**SVEUČILIŠTE U ZAGREBU**

**FAKULTET ORGANIZACIJE I INFORMATIKE**

**V A R A Ž D I N**

**Tim: AIR1613**

**Paula Kokić/ Acerinth**

**Ivan Pokec/ ivanpokec**

**Žana Zekić/ zanzekic**

**GitHub repozitorij:** [**https://github.com/ivanpokec/Indoor-Tracking**](https://github.com/ivanpokec/Indoor-Tracking)

**INDOOR TRACKING**

**Projekt iz kolegija Analiza i razvoj programa**

**TEHNIČKA DOKUMENTACIJA**

**Mentor:**

doc.dr.sc. Zlatko Stapić

**Varaždin, siječanj 2017.**

**Sadržaj**

[1. Korištene tehnologije i alati 1](#_Toc473637767)

[2. Arhitektura aplikacije i sustava 2](#_Toc473637768)

[2.1. Arhitektura sustava 2](#_Toc473637769)

[2.2. Arhitektura aplikacije Indoor Tracking 3](#_Toc473637770)

[3. Podatkovni model 4](#_Toc473637771)

[4. IndoorTracking WEB API 5](#_Toc473637772)

[5. Dijagram klasa 6](#_Toc473637773)

[6. Implementirane funkcionalnosti 8](#_Toc473637774)

[6.1. Prijava i odjava korisnika 8](#_Toc473637775)

[6.2. Pozicioniranje korisnika u zgradi 8](#_Toc473637776)

[6.3. Povijest kretanja 9](#_Toc473637777)

[6.4. Pregled svih lokacija 10](#_Toc473637778)

[6.5. Pregled svih korisnika 10](#_Toc473637779)

[6.6. Pregled i djelomično uređivanje vlastitog profila 10](#_Toc473637780)

[7. Testiranje aplikacije 11](#_Toc473637781)

1. Korištene tehnologije i alati

U sklopu ovog projekta, odlučili smo koristiti sljedeće tehnologije i alate:

* Android Studio (v 2.2) – razvojno okruženje za Android OS
* VivifyScrum (<https://www.vivifyscrum.com>) – online alat za agilno upravljanje projektom i kolaboracijom koristeći Scrum metodu
* GitHub (<https://github.com/>) – online sustav za verzioniranje programskog koda
* SourceTree – alat za lakše upravljanje kodom u skladu s GitHub sustavom
* VisualParadigm – alat za izradu UML dijagrama
* VisualStudio – razvojno okruženje za .NET i C# aplikacije
* MS SQL Management Studio – alat za upravljanje MS SQL bazom podataka

Svi korišteni alati su besplatni za preuzimanje i korištenje. VivifyScrum, VisualStudio i MS SQL Management Studio dobiveni su besplatno putem studentske licence.

Projekt je impementiran za Android 4.4 KitKat (API level 19).

1. Arhitektura aplikacije i sustava
   1. Arhitektura sustava

Na Slici 2. prikazana je arhitektura cijelog sustava.

Slika 1. Dijagram arhitekture sustava

Aplikacija Indoor Tracking instalira se na Android pametni telefon svakog korisnika. Aplikacija zahtijeva korištenje Bluetooth i Wi-Fi tehnologija. Nakon pokretanja aplikacije, uređaj može putem Bluetooth-a učitavati obližnje ICD uređaje, koji su postavljeni po jedan u svakoj prostoriji zgrade. Nakon određivanja najbližeg ICD uređaja i dohvaćanja njegove MAC adrese, aplikacija putem Interneta, preko Web API sučelja šalje zahtjev sustavu za upravljanje bazom podataka (MS SQL Server). Sustav preko istog API-ja vraća odgovor aplikaciji, odnosno naziv lokacije na kojoj se nalazi korisnik. Kada se promijeni lokacija, odnosno najbliži ICD uređaj, postupak se ponavlja na isti način. Kada korisnik putem aplikacije zatraži bilo kakve podatke (npr. o korisnicima, postojećim lokacijama i sl.), podaci se na isti način dohvaćaju preko API-ja iz baze podataka te šalju korisniku na pametni telefon.

* 1. Arhitektura aplikacije Indoor Tracking

Aplikaciju čine 4 modula: App modul, DbAccess modul, BLE modul (eng. *Bluetooth Low Energy*) te Core modul. Slika 1. prikazuje navedenu arhitekturu te veze između modula.



Slika 2. Arhitektura aplikacije - moduli

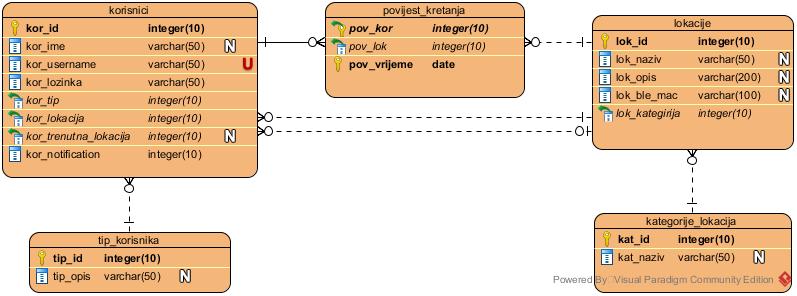
App modul implementirat će korisničko sučelje te će služiti za prikaz podataka, koje dobiva od Core modula. Core modul implementira sva sučelja i klase potrebne za komuniciranje svih modula, tako da je on u vezi za svim modulima. BLE modul sadrži klase koje implementiraju algoritme za učitavanje ICD uređaja te algoritme za određivanje najbližeg ICD uređaja, odnosno određivanje korisničke lokacije. DbAccess modul sadrži web API za pristupanje serverskoj MS SQL bazi podataka, budući da baza nije javna te se nalazi iza sigurnosne stijene (eng. *firewall*) pa joj se ne može direktno pristupiti.

Navedeni opis odnosi se na aplikaciju koja se instalira na svaki Android pametni telefon, pojedinačno za svakog korisnika.

1. Podatkovni model

Na Slici 3. prikazan je podatkovni model (ERA dijagram) aplikacije.

Slika 3. ERA model



Tablica *korisnici* sadrži sve potrebne podatke o korisnicima aplikacije. Kao primarni ključ koristit će se atribut *kor\_id*, korisničko ime sprema se u atribut *kor\_username*, a ostali atributi služe za općeniti opis pojedinog korisnika. *Kor\_ime* predstavlja ime i prezime korisnika, *kor\_lozinka* njegovu lozinku, *kor\_tip* predstavlja tip korisnika, *kor\_lokacija* predstavlja korisnikovu primarnu lokaciju (njegovu radnu sobu), *kor\_trenutna\_lokacija* predstavlja lokaciju na kojoj se korisnik trenutno nalazi, dok se u *kor\_notification* sprema korisnikovo dopuštenje o primanju obavijesti.

Tablica *tip\_korisnika* sadrži kategorije korisnika, za potrebe kategoriziranja korisnika (obični korisnik, šef odjela, direktor i sl.).

Tablica *lokacije* sadrži podatke o lokacijama (prostorijama) u zgradi. Primarni ključ je *lok\_id*, a svaku lokaciju dodatno opisuju sljedeći atributi: *lok\_naziv* predstavlja naziv prostorije, *lok\_opis* predstavlja njezin opis, *lok\_ble\_mac* predstavlja MAC adresu BLE uređaja koji je postavljen u toj prostoriji (i prema kojemu se zapravo određuje položaj korisnika) i *lok\_kategorija* predstavlja vanjski ključ na kategoriju lokacije.

Tablica *kategorija\_prostorija* sadrži nazive kategorija prema kojima su svrstane prostorije, radi lakše pretrage u aplikaciji.

Navedeni podatkovni model impelementiran je kao zasebna baza podataka unutar MS SQL Servera tvrtke Mobilisis. Podacima se pristupa preko Web API servisa koji će biti opisan u sljedećem poglavlju.

1. IndoorTracking WEB API

IndoorTracking Web API je RESTful online servis preko kojeg se dohvaćaju podaci iz baze koja se nalazi iza sigurnosne stijene (eng. *firewall*) te nema javni pristup. REST (eng. *Representational State Transfer*) ili RESTful web servisi su jedan od načina za pružanje interoperabilnosti računalnih sustava. Ovaj servis omogućuje dohvaćanje i manipulaciju podacima preko predefiniranih setova operacija. Ovom servisu se pristupa preko URI adrese http://development.mobilisis.hr/IndoorTracking/api te ovisno o poslanim parametrima se vraća odgovor u JSON obliku ili u obliku zadanom u poslanom zaglavlju HTTP zahtjeva. Unutar servisa su implementirane GET i POST metode. Kako bismo dobili željene podatke, kreiraju se klase kontrolori koje služe za upravljanje zahtjevima i klase modeli koje predstavljaju oblik odgovora koji se vraća na zahtjev. Unutar klasa kontrolera je implementirana komunikacija s bazom i vraćanje podataka korisniku. Kako bismo dohvatili željene podatke o lokaciji moramo preko POST zahtjeva na http//development.mobilisis.hr/IndoorTracking/api/location poslati u JSON obliku MAC adresu najbližeg uređaja npr.:{ 'mac':'A0:E6:F8:4F:AF:BC'} te ćemo ako je tražena MAC adresa nađena u bazi, kao odgovor dobiti polje koje sadrži jedan element čija struktura je definirana u klasi Location unutar servisa. Prema tome, za našu MAC adresu će biti vraćen odgovor:

[{

"Id": 1,

"name": "soba1 T1",

"macAddress": "A0:E6:F8:4F:AF:BC",

"description": "T1 ICD"

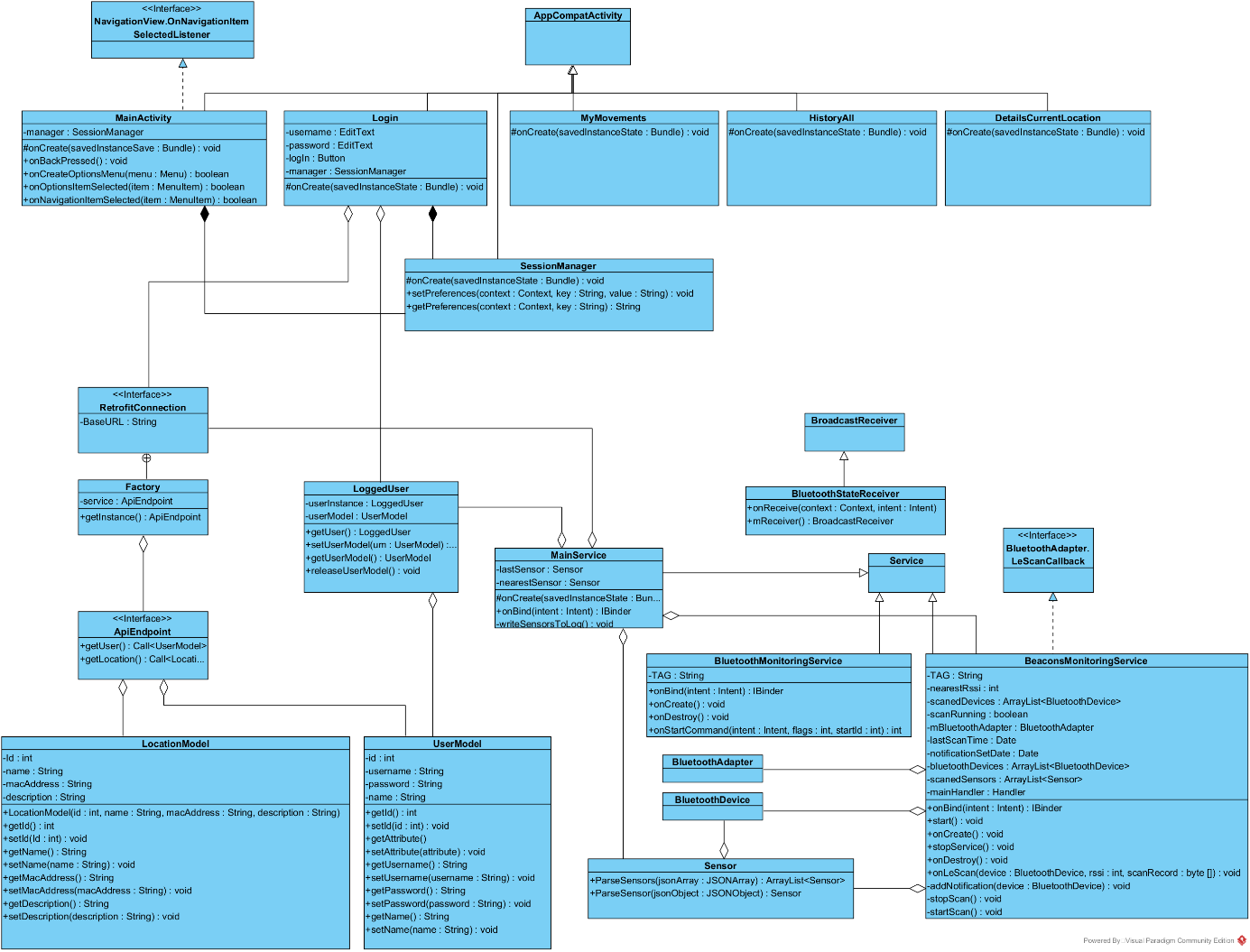
}]

Popis svih kontrolera se nalazi u tablici koja je u nastavku.

|  |  |  |  |
| --- | --- | --- | --- |
| Putanja | Vrsta | Parametri | Opis |
| History/GetAllHistoryForUser | POST | UserId | Vraća cijelu povijest kretanja za traženog korisnika |
| History/GetDateForUser | POST | UserId | Vraća listu datuma za koje postoji neki zapis unutar tablice *povijest\_kretanja* |
| History/GetHistoryForDateAndUser | POST | UserId i date | Vraća popis kretanja nekog korisnika za određeni dan |
| History/GetHistoryFromTo | POST | UserId, dateFrom i dateTo | Vraća popis kretanja korisnika za određeni period |
| History/GetHistoryForUserLocation | POST | UserId i locationId | Vraća datum i vrijeme kada je korisnik bio na određenoj lokaciji |
| Location | GET |  | Vraća listu svih lokacija i podatke o njima |
| Location | POST | MacAddress, UsrId | Služi za zapis lokacije u *povijest\_kretanja* i za dohvaćanje podataka o lokaciji na kojoj se korisnik nalazi |
| UsersOnLocation | POST | locationId | Služi za dohvaćanje svih korisnika koji su trenutno na nekoj lokaciji |
| Category | GET |  | Vraća listu svih kategorija |
| LocationInCategori | POST | catId | Vraća listu svih lokacija koje se nalaze unutar određene kategorije |
| UserPassUpdate | POST | passWord i usrId | Služi za ažuriranje korisnikove lozinke |
| User | GET |  | Vraća listu svih korisnika i njihove podatke |
| User | POST | UserId | Vraća podatke o određenom korisniku |
| UserLocation | POST | locationId | Vraća sve korisnike koji bi trebali biti na lokaciji kojoj su pridruženi, a trenutno se ne nalaze na njoj |
| Login | POST | userName i passWord | Služi za prijavu u aplikaciju i vrača sve podatke o prijavljenom korisniku, zajedno sa njegovim postavkama o primanju obavijesti |
| UserUpdateNotification | POST | Notification i usrId | Služi za ažuriranje postavki o primanju obavijesti |

Tablica 1. Tablica kontrolera za pozive prema IndoorTracking API-u

1. Dijagram klasa

Na Slici 4. prikazan je dijagram klasa za aplikaciju Indoor Tracking.

Slika 4. Dijagram klasa

Na dijagramu su navedene klase koje su trenutno implementirane u projektu, a navedene su i neke općenite Android klase i sučelja koja se koriste (AppCompatActivity, Service, BluetoothAdapter, BluetoothDevice i sl.) te njihovi atributi i metode nisu navedene.

Klasa MainActivity prikazuje početni zaslon aplikacije (u budućnosti Trenutnu lokaciju), te su u njoj implementirane metode za upravljanje glavnim izbornikom aplikacije.

Klasa Login prikazuje zaslon za prijavu (prije zaslona MainActivity klase) i zadužena je za prijavu korisnika i kreiranje njegove sesije nakon uspješne prijave. Zato ona koristi klasu RetrofitConnection koja se spaja na IndoorTracking Web API servis i provjerava podatke za prijavu, odnosno vraća sve ostale podatke o korisniku koje nam trebaju.

Klasa SessionManager zadužena je za pohranjivanje korisničke sesije, jer izlazak iz aplikacije ne znači odjavu korisnika. Prijavljeni korisnik se sprema u sesiju, i prilikom ponovnog pokretanja aplikacije ne traži se prijava, već se odmah otvara zaslon MainActivity. Ipak, kada se klikne na gumb „Odjava“, korisnik se odjavljuje, sesija se briše i prilikom ponovnog pokretanja aplikacije traži se korisnička prijava.

Klasa BluetothMonitoringService provjerava da li je došlo do kakvih promjena vezanih uz sam bluetooth na uređaju, te bilježi te promjene. Unutar klase Sensor se nalazi struktura podataka vezana uz BLE beacone te upravljanje njihovim promjenama, dok se u klasi BeaconsMonitoringService nalaze metode koje se cijelo vrijeme izvršavaju u pozadini i detektiraju BLE beacone, čitaju njihove podatke te ih zapisuju u liste iz kojih se zatim određuje koji je beacon najbliži i koja je njegova MAC adresa. Važno je napomenuti da smo ovu klasu, kao i sve ostale klase koje sudjeluju u algoritmu detekcije BLE uređaja te određivanja najbližeg uređaja, dobili gotove na slobodno korištenje od tvrtke Mobilisis. Bilo je potrebno samo uskladiti neke metode, odnosno izvući iz tih klasa ono što je nama bilo potrebno za aplikaciju Indoor Tracking te integrirati s ostatkom koda.

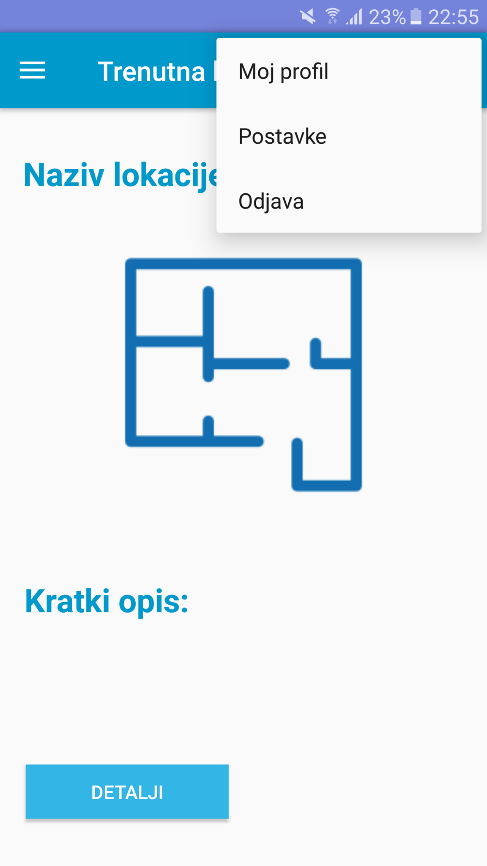
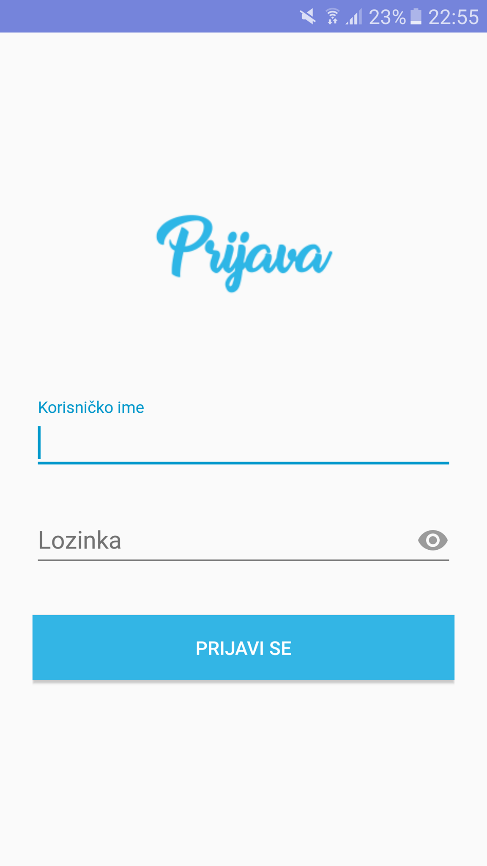
Sučelja RetrofitConnection i ApiEndpoint služe za pristupu bazi podataka. LocationModel i UserModel su klase koje odgovaraju entitetima lokacija i korisnik iz baze podataka te sadrže odgovarajuće atribute koje ih opisuju. Klasa LoggedUser je klasa namijenjena pristupu klasi UserModel, koja sadrži podatke o logiranom korisniku. Klasa MainService je klasa koja povezuje sve algoritme za nadzor i određivanje najbližeg senzora, te javlja ostalim klasama kada dođe do promjene lokacije. Klase MyMovements, DetailsCurrentLocation i HistoryAll još su uvijek u razvoju, a služit će realizaciji prikaza pojedinih funkcionalnosti.

1. Implementirane funkcionalnosti

U nastavku će biti prikazani ekrani implementiranih funkcionalnosti aplikacije, prema zahtjevima korisnika specificiranim u Projektnoj dokumentaciji.

* 1. Prijava i odjava korisnika

Prilikom pokretanja aplikacije prikazuje se zaslon za prijavu. Korisnik mora unijeti ispravno korisničko ime i lozinku kako bi mogao početi s korištenjem aplikacije. Za registraciju korisnika potrebno se javiti u tvrtku Mobilisis. Na Slici 5. prikazan je ekran za prijavu korisnika, a na Slici 6. prikazan je izbornik iz kojeg je dostupna opcija za odjavu korisnika.

**

Slika 6. Odjava korisnika

Slika 5. Prijava korisnika

* 1. Pozicioniranje korisnika u zgradi

Pozicioniranje korisnika u zgradi obuhvaća sljedeće 2 funkcionalnosti:

* prikaz korisnikove trenutne lokacije i njezinih detalja („Trenutna lokacija“)
* prikaz obavijesti o promjeni lokacije

Na Slici 7. nalazi se prikaz glavnog izbornika, odnosno lista svih implementiranih funkcionalnosti. Na Slici 8. prikazan je početni ekran aplikacije, tj. „Trenutna lokacija“. Prikazuje se naziv lokacije, kategorija kojoj ona pripada te kratki opis. Sličica pokazuje gdje se korisnik nalazi u okviru kata zgrade. Klikom na gumb „Detalji“ prikazuje se popis korisnika koji se trenutno nalaze na toj lokaciji te popis korisnika kojima je ta lokacija pridružena kao njihov ured (a trenutno se ne nalaze tamo), kao što je prikazano na Slici 9.

Na Slici 10. prikazan je ekran s primljenom obavijesti o promjeni lokacije. Ako se obavijest ne ukloni, nakon dolaska u novu lokaciju, obavijest će samo ažurirati svoje podatke o lokaciji u kojoj se korisnik nalazi.

* 1. Povijest kretanja

Povijest kretanja obuhvaća sljedeće funkcionalnosti:

* spremanje podataka o kretanju korisnika (kada je bio na kojoj lokaciji)
* mogućnost pregleda vlastite povijesti kretanja, filtriranje pregleda prema datumu (vremenskom rasponu) i prema lokaciji.

Prilikom svake posjete korisnika novoj lokaciji, u bazu podataka na serveru spremaju se podaci o njegovoj posjeti, tj. datum, vrijeme i mjesto.

Pregled vlastite povijesti kretanja dostupan je odabirom opcije „Moja kretanja“ iz glavnog izbornika. Otvara se ekran kao na Slici 11., gdje su dostupne tri opcije prikaza povijesti. Prva opcija je prikaz cijele povijesti: najprije se ispisuju datumi, a klikom na datum možemo dobiti ispis vremena i mjesta, kao što je prikazano na slikama 12. i 13.

Druga opcija je filtriranje prikaza prema datumu – nakon odabira određenog vremenskog raspona, otvara nam se ekran s listom datuma, vremena i mjesta iz odabranog raspona, kao što je prikazano na Slici 14.

Treća opcija je filtriranje prikaza prema lokaciji – nakon odabira lokacije, otvara se ekran s listom datuma i vremena kada je korisnik bio na odabranoj lokaciji, kao što je prikazano na Slici 15.

Funkcionalnost pregleda vlastite povijesti kretanja implementirali smo modularno. Kreirali smo sučelje kojim smo definirali metodu za prihvaćanje podataka te četiri različite klase (dvije služe za prikaz cijele povijesti, te po jedna za filtriranje po datumu i po lokaciji) koje implementiraju to sučelje. Kreirana je jedna klasa koja sadrži adapter i ostale elemente za prikaz podataka na ekranu, koja ovisno o pritisnutom gumbu u prethodnoj aktivnosti, instancira objekt jedne od ove četiri klase i poziva metodu za dohvaćanje podataka nad tim objektom.

* 1. Pregled svih lokacija

Pregled svih lokacija je obuhvaća funkcionalnosti ispisa svih dostupnih lokacija te ispisa korisnika koji se trenutno nalaze na tim lokacijama, odnosno onih korisnika kojima je ta prostorija definirana kao ured (a trenutno se ne nalaze tamo). Na Slici 16. prikazan je ispis svih lokacija, te na Slici 17. ispis svih korisnika odabrane lokacije.

* 1. Pregled svih korisnika

Pregled svih korisnika obuhvaća ispis svih korisnika te prikaz detalja o odabranom korisniku. Na Slici 18. prikazan je ispis korisnika, a na Slici 19. prikaz detalja o odabranom korisniku.

* 1. Pregled i djelomično uređivanje vlastitog profila

Iz izbornika kojeg smo vidjeli na početku ovog poglavlja, na Slici 6., dostupna je opcija „Moj profil“ za pregled vlastitog profila. Tu se nalazi korisničko ime, ime i prezime, lozinka, ured (odjel kojemu pripada), te gumb za odjavu, kako je prikazano na Slici 20. Lozinku je moguće promijeniti samim klikom na nju, kao što je prikazano na Slici 21.

1. Testiranje aplikacije

Aplikaciju je potrebno testirati na odgovarajućim Android uređajima, s instaliranim API-jem 19 ili više. Prije pokretanja aplikacije, potrebno je uključiti Bluetooth, osigurati valjanu internetsku vezu te dozvoliti upotrebu lokacije.

Pri prvom pokretanju, aplikacija će zahtijevati autetikaciju. Podaci za prijavu koji se mogu koristiti jesu:

korisničko ime: *zzekic*

lozinka: *zanaz*

Kako bi mogli u potpunosti koristiti sve funkcionalnosti aplikacije, potrebno je nalaziti se u blizini barem jednog ICD uređaja koji je registriran u bazu podataka aplikacije. Za naše potrebe razvoja i testiranja aplikacije, tvrtka Mobilisis nam je posudila na slobodno korištenje svoje vlastite ICD uređaje.