott eset közelében tartózkodott. Ennek köszönhetően az adatbázis folyamatosan tárolta az eseteket, az adott felhasználó adatai közt. Az adatbázis felépítését, a következő egyed-kapcsolat diagram ábrázolja.



4.3.ábra: A Firebase adatbázis egyed-kapcsolat diagramja

Amint azt a *4.3.ábra* is szemlélteti, az adatbázisban két táblát különíthetünk el, egy táblát, mely a regisztrált felhasználókat tartalmazza, továbbá a felhasználókhoz tartozó információkat, e-mail cím, felhasználónév, illetve jelszó, ezeket a regisztráláskor a felhasználónak kell megadnia. Továbbá a rendszer a regisztrálást követően a felhasználó állapotát egészségesre állítja, valamint a felhasználókhoz hozzárendel egy automatikusan generált userId-t, melynek célja, hogy úgy tudjunk hivatkozni a felhasználókra, hogy ne használjuk fel a személyes adataikat. Egy másik táblát az esetek képviselnek, amely tábla a userId-n keresztül kapcsolódik a felhasználók táblához, így ennek köszönhetően az esetek hozzárendelhetők a felhasználókhoz, így megvalósítható az, hogy amennyiben a felhasználó egy adott eset közelében járt, azt jelezvén az alkalmazásban, az adatbázisban hozzárendelődik a saját adataihoz. Ezen tábla információkat tartalmaz az esetekről, így minden esetnek saját azonosítója van, helyszíne, ahol azonosították, időpont, amikor azonosították, illetve állapota is van, meghatározva, hogy igazoltan azonosított esetről van szó. Az alábbiakban egy ábra szemlélteti az adatbázisnak a tartalmát, melyet az alkalmazás fejlesztői láthatnak, az alkalmazás működtetése során.



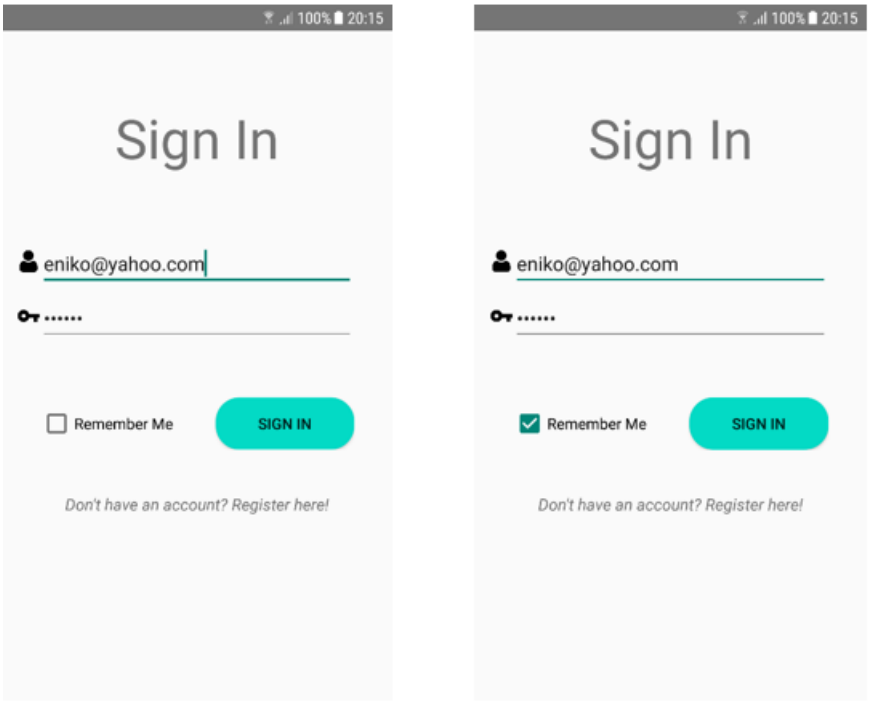
4.4.ábra: A Firebase adatbázis RegisteredUsers tábla tartalmának ábrázolása

Az *4.4.ábrán* látható, hogy a regisztrált felhasználók, egy egyéni azonosítóval rendelkeznek, így ezen azonosító alapján hivatkozni tudunk rájuk, továbbá látható, amint egy adott felhasználóhoz esetek vannak hozzárendelve, az adatai mellett. Az alábbiakban egy ábra fogja szemléltetni azt, hogy hogyan jelennek meg az adatbázisban a különböző esetek, melyek egy felhasználóhoz lettek hozzárendelve.

4.5.ábra: A Firebase adatbázis Cases tábla tartalmának ábrázolása

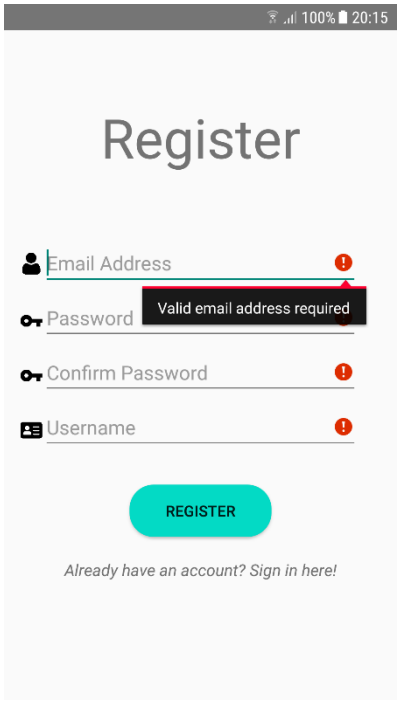
Az *4.5.ábrán* látható egy adott felhasználóhoz hozzárendelt eset, és a hozzá tartozó adatok, valamint a felhasználónak az általános adatai. Megfigyelhető az is, hogy a felhasználókhoz hasonlóan, az esetekhez is egyéni azonosítók lettek hozzárendelve, annak érdekében, hogy ezek

alapján hivatkozhassunk rájuk. Az általam használt API által szolgáltatott esetek adatai a továbbiakban a felhasználói felület elemzése során bemutatásra kerülnek. A továbbiakban pedig a felhasználói felület, vagyis az alkalmazás és annak működésének bemutatásáról lesz szó.
Az alkalmazás elindításakor kezdetben egy bejelentkezési oldal fogadja a felhasználókat, amelyet az alábbi ábra szemléltet.



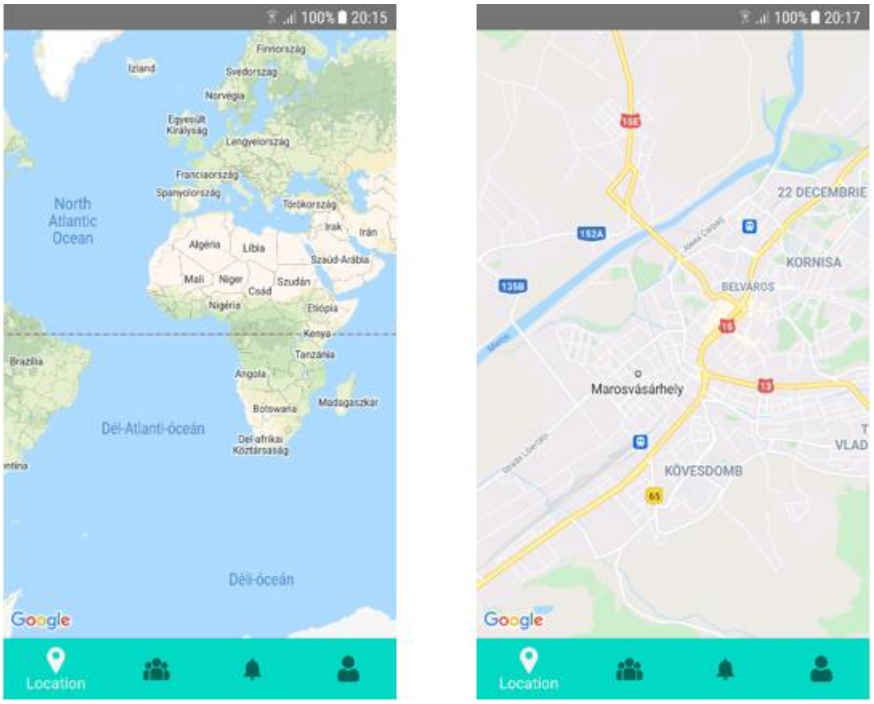
4.6.ábra: A bejelentkezési oldal ábrázolása

Az *4.6.ábrán* látható, hogy a bejelentkezési oldal két szöveg mezőt tartalmaz, melyben a bejelentkezéshez szükséges adatokat várja el az alkalmazás a felhasználótól, pontosabban egy valós e-mail címet, illetve egy minimum 6 karakterből álló jelszót. A jobb oldali képen látható, a bejelentkezés oldal azon funkciója, melyet igénybe véve a felhasználónak elegendő mindössze egyszer megadni az adatait, ugyanis a „Remember Me” négyzetet bejelölve, az alkalmazás lokálisan elmenti a bejelentkezési adatokat, innentől kezdve a felhasználónak már csak a bejelentkezést lehetővé tevő gombot kell igénybe vennie. Abban az esetben, ha az alkalmazás felhasználójának még nincs regisztrált profilja a rendszerben, megteheti a regisztrálást az úgynevezett „Don’t have an account? Register here!” szövegrészre kattintva. Ezt a lehetőséget igénybe véve a regisztrációs oldalra navigálja az alkalmazás a felhasználót, amelyet a *4.7.ábra* szemléltet.



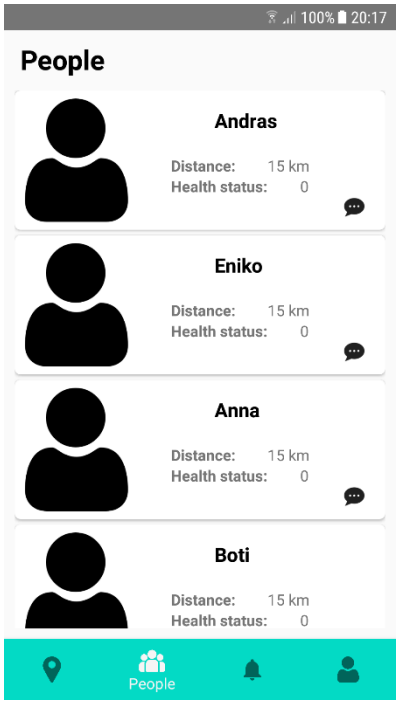
4.7.ábra: A regisztrációs oldal ábrázolása

Ez az ábra a regisztrációs oldal, és annak tartalmát szemlélteti, láthatóak azok a szöveg mezők, melyeket az alkalmazás igényel kitölteni a regisztrálás érdekében. A felhasználónak szükséges megadnia az e-mail címét, a jelszavát, illetve a jelszavát még egyszer, az esetleges hiba lehetőségek elkerülése érdekében, illetve egy felhasználónevet, amit később megváltoztathat majd az alkalmazásban. Amint azt az ábra is szemlélteti, a kitöltendő szöveg mezők érvényesítési protokollt használnak, tehát a rendszer ellenőrzi azt, hogy valós adatok alapján legyen kitöltve a regisztrálási oldal, így szükséges egy valós e-mail címet megadni, egy minimum 6 karakterből álló jelszót, illetve egy egyénileg kiválasztott felhasználónevet. A regisztrálást a regisztrációt megvalósító gombbal véglegesíteni lehet, majd az alkalmazás jelzi, amennyiben sikeres, vagy sikertelen volt-e a folyamat. A regisztrálást követően a bejelentkezésnek a folyamata következik, így az „Already have an account? Sign in here!” szövegre kattintva az alkalmazás a bejelentkezési oldalra navigálja a felhasználót. Amennyiben a kezdeti lépések sikeresen végrehajtódtak, így a regisztrálást és bejelentkezést követően tovább léphetünk az alkalmazásban. Bejelentkezve az alkalmazásba láthatóvá válik a kezdő oldal, illetve egy navigációs menüsor, amelynek célja megkönnyíteni az alkalmazás különböző oldalai közti navigációt. A bejelentkezést követően az alábbi oldal fogadja a felhasználót.



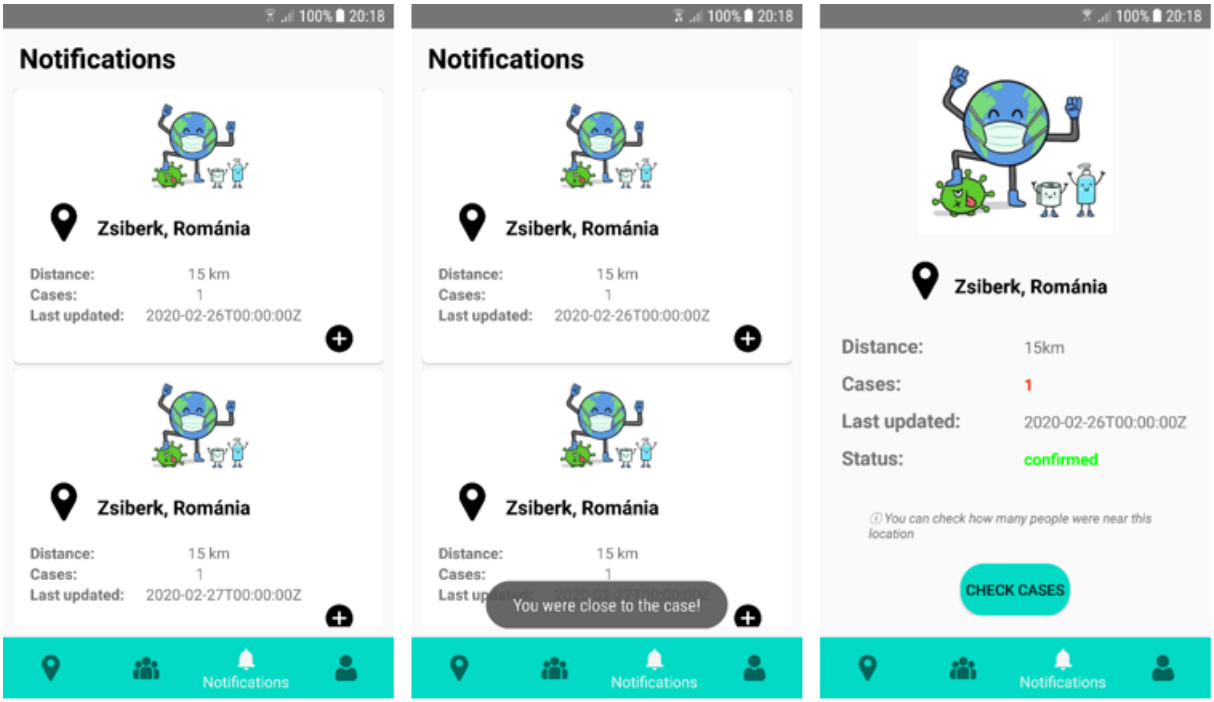
4.8.ábra: Az Elhelyezkedés (Térkép) oldal ábrázolása

A *4.8.ábra* a bejelentkezést követő első oldalt, az elhelyezkedést megjelenítő oldalt, pontosabban egy térképet szemléltet a felhasználóval. Ennek az oldalnak a célja egy jobb felhasználói élményt nyújtani az alkalmazás felhasználóinak, továbbá a GPS technológia segítségével a felhasználó meghatározhatja saját tartózkodási helyét. Amennyiben a felhasználó nem szeretné igénybe venni a GPS technológia által nyújtotta szolgáltatást, az alkalmazás célja, a kontaktkutatás megvalósítása továbbra is elérhető funkció marad, ugyanis annak érdekében a Bluetooth technológia alkalmazását használja a rendszer. Tovább navigálva a menüpontok között, a következő oldal a kezdetleges elképzelések alapján a rendszerben regisztrált felhasználók listáját szemlélteti. Ennek a megvalósításnak a fő célja az volt, hogy ellenőrizni tudjuk, hogy az alkalmazás helyesen jeleníti-e meg az adatbázisban tárolt adatokat. A következő ábra a felhasználókat megjelenítő oldalt ábrázolja.



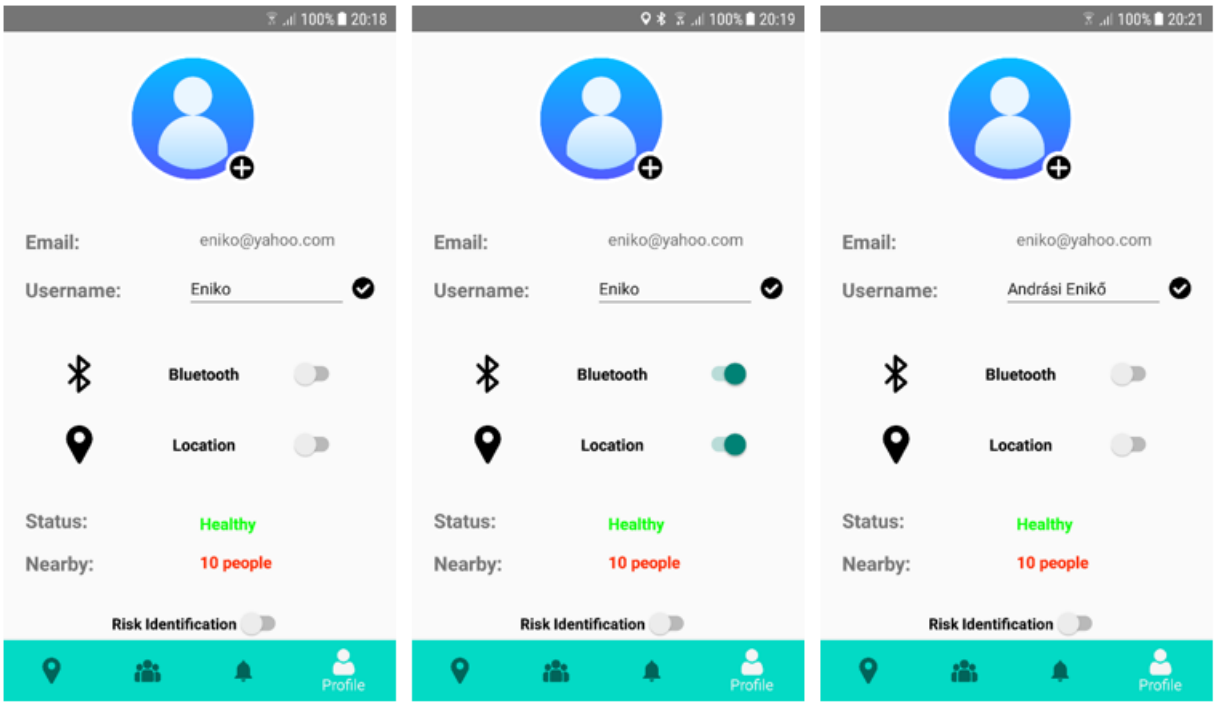
4.9.ábra: A felhasználók listáját megjelenítő oldal ábrázolása

Amint a *4.9.ábrán* látható a felhasználók listáját az alkalmazás helyesen megjelenítette, a különböző információk mindössze a tesztelés szemszögéből lettek generálva. A következő oldalt az értesítések oldala képviseli, amelynek a felépítése a *4.10.ábrán* látható.



4.10.ábra: Az eseteket tartalmazó oldal ábrázolása

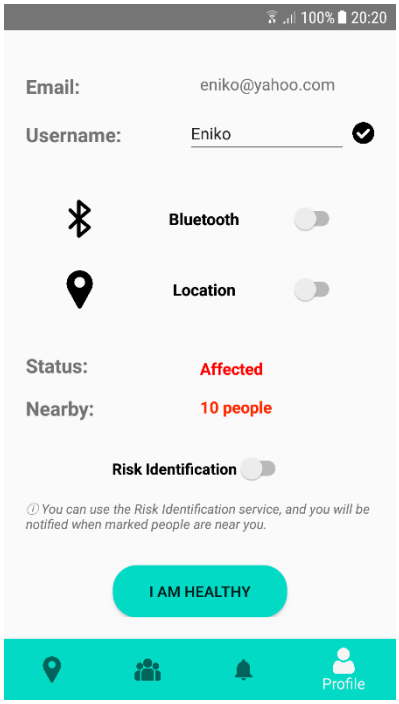
Esetünkben az értesítéseket az általam felhasznált API-ról származó esetek jelentik. Az előző ábra szemlélteti az ezen oldalon megjelenített eseteket, és a hozzájuk tartozó néhány információt. A felhasználónak lehetősége van ezen az oldalon figyelemmel követni, hogy hol és mikor észleltek új eseteket, továbbá egy esetet kiválasztva arról részletesebb információkat is olvashat. Megfigyelhető, hogy minden értesítés esetében egy gomb helyezkedik el a jobb alsó sarokban. Ennek a gombnak a funkciója az, hogy amennyiben a felhasználó az esethez közel volt, esetleg ott tartózkodott, ezt megnyomva jelezheti a rendszernek, amit az adatbázisban eltárol a rendszer. Ezt a folyamatot a jobb oldali kép szemlélteti. Tovább navigálva az alkalmazásban az utolsó oldalra érkezik a felhasználó, amely a profil oldalt jeleníti meg. Ez az alábbiakban kerül bemutatásra.



4.11.ábra: A felhasználó profilját megjelenítő oldal ábrázolása

A *4.11.ábra* elemzése során észrevehető, hogy ez az az oldal, amely az alkalmazás céljának megfelelően a legfontosabb funkciókat tartalmazza. Itt a kapcsoló gombok használatával aktiválható az alkalmazás működése érdekében elengedhetetlen funkció, a Bluetooth technológia, illetve igény szerint a GPS, vagyis a mobil eszköz helymeghatározó rendszere. Ez az oldal megjeleníti az adott felhasználó adatait, melyek közül a nevét bármikor megváltoztathatja. Az oldal alján két fontos információt jelenít meg az alkalmazás, mégpedig a felhasználó pillanatnyilag, a rendszerben tárolt egészségügyi állapotát, illetve a közelben tartózkodó más

felhasználók számát. Továbbá ugyancsak ezen az oldalon helyezkedik el, az egyik legfontosabb funkciót betöltő gomb is, melyet használva a felhasználó tudathatja a rendszerrel, hogy megváltozott egészségügyi állapota, így a rendszer majd figyelmeztető értesítéseket küldhet az érintett felhasználók számára. Az egészségügyi státusz megváltoztatásának a folyamatát a *4.12.ábra* szemlélteti.



4.12.ábra: A felhasználó profilját megjelenítő oldal ábrázolása

Látható, hogy amennyiben a felhasználó a gombra kattint, annak megváltozik a rajta elhelyezkedő szövege, továbbá az állapotát jelölő szöveg is átvált. Fontos megjegyezni azt is, hogy folyamatos internet használat mellett, ez az információ valós időben bekerül, illetve megváltozik az adatbázisban is.

Ebben a fejezetben az első tervezési fázisnak a szakaszairól, és részletes bemutatásáról volt szó, a következőkben pedig a második tervezési fázisnak a szakaszi, illetve részletes bemutatására kerül sor.

4.2. Második tervezési fázis

A második tervezési fázis, egyfajta újra tervezésnek, illetve tovább tervezésnek a fázisaként említhető. Az újra tervezés alatt, bizonyos funkciók megvalósításának újra gondolását, átformálását, a tovább tervezés fogalma alatt pedig újabb funkciók megvalósítását, az alkalmazás

felhasználóbarát környezetének a továbbfejlesztését értjük. Abból a tényből kiindulva, hogy az első tervezési fázis, megalapozta az alkalmazás megvalósítását, úgy az architektúrájának felépítése szempontjából, mint a felhasználói felület megtervezése és a rendszer adatbázissal való kommunikációjának megvalósítása szempontjából, a második tervezési fázisnak a szakasza az elsőnek az eredményéből bontakozik tovább. Annak érdekében, hogy a fejlesztett alkalmazás, illetve szoftver rendszer a célkitűzések nagyobb részének eleget tegyen, elengedhetetlen további tervezési fázisok folyamatba helyezése, ugyanis így érhető el, egy a céloknak megfelelő alkalmazás, rendszer megvalósítása. A második tervezési fázisban az alkalmazás különböző oldalai által megvalósított funkciók újragondolása, javítása volt az elsődleges cél. Kezdetben az első menüpont, a térkép megjelenítését tartalmazta. Ennek tovább fejlesztésének a terve az volt, hogy amennyiben a felhasználó aktiválja a GPS funkciót, abban az esetben a térképen megjelenjen a felhasználó pillanatnyi tartózkodási helye, továbbá az alkalmazást használó más felhasználók tartózkodási helye is, ennek célja egy összképet mutatni a felhasználók számára az alkalmazás felhasználóinak számáról, illetve helyszínéről. Fontos szempontnak tartottuk azt, hogy a különböző felhasználók térképen való megjelenítése, mindössze csak arról adjon tudomást a felhasználó számára, hogy körülbelül hány felhasználó van a környezetében, és kiemelten fontos az, hogy róluk ne jelenítsen meg a rendszer semmilyen információt. Az első tervezési fázis eredményeként a második oldalon a regisztrált felhasználóknak a listája volt látható bármely felhasználó számára. Újra gondolva az alkalmazás lényegét, ennek az oldalnak a tartalma a második tervezési folyamatban újra terveződött. Az új tervek alapján ezen az oldalon a felhasználók listája helyett, az úgynevezett érintkezések listája válik láthatóvá a felhasználó számára. Fontos szempont az is, hogy nem az összes igazolt kontaktus jelenik meg ezen az oldalon, hanem kizárólag csak azok, akikkel az adott felhasználó volt kapcsolatban. A legfontosabb szempont a kapcsolatok megjelenítése során az, hogy azok ne közöljenek információt a más felhasználók személyéről, így mindössze csak azok Bluetooth azonosítóját, illetve a találkozásnak a dátumát jelenítik meg. Ennek az oldalnak a célja az, hogy a felhasználó egy összképet kapjon arról, hogy körülbelül hány személlyel volt kontaktusban, elkerülve azt az információt, hogy kikkel. A harmadik oldalnak a célja az első fázisban megjelenített értesítések átalakítása, ugyanis azok egy publikus API által szolgáltatott információk alapján az igazolt esetekről szolgáltattak információkat, úgynevezett értesítéseket. Ebben a tervezési fázisban az volt a cél, hogy ezen az oldalon a szerver által küldött figyelmeztető üzenetek jelenjenek meg, amelyek abban az esetben szükséges, hogy kiküldésre kerüljenek, amikor a felhasználó, egy számára veszélyt jelentő

személlyel volt kapcsolatban. Végül pedig az utolsó oldalnak, a profilt megjelenítő képernyőnek a funkcionalitásai megmaradtak, csak a megvalósításai lettek újra tervezve.

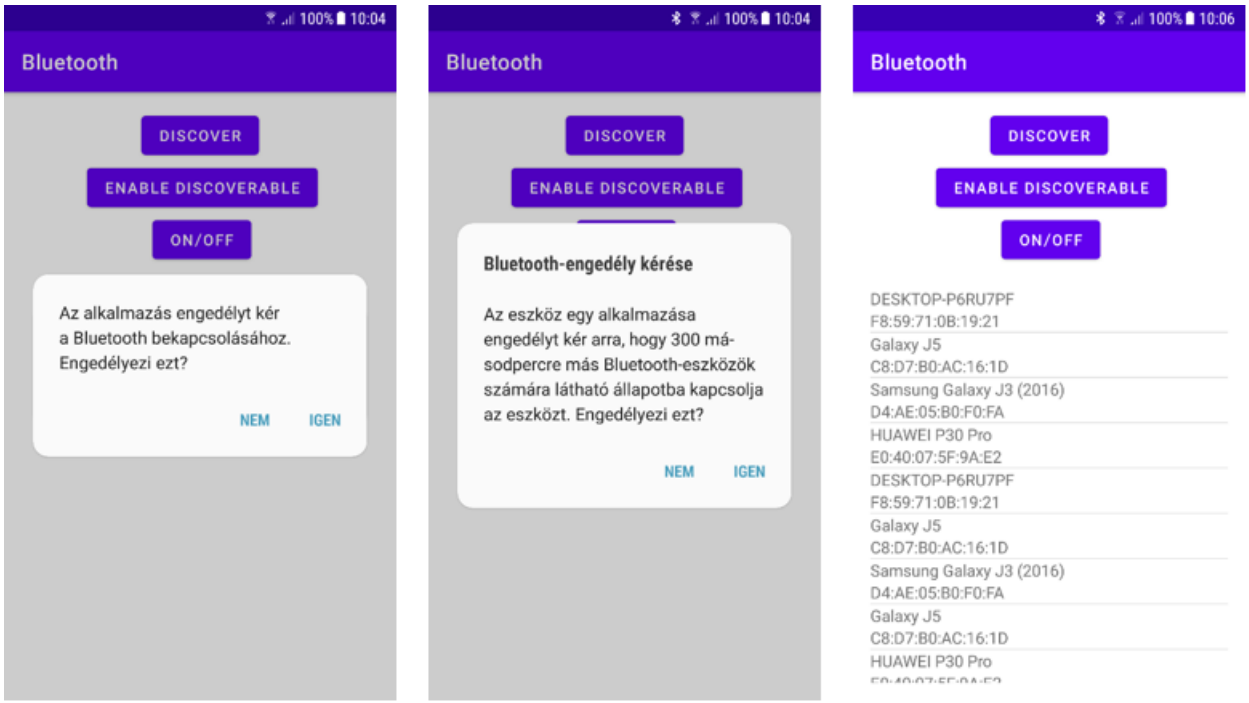
Az alkalmazás különböző oldalainak az újra tervezéséből, vagyis a Frontend újra tervezéséből következett a Backend rész, avagy a háttérben zajló folyamatokat, működést lehetővé tevő résznek az újra tervezése. Kezdetben a tervezés folyamatát a rendszer architektúrájának az újra tervezésével kezdtem, melyet az előző fejezetben, pontosabban a harmadik fejezetben, a rendszer architektúrájának elemzését szolgáló alfejezetben részletesen tárgyaltam. Ennek alapján nyilvánvalóvá vált számomra, hogy annak érdekében, hogy a rendszer, egy nagyobb adatforgalomnak a kiszolgálását biztosítani tudja, szükségünk lesz egy szerverre, aminek a feladata az alkalmazás felhasználói felülete és az adatbázis közötti kapcsolat megteremtése. Tekintettel arra, hogy az alkalmazás megvalósításának az elvárt eredménye egy nagyobb számú felhasználó csoportot megcélozni, így várhatóan sok felhasználó közreműködésére számíthatunk. Ennek következtében a rendszer által tárolt adatok mennyiségének a mértéke is nagyszámú növekedéshez vezethet, így a tovább fejlesztés fázisában felmerült egy másik adatbázisnak a használata is, amelyben gond nélkül tárolhat az alkalmazás több adatot is. A következő alfejezetben a második tervezési fázisnak a megvalósításai kerülnek részletesen bemutatásra.

4.2.1. A második tervezési fázis megvalósításának a részletes bemutatása

Az ebben a fázisban készített tervek alapján, a különböző funkcionalitások megvalósítását külön csoportosítva terveztem megvalósítani. Ennek következtében négy nagyobb feladatra osztottam fel a második tervezési fázist. Ezzel azt szerettem volna elérni, hogy külön alkalmazásokban sikerüljön kezdetben megvalósítani a különböző funkciók működését, amennyiben ez a folyamat pedig sikeresnek minősül, ezeket az első fázisban tervezett alap alkalmazásba terveztem integrálni.

Az első nagyobb feladat keretén belül az alkalmazás egyik legfontosabb tulajdonságát, illetve feladatát terveztük megvalósítani, mégpedig a Bluetooth technológia által megvalósított kontaktusok felderítését. Kezdetben egy különálló Android alkalmazásban sikerült megvalósítani a Bluetooth technológia felhasználását. Esetünkben pontosan azt kellett megvalósítani, amit az okostelefonok beállításaiban el lehet végezni a Bluetooth beállításoknál. Szükségünk volt annak a megvalósítására tehát, hogy az alkalmazásból ki-be tudjuk kapcsolni a mobil eszközünkbe integrált Bluetooth technológiát, továbbá pedig célunk volt elindítani a Bluetooth hatókörében elhelyezkedő más eszközök felderítési folyamatát. Amennyiben ezek a célok sikeresen

megvalósultak, az eszköz által felderített más eszközöket listáztam az alkalmazásban, hogy ezáltal is tesztelni lehessen a funkcionalitás helyes működését. Figyelve a személyi jogok védelmére, a környező eszközöknek nem a Bluetooth nevét kértem le az alkalmazásban, hanem azok azonosítóját, amelynek felépítése megfelel egy biztonságos anonim azonosítónak, illetve belőle nem lehet következtetni a felhasználó személyére. A Bluetooth technológia felhasználásának a megvalósítását egy Android alkalmazásban, az alábbi ábrán szemléltetem.



4.13.ábra: A Bluetooth technológia felhasználása egy Android alkalmazásban

A *4.13.ábrán* három képernyőkép mutatja be azt a folyamatot, melyet sikerült megvalósítani a Bluetooth technológia segítségével egy különálló Android alkalmazásban. A példában három gomb felhasználásával ki, illetve be kapcsolható az eszköz Bluetooth-ja, láthatóvá tehető más eszközök számára, egy 300 másodperces időre, illetve a harmadik képen az eszköz hatókörében levő, érzékelt más eszközök listázása látható. Ebben a tesztalkalmazásban még megjelenítettem a felderített eszközök neveit is, annak érdekében, hogy könnyedén tesztelhető legyen az alkalmazás. A későbbiekben, kizárólag csak az eszközök Bluetooth azonosítóját jelenítettem meg, a személyi jogok védelmének érdekében. Az egyes funkcióknak a megvalósításait különböző módszerek által sikerült megvalósítani, az alábbiakban bemutatásra kerül a hatókörben levő eszközök felderítését és listázását megvalósító függvény, mely az alkalmazás egyik legfontosabb funkciójának az alapjául szolgál.

```
fun btnDiscover(view: View?) {
    Log.d(TAG, msg: "btnDiscover: Looking for unpaired devices.")
    if (mBluetoothAdapter!!.isDiscovering) {
        mBluetoothAdapter!!.cancelDiscovery()
        Log.d(TAG, msg: "btnDiscover: Canceling discovery.")

        //check BT permissions in manifest
        checkBTPermissions()
        mBluetoothAdapter!!.startDiscovery()
        val discoverDevicesIntent = IntentFilter(BluetoothDevice.ACTION_FOUND)
        requireContext().registerReceiver(mBroadcastReceiver3, discoverDevicesIntent)
    }
    if (!mBluetoothAdapter!!.isDiscovering) {

        //check BT permissions in manifest
        checkBTPermissions()
        mBluetoothAdapter!!.startDiscovery()
        val discoverDevicesIntent = IntentFilter(BluetoothDevice.ACTION_FOUND)
        requireContext().registerReceiver(mBroadcastReceiver3, discoverDevicesIntent)
    }
}
```

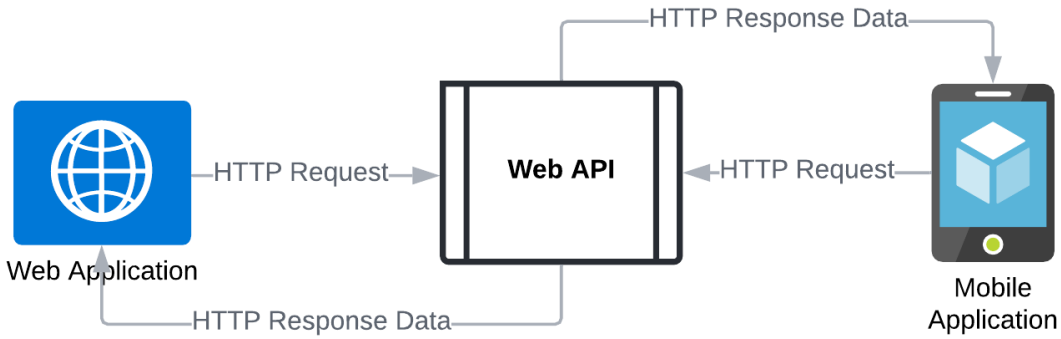
4.14.ábra: Az eszköz hatókörében elhelyezkedő eszközök felderítésére szolgáló függvény

A *4.14.ábra* az eszközök felderítését szolgáló műveleteket szemlélteti, egy függvényt, mely az eszközök felderítését teszi lehetővé a Bluetooth technológia segítségével [36]. Mint előző megoldásokban, így itt is feltételes utasítások segítségével döntjük el, hogy az adott kódrészletet szükséges-e végrehajtani. Ezen utasítások a működés közbeni lehetséges hibák elkerülését szorgalmazzák. A kódrészlet elején az első utasításban az kerül ellenőrzésre, hogy az eszközön aktiválva van-e már az eszközök felderítése funkció, tehát, hogy keresi-e már az eszközöket a rendszerünk. Amennyiben az eszközön már folyamatban van a felderítési folyamat, ez megszakításra kerül, majd újra elkezdi a rendszer az eszközök felderítését. Fontos kiemelni ebben a részben azt, hogy amennyiben a felhasználó az úgynevezett Lollipop rendszernél újabb Android rendszert használ eszközén, a helyes működés érdekében itt szükséges leellenőrizni, hogy engedélyezve van-e a Bluetooth használat. Ennek leellenőrzéséért egy külön függvény felel, amelyet itt meghívunk, meghívása során pedig végre hajtja az ellenőrzés folyamatát, az úgynevezett Manifest fájlban. Amennyiben sikeresen megtalálja a Bluetooth-al kapcsolatos szükséges engedélyeket, elkezdi az eszközök felderítésének a folyamatát. Az ábrán szereplő következő feltételes utasítás már azt ellenőrzi, hogy az eszköz nem kezdte-e még el a felderítést. Amennyiben még nem kezdődött el ez a folyamat, akkor a következő lépésben a rendszer elkezdi ennek a folyamatnak a megvalósítását, szintén az engedélyek ellenőrzését elvégezve.

A második nagyobb feladat keretén belül, a szerverhez való hozzáférés megvalósítására, egy úgynevezett Web API megvalósítása volt a cél, ami a kliensoldali alkalmazás és az adatbázis között áll és melynek feladata megvalósítani a kapcsolatot az adatbázis és a kliensoldali

alkalmazás között. Abból a tényből kiindulva, hogy a kliensoldali alkalmazás nincs közvetlen kapcsolatban az adatbázissal, szükséges volt egy személyre szabott API megvalósítására, ami képes megvalósítani az alkalmazás kommunikációját az adatbázissal. Amint azt az API általános meghatározása is leírja, ez egy olyan alkalmazásprogramozási felület, angolul Application Programming Interface, amely különböző protokollok és definíciók összesége az egyes szoftverek és alkalmazások felépítéséhez [27] . Ezen belül a Web API, egy olyan API-ként értelmezhető, ami az interneten keresztül a HTTP protokoll segítségével érhető el. Mint ismert egy Web API-t különféle technológiák segítségével lehet megvalósítani, jellegzetesebb ezek közül a .NET, valamint a Java technológiákban felépített alkalmazásprogramozási felületek [27].

Az általam megvalósított Web API létrehozása érdekében a .NET technológiát használtam fel, ennek köszönhetően sikerült felépíteni egy ASP.NET Web API-t. Tudva azt, hogy az ASP.NET Web API egy bővíthető keretrendszer HTTP alapú szolgáltatások felépítéséhez, továbbá egy ideális platform RESTful szolgáltatások megvalósításához, illetve különböző válaszformátumokat is támogat, mint például a JSON, XML és a BSON-t is, ezért ideálisnak bizonyult az általam megtervezett alkalmazás megvalósításához [27]. A megvalósítás szempontjából hasznosnak minősült egy API létrehozása, ugyanis előnyként tekinthető, hogy függetlenül attól, hogy ki szeretné használni, elérni az adatokat, az API képes eldönteni, hogy hogyan reagáljon bizonyos kérésekre, mint például egy feltöltésre, vagy éppen egy lekérdezésre. Az alábbiakban szemléltetem az általam megtervezett API architektúráját, mely ábrán van összefoglalva ennek működési elve.

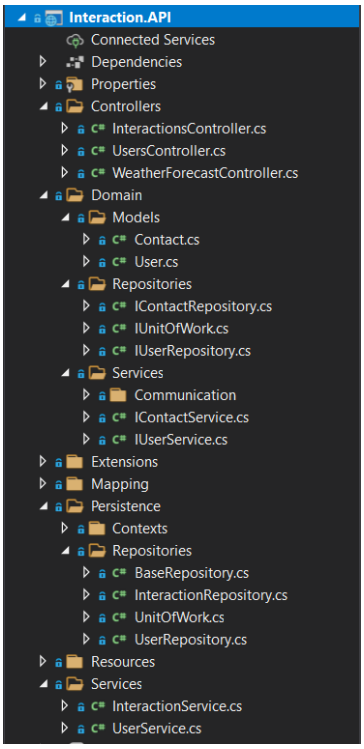


4.15.ábra: Az ASP.NET Web API architektúrája

A *4.15.ábra* szemlélteti a rendszerbe integrált szerverhez való hozzáférést és kommunikációt megvalósító Web API-nak az architektúráját. Látható, hogy középen helyezkedik el a Web API, ami a HTTP protokollnak megfelelő kéréseket szolgál ki. A kérések rendszerint két irányból

érkeznek, az egyik maga a Web alkalmazás, a másik a mobil alkalmazás, amelyek úgynevezett HTTP kéréseket küldenek az API fele. Az API-nak a feladata pedig fogadni ezen kéréseket, és válaszként úgynevezett HTTP válasz adatokat szolgáltatni a kéréseket megfogalmazó felek irányába. Az előzőekben bemutatott architektúra alapján sikerült megvalósítani egy strukturált, könnyen karbantartható RESTful API-t az ASP.NET Core segítségével. Fontos, hogy a hatékony megvalósítás céljából megfelelő eszközöket válasszunk a RESTful szolgáltatások megvalósításához, ugyanis fontos odafigyelni a karbantarthatóságra és a skálázhatóságra egyaránt. Ennek érdekében használtam fel az ASP.NET Core által nyújtott lehetőségeket, ami egy hatékony, könnyen kezelhető API megvalósítását tette lehetővé. Munkám során egy olyan web API-t szerettem volna megvalósítani, ami a következő kéréseket tudja kiszolgálni, úgy megalkotva ezt, hogy a GET, POST, PUT, DELETE metódusok mindegyikére választ tudjon adni. Inspirációként szolgált ennek megvalósítására egy olyan útmutató, mely egy blogokkal kapcsolatos web API elkészítését tárgyalta, amelyben arra is sikerült útmutatást kapni, hogy hogyan kell egy ilyen API-t összekötni a kliensoldali alkalmazással [28].

A továbbiakban egy ábrán fogom szemléltetni az általam megvalósított API-nak a fájlstruktúráját, bemutatva ezáltal a felépítéséhez szükséges osztályokat és interfészeket.



4.16.ábra: A rendszer API részének a fájl struktúrája

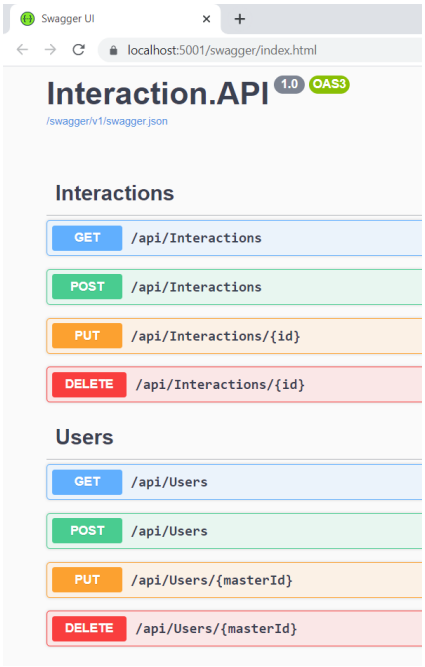
A *4.16.ábra* szemlélteti az API átlátható struktúráját, és az azt felépítő fontosabb osztályokat, interfészeket. Az interfészeket az I-betűvel kezdődő fájlok jelképezik, továbbá bizonyos sablonok megadásával utaljuk a rendszert az elvárt működésre. Továbbá kiemelném a munkatervezési eljárásként használt Repositories könyvtárat, melynek felhasználásának a célja az volt, hogy átláthatóságot biztosítson, illetve célja továbbá elválasztani a servicet a konkrét kérésektől. Egy további munkatervezési folyamatként lett felhasználva az Entity Framework Core, melynek célja egységbe zárni az objektumokat és a műveleteket, amelyeket ezekkel végzünk úgy, hogy egy objektumorientált látványt biztosít ezáltal a megvalósításnak. A következőkben egy ábra fogja szemléltetni az általam használt egyik interfészt és az általa definiált metódusokat.

```
namespace Interaction.API.Domain.Services
{
    4 references
    public interface IContactService
    {
        2 references
        Task<IEnumerable<Contact>> ListAsync();
        2 references
        Task<InteractionResponse> SaveAsync(Contact interaction);
        2 references
        Task<InteractionResponse> UpdateAsync(int id, Contact interaction);
        2 references
        Task<InteractionResponse> DeleteAsync(int id);
    }
}
```

4.17.ábra: A kontaktusok szolgáltatásait egységbezáró interfész

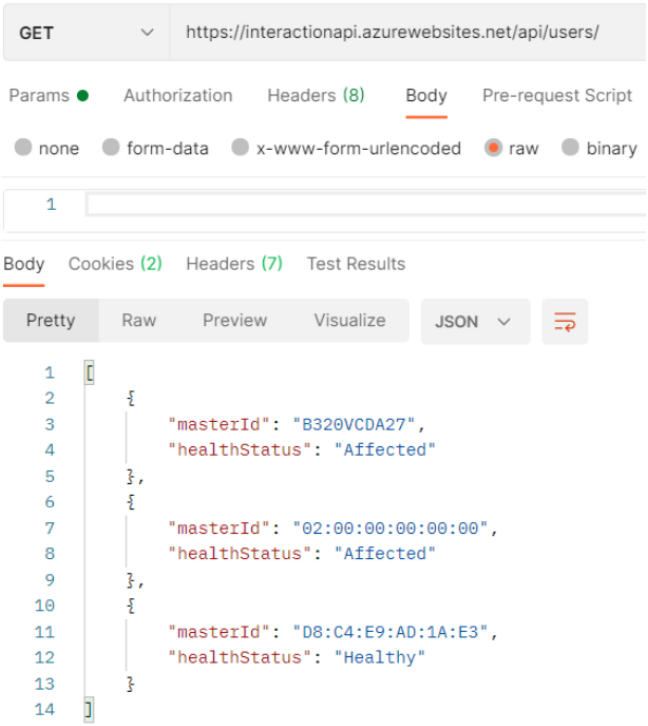
A *4.17.ábrán* egyike látható az általam implementált interfészeknek, az API megvalósítása során. Ezen ábra azt hivatott szemléltetni, hogy milyen sablonszerű feladatokat kellett ellátnia az API-nak. Ez alapján az API-nak lehetőséget kellet biztosítania az interakciók listázására, új interakciók elmentésére, esetleges változtatására, végül pedig a törlésre, mert egy adott idő után szükséges az adott érintkezéseket törölni, az adatok frissítése szempontjából.

Az API dokumentálására a jól ismert Swagger-t használtam fel, ami nyílt forráskódú és professzionális eszközkészletével elősegítette az API fejlesztését [29].



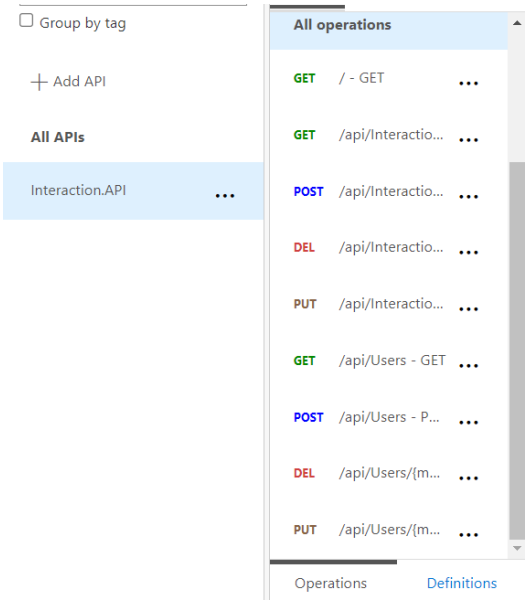
4.18.ábra: A Swagger felhasználói felület ábrázolása

Az API tesztelésével kapcsolatban megemlíthető, hogy a böngésző mellett segítségemre szolgált a Postman is, ami egy együttműködési platformként szolgált az API fejlesztéshez [30].



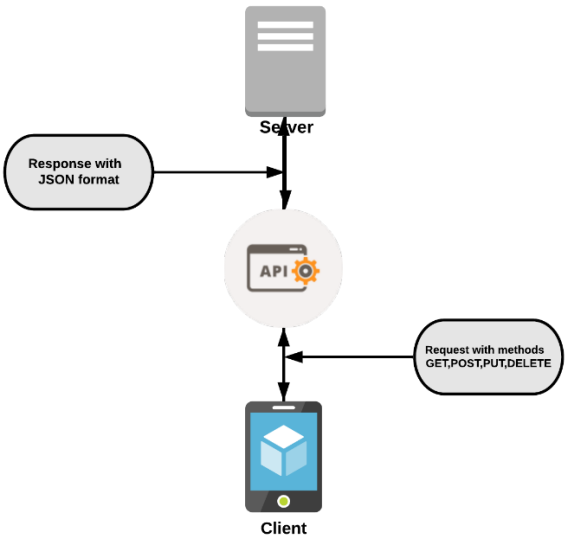
4.19.ábra: A Postman felületén követhető eredmények

Annak érdekében, hogy az API bárhonnan elérhető legyen, ne csak az adott hálózatról, felhasználtam az Azure ingyenes szolgáltatásait, annak érdekében, hogy bármely hálózatról és más mobil eszközökről is akadálymentesen csatlakozni lehessen [31].



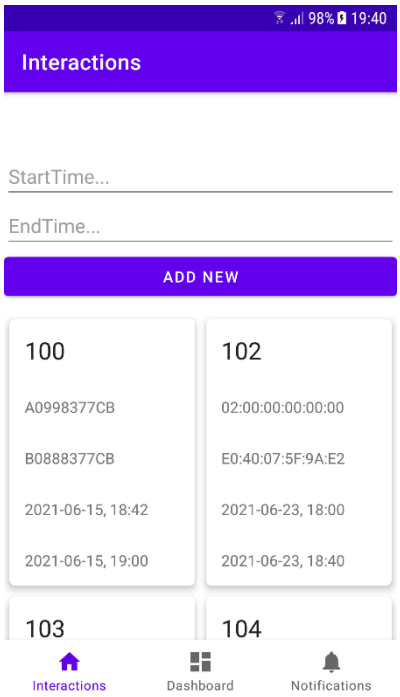
4.20.ábra: Az Azure felületén megjelenített kérések

Az általam felosztott feladatok harmadik feladatának keretén belül ugyancsak egy különálló alkalmazásban valósítottam meg egy Android alkalmazás és a webszerver között létrejövő kommunikációt, az általam felépített API segítségével. A tesztalkalmazás célja az volt, hogy sikerüljön megvalósítani a kommunikációt az alkalmazás és a szerver között, mindezt úgy, hogy lehetőség legyen az alkalmazásból is elérni az API által támogatott különböző HTTP protokoll alapú metódusokat. Esetünkben a GET, POST, DELETE és UPDATE metódusok működését volt szükséges megvalósítani, annak érdekében, hogy manuálisan lehetőségünk legyen egy teljes információval ellátott kapcsolatot, úgynevezett érintkezést hozzáadni az adatbázishoz, az Andorid-os alkalmazásból, lekérni, törölni, illetve frissíteni azt. Ennek a feladatnak a célja az volt, hogy amennyiben sikerül megvalósítani a manuálisan létrejövő kommunikációt a két egység között, célunk elérése érdekében már csak automatizálni kell ezen folyamatot. Az alábbiakban bemutatásra kerül a rendszerben felhasznált Retrofit architektúrája, mely egyben a rendszer működésének az alapját képezi.



4.21.ábra: A Retrofit architektúrája

A *4.21.ábra* a rendszer működési elvét szemlélteti, mely alapján a kliensoldali alkalmazás a HTTP protokoll metódusai alapján kéréseket fogalmaz meg az API irányába, mely továbbítja ezen kéréseket a rendszert működtető szerver irányába, majd válaszként JSON formátumú eredményt kap vissza. Az előzőekben említett folyamatot megvalósító alkalmazást, az alábbiakban szemléltetem.



4.22.ábra: Az érintkezések manuális, alkalmazásból való elmentését megvalósító alkalmazás

A *4.22.ábrán* az érintkezések kézi elmentése az adatbázisba és azok megjelenítése az alkalmazásban, látható. Az érintkezések oldalon a szükséges adatokat váró szövegmezők találhatóak, illetve egy gomb, mellyel hozzá lehet adni az adott kapcsolatot a rendszerhez. A kézi hozzáadás érdekében minden mezőt szükséges kitölteni, maga a kontaktus azonosítóját pedig a rendszer automatikusan sorrend alapján rendeli hozzá az érintkezésekhez. Mint látható a 100-as azonosítójú kontaktus után a 102-es következik, ennek célja szemléltetni, hogy a 101-es interakció törlésére került sor, így tesztelhető volt a lekérés és törlés metódusok helyes működése. Az említett folyamatok megvalósításához a következőkben bemutatott megoldások vezettek. Kezdetben a rendszernek az úgynevezett REST Client része kerül bemutatásra, amelyet a kliens oldalon valósítottam meg, az Android alkalmazásban. A következő kódrészlet egy interfész megvalósítását ábrázolja, mely definiálja a viselkedésmódot, esetünkben meghatározza a felhasznált HTTP műveleteket.

```
interface IContactService {

    @GET( value: "api/interactions")
    fun getInteractions(): Call<List<Interaction?>?>?

    @POST( value: "api/interactions")
    fun addNewInteraction(@Body interactionResponse: InteractionResponse?): Call<InteractionResponse?>?
}
```

4.23.ábra: A REST Client által felhasznált interfész

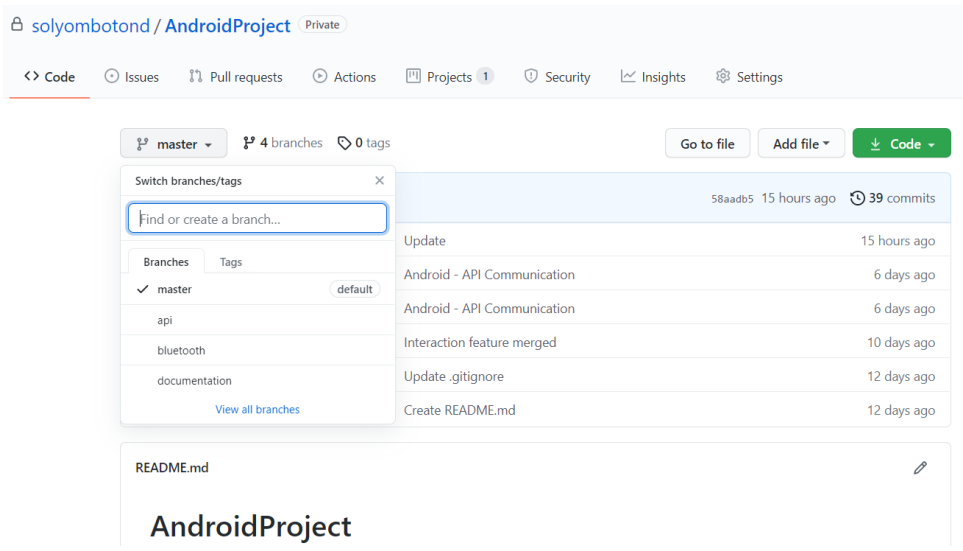
Az ábra egy interfészt és az általa definiált viselkedésmódokat tartalmazza, pontosabban egy GET, lekérdező HTTP protokoll alapú metódust definiál és egy POST, vagyis hozzáadást megvalósító metódust. Az interfészen belül implementált bármely metódus egy lehetséges API hívást reprezentál. Szükséges a metódusok felett, az úgynevezett HTTP annotációt alkalmazni, annak érdekében, hogy meghatározzuk a kérés típusát, illetve az elérési URL címet. A metódusnál meghatározott visszatérítési érték egy úgynevezett hívásobjektumba burkolja a választ a várt eredmény típusával együtt.

A továbbiakban egy létrehoztunk egy úgynevezett RetrofitService osztályt, amely szorosan kapcsolatban áll az előzőekben bemutatott interfésszel, ugyanis implementálja azt. Mivel az ezen osztály implementálja az említett interfészt, akkor példányai az interfészben meghatározott viselkedéssel fognak rendelkezni. Ezen osztály segítségével gyakorlatilag egy olyan példányt sikerült fel építeni, amelynek a segítségével fogjuk a későbbiekben a lekérdezéseket megvalósítani. Megjegyzendő az, hogy a megszokott osztály definiálása helyett ezúttal objektumot hozunk létre. Fontos megjegyezni azt, hogy azért használunk objektumot, egy általános osztály

helyett, mert az általános osztálynak lehet több példánya is, az objektumnak viszont ebben az esetben egyetlen egy példánya lesz, ugyanis ez elegendő ahhoz, hogy az API-tól lekérdezzünk adatokat. Ezáltal implementálva van egy a szoftvertervezés folyamatában gyakran használt tervezési minta, angolul design pattern, amelyet Singleton-nak nevezünk. Az úgynevezett Egyke tervezési mintának köszönhetően megvalósítható tehát, hogy az osztályunknak csak egy példánya legyen, miközben globális hozzáférést biztosít ehhez a példányhoz.

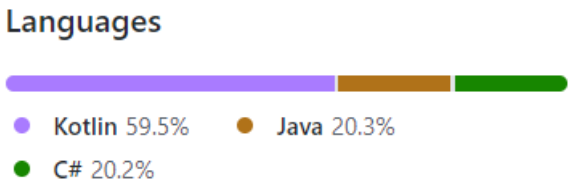
Végül pedig az előzőekben részletesen elemzett feladatok megvalósítása után, a második tervezési fázis negyedik és egyben az utolsó feladata a külön elkészített teszt alkalmazások összerakása következett az első tervezési fázisban elkészített alkalmazásba beépítve. Ezen folyamat során sikerült elindítani az alkalmazás működését, így az ebben a fázisban összeállított alkalmazás, már mind implementálja az előzőekben bemutatott funkciókat és megvalósításokat egyaránt.

A részletes tervezés fejezet összegzésében olyan a tervezést elősegítő rendszereket szeretnék bemutatni, amelyeknek a használata elősegítették a projektem megvalósításának, rendszerezésének és biztonságának állandó megőrzésének a lehetőségét. A projekt megvalósítása során az egyetemi feladatok és más feladatok során is gyakran használt, GitHub verziókövető rendszert használtam, mely elősegítette a különböző tervezési fázisok során való munkám rendszerezését és biztonsági tárolását [32]. Saját, ezen projekt számára létrehozott, privát úgynevezett repository-ban, adattáramban 4 különböző branchet, magyarul elágazást használtam, melyeken a külön tervezési fázisokban felváltva dolgoztam. Az alábbi ábra a böngészőben megnyitott adattáramat szemlélteti a különböző elágazásokkal.



4.24.ábra: A projekt verziókövetésére használt rendszer ábrázolása

Mint ahogy a *4.24.ábrán* is látható, a 4 különböző branchek között váltani lehet a rendszerben, így lehetőség adódik a különböző fázisokon dolgozni. Esetemben a master elnevezésű ágon az összetett alkalmazás található, az api ágon az API megvalósítása, a bluetooth ágon a Bluetooth tehcnológia Andorid-ban való megvalósítása, illetve a documentation ágon a projekt dokumentálása található. Amint azt a következő ábra is szemlélteti a teljes rendszer megvalósítása során több programozási nyelvben is dolgoztam, melyek közül a felhasználói felület tervezését képviselő Kotlin programozási nyelv említhető nagyobb arányban. Továbbá a Java nyelv szintén az Android programozásban segített munkám során, a C# pedig az API megvalósítását szorgalmazta.



4.25.ábra: A projektem során felhasznált programozási nyelvek arányának a megoszlása

A GitHub verziókövető rendszer és a számítógép közötti kapcsolat megteremtésének érdekében a GitHub Desktop, asztali alkalmazást használtam, mely elősegítette a projekt állandó szinkronizálását [33].

5. Üzembe helyezés és kísérleti eredmények

A részletes tervezési lépések megalkotásának, illetve azok kivitelezésének bemutatása után a különböző fázisok során összetett alkalmazás tesztelése és üzembe helyezése következett. A végleges alkalmazás verziójának tesztelése során sajnos nem tudtunk nagyszámú közösségeket bevonni, azonban kisebb, otthoni környezetben sikerült tesztelni az alkalmazás jelenlegi működését. Ebben a fejezetben a saját eszközeimnek a felhasználásával megvalósított kísérleti eredményeket szemléltetem, illetve a projekt megalkotása során felmerült problémákat és azok megoldásait.

5.1. Üzembe helyezési lépések

A rendszert egy okostelefon felhasználásával sikerült tesztelnem, amihez más, a Bluetooth technológiát implementáló mobil eszközöket is felhasználtam. Az alkalmazást egy telefonon futtattam, illetve a Bluetooth-ot több eszközön is aktiváltam, annak érdekében, hogy valós teszt eredményekhez juthassunk, és hogy ellenőrizni lehessen, hogy valóban reális adatok kerülnek be a rendszerbe. Mivel a kísérleti folyamatot megvalósító eszközök mind a tulajdonomban voltak, ezért az eredmények helyességének ellenőrzése szempontjából azoknak a beállítási menüpontjában ellenőrizni tudtam, hogy valóban az adott eszköz Bluetooth azonosítója került be a rendszerbe. Az eredményeket az általam tervezett, Azure rendszerre feltöltött API-nak segítségével ellenőrizhettem a Microsoft által kezelt adatközpontokon keresztül. Az eredmények jelenleg bárki számára megtekinthetők, akiknek birtokában van az odavezető URL cím.

5.2. Felmerült problémák és megoldásaik

Ebben az alfejezetben a rendszer tervezése és megvalósítása során felmerült problémák és azok esetleges megoldásai kerülnek bemutatásra. A projekt kivitelezése során számos olyan akadályokba ütköztünk, amelyek újra tervezést, illetve más módszerek alkalmazását igényelték a továbbhaladás szempontjából.

Kezdetben egy olyan kisebb probléma merült fel, hogy az első tervezési fázisban felhasznált API-ra egy bizonyos idő után már nem kerültek fel újabb adatok, így már nem tudott frissen regisztrált adatokat közvetíteni az alkalmazás felé. Bár ez a megoldás tökéletesnek minősült, az API és az Android alkalmazás kommunikációjának megvalósítására, azonban célunk elérése érdekében már nem bizonyult elegendőnek. Ezen helyzet javítása érdekében szükségszerű volt egy személyre szabott API létrehozása, amely az eredeti tervnek megfelelően valós, reális adatokkal látja el a rendszert. Ennek megvalósítását segítette elő a .NET technológiák felhasználása, melynek felhasználásával egy összetett, saját API-t készíthettünk.

Egy másik probléma, ami felmerült a rendszer megvalósítása során, az az Android alkalmazásból való kapcsolódás az API-ra volt, ugyanis ezt lokálisan sajnos nem sikerült megoldani a fizikai eszközről, csak az emulátorról. Bár az emulátoron keresztül sikerült elérni a kapcsolódást, a fizikai eszközök esetében kapcsolódási hibába ütközött a rendszer. Ennek a hibának a javítása érdekében, és annak a célnak a megvalósítása érdekében, hogy globálisan több helyről is elérhető legyen az API által nyújtotta szolgáltatások, publikáltam az API-t a Microsoft vállalat által fejlesztett Azure, felhő alapú webszerverre, amelynek célja elősegíteni, hogy nagyobb

számú közösségekben is és akár más hálózaton levő eszközök esetében is kísérletezni lehessen a rendszerrel.

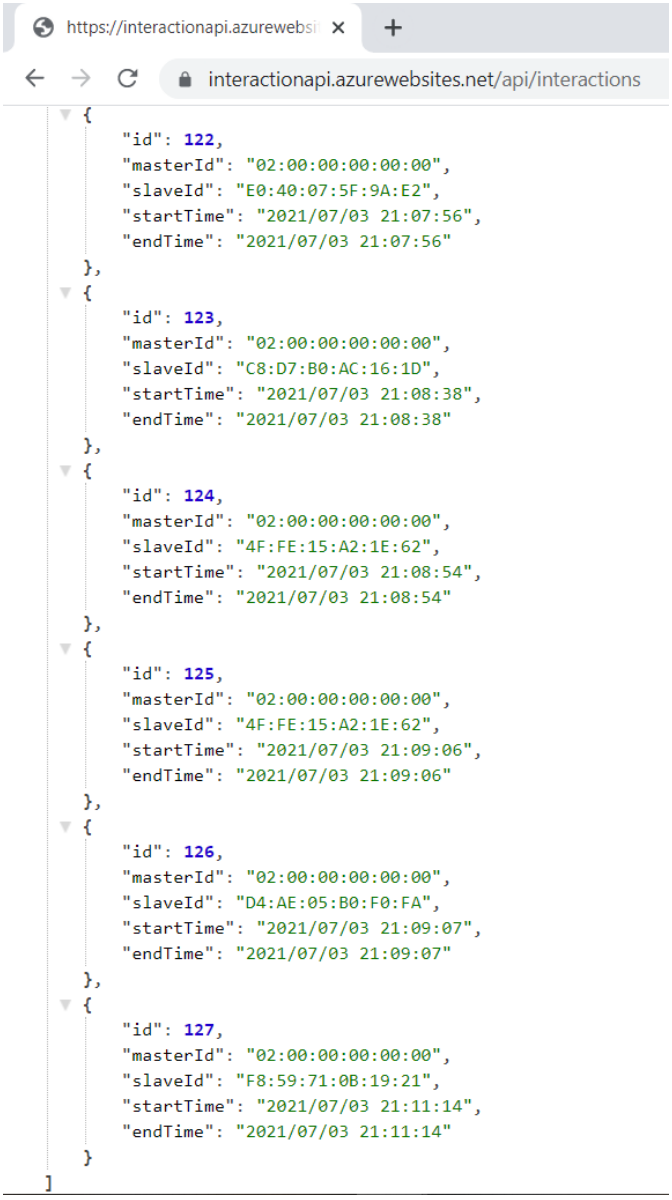
Végül pedig egy harmadik akadály, amibe az alkalmazás kísérleti eredményeinek vizsgálata során ütköztünk, az az volt, hogy bár a rendszer helyesen kéri le a környező eszközök Bluetooth azonosítóját, azonban az alkalmazást futtató eszköznek egy általánosan használt azonosítót észlel, különböző biztonsági okok érdekében. Ezt a biztonsági megoldást még sajnos nem sikerült megkerülni, ugyanis egy bizonyos Android verzió fölött ez a védelmi funkció alapból integrálva van a rendszerekben.

5.3. Kísérleti eredmények, mérések

Ebben a fejezetben bemutatásra kerül a rendszer tesztelése során elért eredmények, illetve az alkalmazás jelenlegi verziójának a bemutatása, néhány ábra megjelenítésével és azok elemzése során. Az alkalmazás üzembe helyezése során elért eredmények, melyeket otthoni környezetben, néhány eszköz által sikerült feljegyezni a böngészőben egy bárki számára használható URL cím segítségével követhető. Az eredmények megtekintése érdekében szükséges az adott cím ismerete, mely alapján nyomon követhetőek az alkalmazás felhasználói és a rendszer által elmentett interakciók is. A kísérlet során elért eredményeket az alábbiakban szemléltetem.

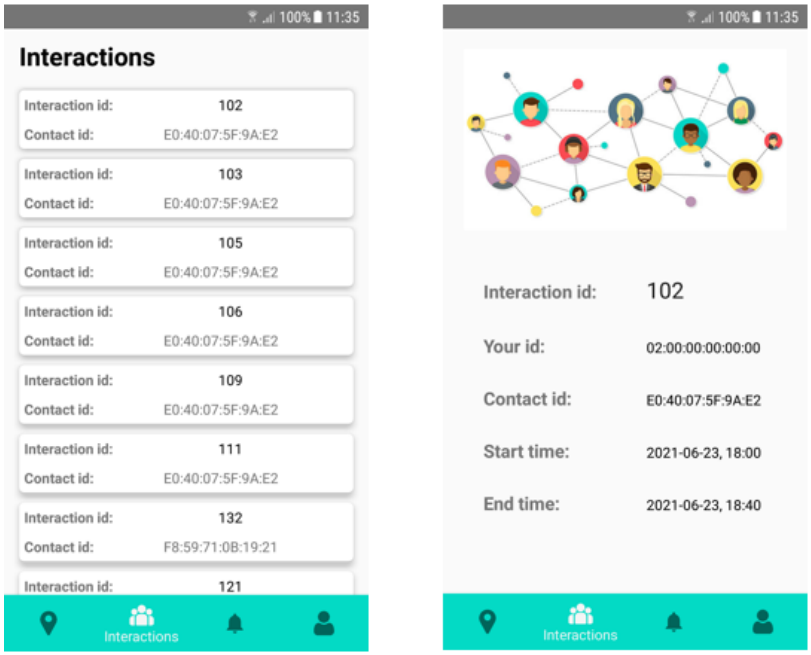
5.1.ábra: Bluetooth azonosító által regisztrált felhasználók megjelenítése a böngészőben

Az *5.1.ábrán* a böngészőben nyomon követhető, a rendszer által elmentett felhasználók láthatóak, az általuk használt eszköz Bluetooth azonosítójával és az egészségügyi állapotuk megjelenítésével. Az általam használt eszköz által érzékelt, hatókörben levő okostelefonok azonosítójával, manuálisan sikerült létrehozni az ábrán szemléltetett felhasználókat. Az alkalmazás jelenlegi verziója a bejelentkezést követően a Firebase [25] nevű adatbázisban menti el a regisztrált felhasználókat és azok adatait, azonban az *5.1.ábra* azt szemlélteti, hogy az alkalmazás továbbfejlesztésével egy adatbázis felhasználásával a felhasználók adatai elmenthetők, Bluetooth azonosító és egészségügyi státusz alapján. A továbbiakban egy újabb ábrán szemléltetem az alkalmazás által elmentett kontaktusokat, melyeket szintén a böngészőben vizsgáltunk meg.



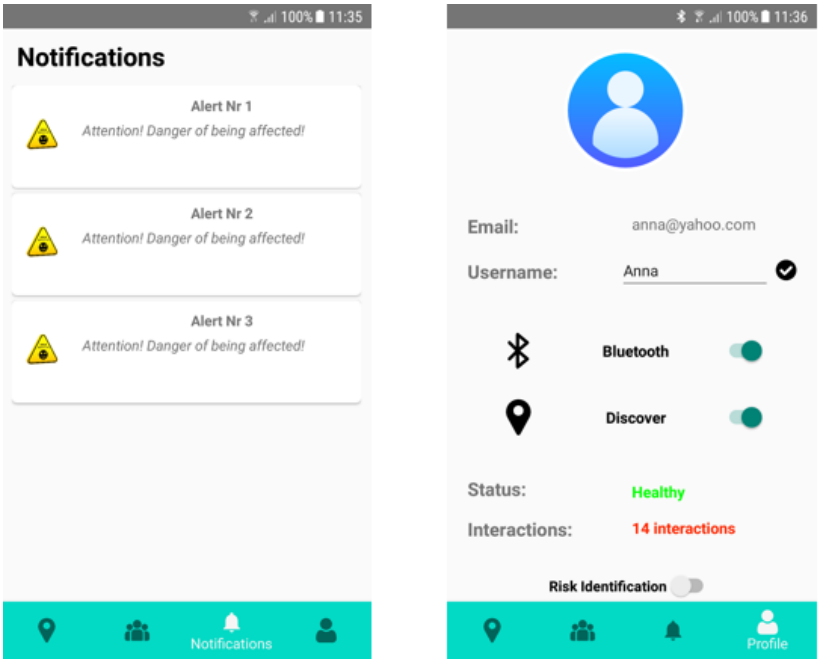
5.2.ábra: A kontaktusok megjelenítése a böngészőben

Az *5.2.ábra* szemlélteti az alkalmazás által a Bluetooth technológiának köszönhetően elmentett kontaktusokat, pontosabban a kísérlet során használt eszköz hatókörében levő eszközöket, amelyek megfeleltethetők, valóságos felhasználókként. Az alábbiakban az alkalmazás jelenlegi verziójának felhasználói felülete kerül bemutatásra. Az *5.3.ábra* szemlélteti az alkalmazás jelenlegi verziójába újdonságként implementált érintkezéseket megjelenítő menüpontot.



5.3.ábra: Az érintkezések megjelenítése az alkalmazásban

Az ábra bal felén látható az adott felhasználó kontaktusainak a listázása, az ábra jobb oldalán pedig egy adott interakció részletes leírása, Bluetooth azonosítók megjelenítésével, megőrizve ezáltal is a személyes adatok védelmét, illetve az észlelt érintkezés időpontjának megjelölésével. Az alábbiakban pedig az értesítéseket, illetve a profilt megjelenítő oldalak bemutatása következik.



5.4.ábra: Az Értesítések és a Profil oldal megjelenítése az alkalmazásban

Az *5.4.ábra* első felében az értesítéseket megjelenítő oldal látható, ami az alkalmazás jelenlegi verziójában általánosan generált üzeneteket jelenít meg a felhasználó számára. Az ábra második felében pedig a profil oldal jelenlegi megvalósítása látható, amelyen újdonságként megjelenítésre került az adott felhasználó az alkalmazás által addig észlelt kontaktusainak száma.

5.4. Az alkalmazás felhasználói felületének a tesztelése

A kísérleti eredmények összegzése mellett figyelmet fordítottunk a szoftver tesztelésének a fontosságára is, ugyanis a szoftverfejlesztésnek egyik legfontosabb folyamata a fejlesztés alatt álló szoftver folyamatos tesztelése. A tesztelés során fontos odafigyelni arra, hogy úgy írjuk meg a teszteket, hogy azok kiküszöböljék a rendszer hibalehetőségeit. A projekt megvalósítása során különböző tesztek megírásával, és azok az alkalmazás működése közbeni futtatásával teszteltük az Android alkalmazást. Ennek megvalósítására az Espresso keretrendszert használtam, amelyet kimondottan Android környezetben megírt alkalmazások felhasználói felületének tesztelése érdekében alkalmaznak [34].

Az Espresso célközönsége, azokat a felhasználókat jelenti, akik úgy vélik, hogy az automatizált tesztelés a fejlesztés életciklusának egy szerves részét képezi [34]. Az Android alkalmazások tesztelése esetén kétféle automatizált Android UI tesztet említünk. A UI teszt, avagy a felhasználói felület teszt egyetlen alkalmazással teszteli azt, hogy a célalkalmazás az elvárt módon viselkedik-e, amikor a felhasználó egy adott műveletet hajt végre vagy esetleg leállít egy tevékenységet. Így

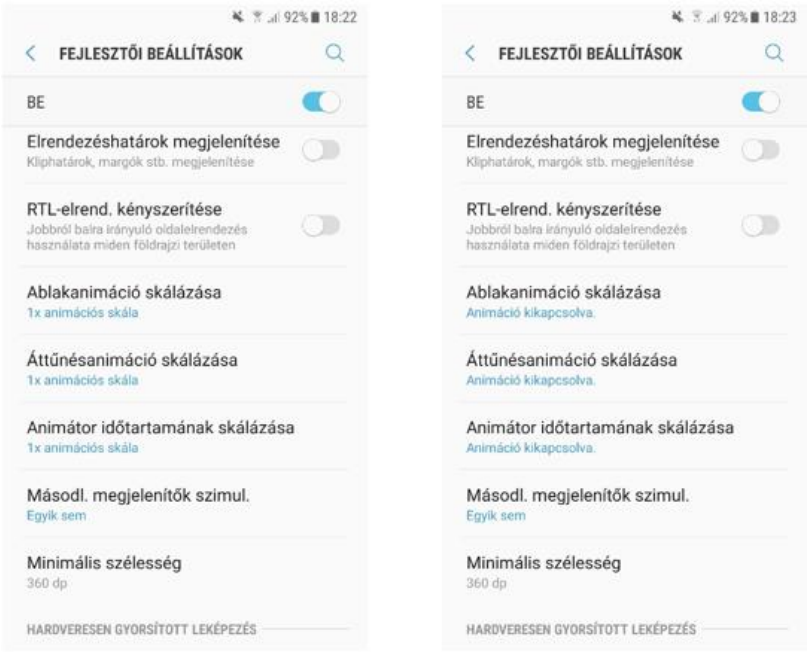
a UI teszteket végző keretrendszerek, mint amilyen az Espresso is, lehetővé teszik a felhasználói műveletek programozott szimulálását és az összetett, alkalmazáson belüli felhasználói interakciók tesztelését. A második tesztelési típus a UI teszt több alkalmazáshoz, amely ellenőrzi a különböző felhasználói alkalmazások és a rendszeralkalmazások közötti interakciók helyes viselkedését. Az Espresso tesztelési keretrendszer támogatja a teszt szkriptek írását Java és Kotlin nyelven egyaránt, ennek köszönhetően az Espresso segítségével tömör, szép és megbízható Android UI teszteket írhatunk [35]. A következőkben a saját alkalmazásom tesztelését szemléltetem az Espresso felhasználásával. Mielőtt saját tesztelési funkciókat írnánk a programba, elengedhetetlen bizonyos kezdő lépések megvalósítása az Espresso használatához. Az Espresso használata érdekében elengedhetetlen néhány fontos lépés elvégzése, amit a fejlesztőnek az Android Studioban kell elvégeznie. Míg más környezetekben számos kiegészítő programot, vagy keretrendszert kell telepíteni a fejlesztőknek, addig az Espresso esetében ezen lépések elkerülhetők, így rövidebb idő alatt integrálható az Espresso tesztelési keretrendszer. Az Espresso telepítése tehát az SDK Manager használatával és a Gradle használatával elvégezhető az alábbi lépések elvégzésével [34].

5.4.1. A tesztkörnyezet beállítása

A rétegesség elkerülése érdekében első lépésként ajánlott a rendszeranimációk kikapcsolása a teszteléshez használt virtuális vagy fizikai eszközön. Ez megtehető a használt eszközön a Beállítások menüpontban a Fejlesztői beállítások részben, ahol szükséges letiltani a következő három beállítást:

- Ablak animáció skála
- Átmeneti animációs skála
- Animátor időtartam skála

A tesztkörnyezet beállításának folyamatát az alábbi ábra szemlélteti.



5.5.ábra: A tesztkörnyezet beállításának ábrázolása

Az *5.5.ábra* az eszköz beállításainak a fejlesztői beállítások menüpontjában történő animációs folyamatok beállítását szemlélteti.

5.4.2. Az Espresso függőségek implementálása

A következő lépést már az Android Studioban szükséges elvégezni, ez pedig a függőségek hozzáadása lenne a projektünkhöz. Mindez megtehető az alkalmazás build.gradle állományában, ahová néhány kódsor bemásolásával és az alkalmazás újra szinkronizálásával a kezdeti lépések megvalósulnak.

5.4.3. Az „Instrumentation runner” implementálása

Szükséges, hogy ugyanezt a build.gradle állományt kiegészítsük egy olyan kódsorral, amely lehetővé teszi, az úgynevezett „instrumentation runner” implementálását aminek alapfeltétele, hogy az android.defaultConfig részbe kerüljön.

Ezen lépések segítségével implementáltuk az Espresso natív tesztelési környezetet, a következő két lépésben a tulajdonképpeni teszt megírása kerül bemutatásra, illetve az adott tesztek futtatása.

5.4.4. A tesztek megírása

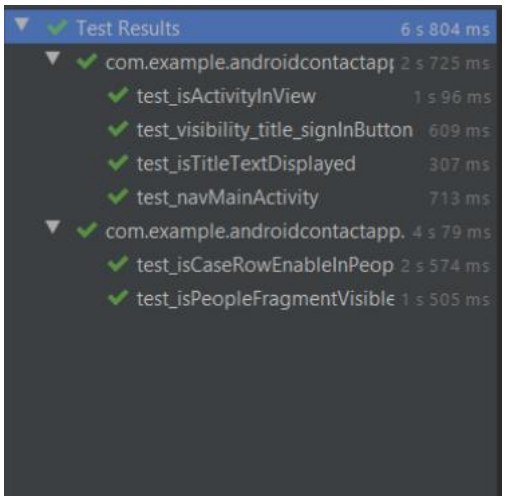
Amennyiben a beállításokat, az Espresso használatához szükséges kezdő lépéseket sikerült megtenni, következhetnek a tesztek megírása és azok futtatása a saját projektünkben. A tesztek

elősegítik annak ellenőrzését, hogy az alkalmazáson belül bizonyos funkciók helyesen működnek-e, vagy, hogy a komponensek léteznek-e az elvárt formában.

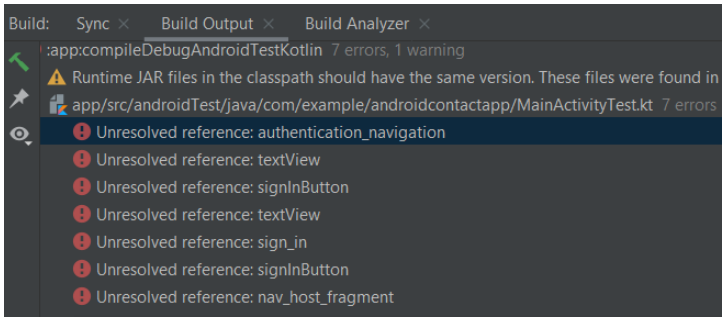
Kezdetben teszteltem az applikáció indításakor megjelenő első Activity megjelenését, az Authentication Activity-t, ami két képernyődarabkát kapcsol össze, a bejelentkezéshez és a regisztráláshoz szükséges Fragmenteket. Eredményként a teszt helyesen futott le, ugyanis a kezdő Activity valóban létezik és megjelenik az alkalmazás indításakor. A továbbiakban sor került az *Activity-k* közötti navigáció tesztelésére, ami esetemben két Activity közötti navigációt jelentette. Fontos volt annak ellenőrzése, hogy a bejelentkezést megvalósító Activity-ről sikeres legyen a navigáció a MainActivity-re. Az alkalmazásban számos funkciót megvalósító elem található, számos alkalmazás szerves részét képezi egy görgethető lista, az őt alkotó lista elemekből felépítve. A tesztelés során az alkalmazásban szereplő bejelentkezett személyek listájának megjelenítésére, illetve a benne megjelenő lista elemek megjelenésének az ellenőrzésére is sor került, ugyanis fontos, hogy az alkalmazás helyesen jelenítse meg a listákban levő elemeket..

Érdekességképpen megemlíthető az Espresso tesztelésnek azon lehetősége is, hogy a teszteket nem szükséges külön, egyenként, vagy akár osztályonként futtatni, hanem ennek egyszerűsítése érdekében adott az a lehetőség, hogy úgynevezett teszt csomagokat hozzunk létre, amely egy osztályban van összefoglalva, és annak futtatásának eredményeként az általunk megírt és a teszt csomagban meghívott összes osztály tesztje egyszerre fut le.

A tesztek futtatásának eredményeit az Android Studio parancssorában követhetjük figyelemmel, az alábbiakban egy helyes teszteredményeket jelző üzeneteket, illetve egy hibajelző üzenetet is szemléltetnek a példák.



5.6.ábra: Helyes teszteredmények a parancssorban



5.7.ábra: Hibás teszteredmények a parancssorban

Az általunk megírt teszteket futtatni lehet az Android Studio segítségével, vagy akár a parancssorból is (command line). Amennyiben az Android Studioból szeretnénk futtatni a tesztet, kezdetben a Run menüpontra kell navigáljunk, ezen belül pedig az Edit Configuration lehetőséget kell kiválasztanunk. Ekkor megnyílik egy új ablak, amelyben egy új Android Test konfigurációt tudunk létrehozni. Ekkor ki kell választani egy modult, illetve hozzá kell adni egy specifikus „instrumentation runner”-t. Végül pedig futtatni kell az új konfigurációt [34].

Előfordulhat az is, hogy a felhasználó a parancssorból (command line) szeretné futtatni az újonnan létrehozott teszteket, az Espresso erre is megoldást biztosít, ugyanis ez megoldható egyszerűen egy kódrészlet futtatásával [34].

6. A rendszer felhasználása

Az üzembe helyezés és a kísérleti eredmények részletes bemutatása után egyfajta összegzésképpen bemutatásra kerül a rendszer felhasználásának lehetőségei, amelyet ebben a fejezetben tárgyalunk. Elsősorban fontos megjegyezni azt, hogy a rendszer felhasználása érdekében a felhasználónak szükséges rendelkeznie egy a legalább Bluetooth technológiát implementáló okostelefonnal, melyen Android operációs rendszer fut. Továbbá az alkalmazás zavartalan működése érdekében minimális elvárása a rendszernek, hogy az adott készülék egy minimum 7.0-ás Nougat, Level API 23-as támogatottságú Android operációs rendszerrel rendelkezzen. Az alkalmazás jelenlegi verziójának telepítéséhez mindössze 25MB szabad tárhelyre van szükség, továbbá a készüléknek rendelkeznie kell minimum 1GB RAM memóriával, annak érdekében, hogy a felhasználói élmény teljességét tudja nyújtani az alkalmazás.

Amennyiben a potenciális felhasználó rendelkezik egy, az előbbiekben leírt eszközzel, lehetősége nyílik használatba venni a rendszer által biztosított lehetőségeket. A rendszer

felhasználásának szempontjából, arra a kérdésre választ megfogalmazva, hogy mire is használható a rendszer, az alábbiakban kerül sor. Az általunk megalkotott rendszer felhasználását tekintve két különálló lehetőséget elemezhetünk, ugyanis az alkalmazás felhasználható közösségi, illetve egyéni célokra egyaránt. A közösségi célok érdekében a rendszer a kontaktkutatással foglalkozó szakemberek, állami szervezetek munkáját segítheti elő, elősegítve ezáltal a közösség, többnyire egészségügyi biztonságának megőrzését. Az egyéni célokat tekintve pedig a rendszer felhasználható arra is, hogy az egyén követni tudja, hogy adott időegységen belül körülbelül hány személlyel kerül kapcsolatba rohanó világunk napjaiban. A rendszer fejlesztésével elérhetővé válhat az egyének számára, hogy akár saját mozgásukat is követni tudják, illetve figyelni arra, hogy hol kerülhetett kapcsolatba az egészsége számára veszélyt jelentő személyekkel. Úgy a közösségi, mint az egyéni célokra való tekintettel, a rendszer pontosabb, és biztosabb eredményeink elérésére szükséges, hogy nagyszámú felhasználói közösségek alkalmazzák szolgáltatásait.

Összegzésképpen a rendszer egy komplex változatának publikálása során, felhasználható lehet úgy egészségügyi szempontból, mint szociális szempontból is. Az egészségügyi szempont alatt az egyén és a közösség biztonságának megőrzését értem, illetve esetleges járványhelyzetek elkerülésének a lehetőségét, továbbá a szociális szempont alatt, egyfajta találkozások nyomon követését szolgáló rendszer használatát.

7. Következtetések

A rendszer felhasználását elemző fejezet után, és egyben a projekt dokumentálásának záró fejezetében a munkánk során elért eredmények és a céljaink összehasonlítása által megfogalmazhatók a megvalósítások, a hasonló rendszerekkel való összehasonlítás, illetve a továbbfejlesztési lehetőségek, egyszóval a következtetések. Ebben a fejezetben a munkánk során elért következtetéseink fogalmazódnak meg, melyeket a következő alpontok szerint tárgyalok.

7.1. Megvalósítások

A projekt során a következő megvalósításokra került sor, melyeket az alábbi alpontokban foglalunk össze:

- A felhasználók azonosítását lehetővé tevő műveletek: regisztrálás és bejelentkezés
- A Bluetooth technológia aktiválásának lehetősége az alkalmazásból

- Az eszköz Bluetooth alapú láthatóságának engedélyezése az alkalmazásból
- Az eszköz hatókörében levő, szintén a Bluetooth technológiát implementáló eszközök felderítésének lehetősége
- Az eszközök felderítése során, a Bluetooth azonosítok, illetve a találkozások idejének elmentése egy adatbázisban
- A rendszer által elmentett találkozások megjelenítése a böngészőben URL cím alapján
- A rendszer által elmentett találkozások megjelenítése az alkalmazásban
- Részletes információ megjelenítése a felhasználó egy adott kontaktusáról (Bluetooth azonosítók, a kontaktus észlelésének az ideje)
- A felhasználó egészségügyi állapotának a változtatási lehetősége az alkalmazásból
- Felhasználónév változtatásának a lehetősége az alkalmazásból
- Egy adott felhasználó találkozásainak számának a megjelenítése az alkalmazásban
- A felhasználó egészségügyi állapotának a megjelenítése az alkalmazásban
- A felhasználó tartózkodási helyének a meghatározása a GPS technológia segítségével
- Személyes adatok védelmének a megőrzése a GDPR irányelvek alapján

Az előző pontokban vannak összefoglalva azok a megvalósítások, melyeket a rendszer jelenlegi verziója implementál, így kiemelhető a dolgozat témájára nézve, a Bluetooth technológiának az implementálása és megvalósítása az Android alkalmazásban. Ezen technológia nyújtotta lehetőségek felhasználásával a rendszer képes érzékelni a hatókörében levő más felhasználók által használt eszközöket, felderítve ezáltal a lehetséges kontaktusokat. A Bluetooth hatókörének távolságából adódóan a rendszer által észlelt eszközök potenciális kontaktusokként értelmezhetők, ezen eszközök azonosítóinak az elmentése lehetőséget nyújt a kontaktkutatás megvalósításához.

Összegzésképpen elmondható, hogy a célkitűzések nagyrészének sikerült eleget tenni, így eredményként egy működőképes alkalmazást sikerült megvalósítani, mely egy a későbbiekben továbbfejleszthető, szélesebb körben használható alkalmazásnak az alap verziójaként szolgálhat.

7.2. Továbbfejlesztési lehetőségek

A projekt bemutatásának zárásaként, a továbbfejlesztési lehetőségeket szeretném elemezni, ugyanis a szoftverrendszerek tervezésének egyik kiemelkedő szakasza az, amikor megfogalmazódnak a rendszer javítását, fejlesztését elősegítő tervek, gondolatok.

Az általam készített rendszer egy a kontaktkutatás lehetőségét biztosító rendszernek az alapját képezhető szoftverrendszer, melynek továbbfejlesztésével egy mindennapi használatra alkalmas, illetve egy széleskörű kontaktkutatást végző alkalmazás fejleszthető, melyet állami szervezetek, illetve a kontaktkutatást végző szakemberek csoportja is felhasználhat munkájuk megkönnyítése érdekében.

A rendszer jelenlegi állapotát továbbfejlesztve elérhető a felfedezett kontaktusok automatizált észlelése, illetve azoknak egy meghatározott időkorlát átlépését követő elmentése. Továbbá az értesítést küldő funkció fejlesztésével elérhető lenne a célszemélyek értesítése is, annak alapján, hogy a felhasználó jelezte a rendszernek az alkalmazáson keresztül, hogy megváltozott egészségügyi állapota. Továbbfejlesztési lehetőségnek minősülne, a térkép oldalon a közelben levő felhasználók megjelenítése, Bluetooth azonosítójuk alapján, tisztességben tartva a felhasználók személyes adatait.

8. Irodalomjegyzék

[1] Mbunge, E. (2020). Integrating emerging technologies into COVID-19 contact tracing: Opportunities, challenges and pitfalls. *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, *14*(6), 1631-1636.

[2] Hernández-Orallo, E., Manzoni, P., Calafate, C. T., & Cano, J. C. (2020). Evaluating how smartphone contact tracing technology can reduce the spread of infectious diseases: the case of COVID-19. *IEEE Access*, *8*, 99083-99097.

[3] Universitat Politecnica De Valencia, “Smartphone contact tracing,” [Online]. Available: https://www.upv.es/noticias-upv/noticia-12105-rastreo-de-mov-en.html

[4] Rivest, R. L., Weitzner, D. J., Ivers, L. C., Soibelman, I., & Zissman, M. A. (2020). Pact: Private automated contact tracing. *Retrieved December*, *2*, 2020.

[5] Bluetooth Technology Website, “Bluetooth Technology Overview,” [Online]. Available: https://www.bluetooth.com/learn-about-bluetooth/tech-overview/

[6] Ericsson Website, “Bluetooth: Vezeték nélküli technológia,” [Online]. Available: https://web.archive.org/web/20071107022717/http://www.ericsson.com/hu/technology/bluetooth.shtml

[7] ELTE Website, “A GPS alapok,” [Online]. Available: http://lazarus.elte.hu/~climbela/gps31.htm

[8] Nwabueze, C. A., & Akaneme, S. A. (2009). Wireless Fidelity (Wi-Fi) Broadband Network Technology: An Overview with Other Broadband Wireless Networks. Nigerian Journal of Technology, 28(1), 71-78.

[9] European Union Website, “What is GDPR, the EU’s new data protection law?,” [Online]. Available: https://gdpr.eu/what-is-gdpr/

[10] Európai Bizottság weboldala, “Mit nevezünk személyes adatnak?,” [Online]. Available: https://ec.europa.eu/info/law/law-topic/data-protection/reform/what-personal-data_hu

[11] VírusRadar alkalmazás weboldala, “A Vírusradarról,” [Online]. Available: https://virusradar.hu/

[12] VírusRadar, “Play Store,” [Online]. Available: https://play.google.com/store/apps/details?id=hu.gov.virusradar&hl=en&gl=US

[13] TraceTogether alkalmazás weboldala, “TraceTogether, safer together,” [Online]. Available: https://www.tracetogether.gov.sg/index.html

[14] TraceTogether, “Play Store,” [Online]. Available: https://play.google.com/store/apps/details?id=sg.gov.tech.bluetrace&hl=en&gl=US

[15] Corona-Warn-App, “Play Store,” [Online]. Available: https://play.google.com/store/apps/details?id=de.rki.coronawarnapp&hl=en

[16] COVID Tracker Ireland alkalmazás weboldala, “Technology the COVID Tracker app uses,” [Online]. Available: https://covidtracker.ie/

[17] COVID Tracker Ireland, “Play Store,” [Online]. Available: https://play.google.com/store/apps/details?id=com.covidtracker.hse

[18] Google developer training website, “Android Developer Fundamentals (Version 2),” [Online]. Available: https://google-developer-training.github.io/android-developer-fundamentals-course-concepts-v2/

[19] Android developers website, “Platform Architecture,” [Online]. Available: https://developer.android.com/guide/platform

[20] Hellman, E. (2013). Android programming: Pushing the limits. John Wiley & Sons.

[21] AndroidPub website, “Consuming REST API using Retrofit Library in Android,” [Online]. Available: https://medium.com/android-news/consuming-rest-api-using-retrofit-library-in-android-ed47aef01ecb

[22] Thai, T., & Lam, H. (2003). . NET framework essentials. " O'Reilly Media, Inc.".

[23] Mukherjee, S. (2019). SQL Server Development Best Practices. International Journal of Innovative Research in Computer and Communication Engineering, 10.

[24] Wikipedia website, “Microsoft SQL Server,” [Online]. Available: https://en.wikipedia.org/wiki/Microsoft_SQL_Server

[25] Firebase website, “Firebase,” [Online]. Available: https://firebase.google.com/

[26] Coronavirus COVID19 API website, “Postman,” [Online]. Available: https://documenter.getpostman.com

[27] Tutorials Teacher website, “What is Web API?,” [Online]. Available: https://www.tutorialsteacher.com/webapi/what-is-web-api

[28] FreeCodeCamp website, “An awesome guide on how to build RESTful APIs with ASP.NET Core,” [Online]. Available: https://www.freecodecamp.org/news/

[29] Swagger website, “API Development for Everyone,” [Online]. Available: https://swagger.io/

[30] Postman website, “The Collaboration Platform for API Development,” [Online]. Available: https://www.postman.com/

[31] Azure website, “Power your vision on Azure,” [Online]. Available: https://azure.microsoft.com/en-us/

[32] GitHub website, “GitHub,” [Online]. Available: https://github.com/

[33] GitHub Desktop website, “GitHub,” [Online]. Available: https://github.com/

[34] Android developers website, “Espresso,” [Online]. Available: https://developer.android.com/training/testing/espresso

[35] Test Automation University’s website, “Android Test Automation with Espresso,” [Online]. Available: https://testautomationu.applitools.com/espresso-mobile-testing-tutorial/

[36] GitHub website, “GitHub,” [Online]. Available: https://github.com/mitchtabian/

9. Függelék

- Forráskód
- Turnitin similarity report
- Tájékoztatás a személyes adatok feldolgozásáról (GDPR)

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE, TÎRGU-MUREȘ
SPECIALIZAREA CALCULATOARE

Vizat decan
Conf. dr. ing. Domokos József

Vizat director departament
Ș.l. dr. ing. Szabó László Zsolt