Java programski jezik i REST API za razvoj aplikacija za Android operacijski sustav

Tomić, Antonijo

VELEUČILIŠTE U VIROVITICI

Preddiplomski stručni studij Računarstvo

ANTONIJO TOMIĆ

JAVA PROGRAMSKI JEZIK I REST API ZA RAZVOJ APLIKACIJA ZA ANDROID OPERACIJSKI SUSTAV

VIROVITICA, 2023.

VELEUČILIŠTE U VIROVITICI

Preddiplomski stručni studij Računarstvo

JAVA PROGRAMSKI JEZIK I REST API ZA RAZVOJ APLIKACJIA ZA ANDROID OPERACIJSKI SUSTAV

Predmet: Projektiranje informacijskih sustava

Mentor: Student:

Ivan Heđi,dipl.ing.,v.pred. Antonijo Tomić

VIROVITICA, 2023.

Veleučilište u Virovitici

Stručni prijediplomski studij Računarstva - Smjer Programsko inženjerstvo

OBRAZAC 1b

ZADATAK ZAVRŠNOG RADA

Student/ica: **ANTONIJO TOMIĆ** **JMBAG:** **0307017309**

Imenovani mentor: Ivan Heđi, dipl. ing., v. pred.

Imenovani komentor: -

Naslov rada:

Java programski jezik i REST API za razvoj aplikacija za Android operacijski sustav

Puni tekst zadatka završnog rada:

U Vašem radu opišite Android operacijski sustav i programski jezik Java za izradu aplikacija. Kao praktični primjer implementacije teorijskog dijele osmislite i izradite aplikaciju pomoću koje studenti/studentice mogu jednostavno pronaći stan i/ili cimera/cimericu. Za slanje obavijesti korisnicima implementirajte sustav Firebase Cloud Messaging. Za dohvaćanje podataka iz baze podataka implementirajte REST API. Za bazu podataka koristiti MSSQL. Osim programskog rješenja, u pisanom dijelu završnog rada opišite ukratko korištene tehnologije, opišite detaljno arhitekturu sustava te opišite detaljno odabrane procese i/ili funkcije unutar same aplikacije. Prilikom opisivanja, osim neformalnih koristite i neke od formalnih metoda koje poznajete.

Datum uručenja zadatka studentu/ici: 31.07.2023.
Rok za predaju gotovog rada: 08.09.2023.

Mentor:

Ivan Heđi, dipl. ing., v. pred.

Dostaviti:
1. Studentu/ici
2. Povjerenstvu za završni rad - tajniku

JAVA PROGRAMSKI JEZIK I REST API ZA RAZVOJ APLIKACJIA ZA ANDROID OPERACIJSKI SUSTAV

Sažetak

Cilj ovog rada je izrada mobilne aplikacije koja služi za olakšavanje pronalaska stana ili cimera. Aplikacija je izrađena za android operacijski sustav koristeći Java programski jezik, koji je jedan od najrazvijenijih programskih jezika u androidu. Pored toga za implementaciju i upravljanje API-jem koji komunicira između aplikacije i servera, aplikacija koristi .NET Entity Framework Core verzije 3.1 pisan u programskom jeziku C#, ova kombinacija tehnologija omogućila je stvaranje aplikacije te brzi i jednostavni pristup podacima. Funkcionalnosti aplikacije uključuje pretraživanje i prikaz željenog stana ili cimera, korištenje raznih filtera putem kojih se olakšava pronalazak željenog smještaja ili cimera, dodavanje vlastitog stana koji je vidljiv drugim korisnicima ili prikaz vlastitog profila u kategoriji cimera. Također aplikacija podržava funkciju slanja i primanja poruke direktno u aplikaciji ili putem vanjskih aplikacija poput Gmail-a.

Ključne riječi: *Stan, mobilna aplikacija, korisnik, cimer, android aplikacija, java, c#*

JAVA PROGRAMMING LANGUAGE AND REST API FOR ANDROID APPLICATION DEVELOPMENT

Abstract

The goal of this work is to create a mobile application designed to facilitate the search for apartments or roommates. The application is developed for the Android operating system using the Java programming language, which is one of the most widely used programming languages in the Android ecosystem. In addition, for the implementation and management of the API that communicates between the application and the server, the application uses .NET Entity Framework Core version 3.1 written in the C# programming language. This combination of technologies enables the creation of the application and provides fast and easy access to data.

The application's functionalities include searching and displaying desired apartments or roommates, using various filters to simplify the process of finding suitable accommodation or roommates, adding one's own apartment, which is visible to other users, or displaying one's own profile in the roommate category. Furthermore, the application supports the feature of sending and receiving messages directly within the application or through external applications such as Gmail.il-a.

Keywords: *Apartment, Mobile application, user, roomate, Android appliication, java, c#*

Sadržaj

1 Uvod

Mobilni uređaj je nekada koristio kao uređaj za komuniciranje među ljudima međutim s vremenom je tehnologija napredovala pa se mobilni uređaj počeo koristiti za puno drugih stvari stoga je korisno razviti aplikaciju koja se nalazi na mobilnom uređaju. Krajem 2008. godine izašao je prvi telefon na Android operacijskom sustavu, a do sada broji više od dvije milijarde aktivnih korisnika. Najpopularniji programski jezici za Android aplikacije su Java i Kotlin.

REST API (engl. Representational state transfer application programming interface) je arhitektura razvoj web usluga koja se često koristi u razvoju Android aplikacija. REST API omogućuje komunikaciju između mobilne aplikacije i udaljenih servera putem HTTP zahtjeva. Kombinacija Java programskog jezika i REST API-ja omogućava programerima da razvijaju moćne i funkcionalne Android aplikacije koje mogu komunicirati s različitim uslugama i izvorima podataka putem interneta. Ovaj spoj pruža fleksibilnost i skalabilnost za razvoj aplikacija na Android platformi.

U ovome radu će biti opisane tehnologije i arhitekture korištene za izradu ove aplikacije, razvojna okruženja, baze podataka, te detaljna funkcionalnost aplikacije.

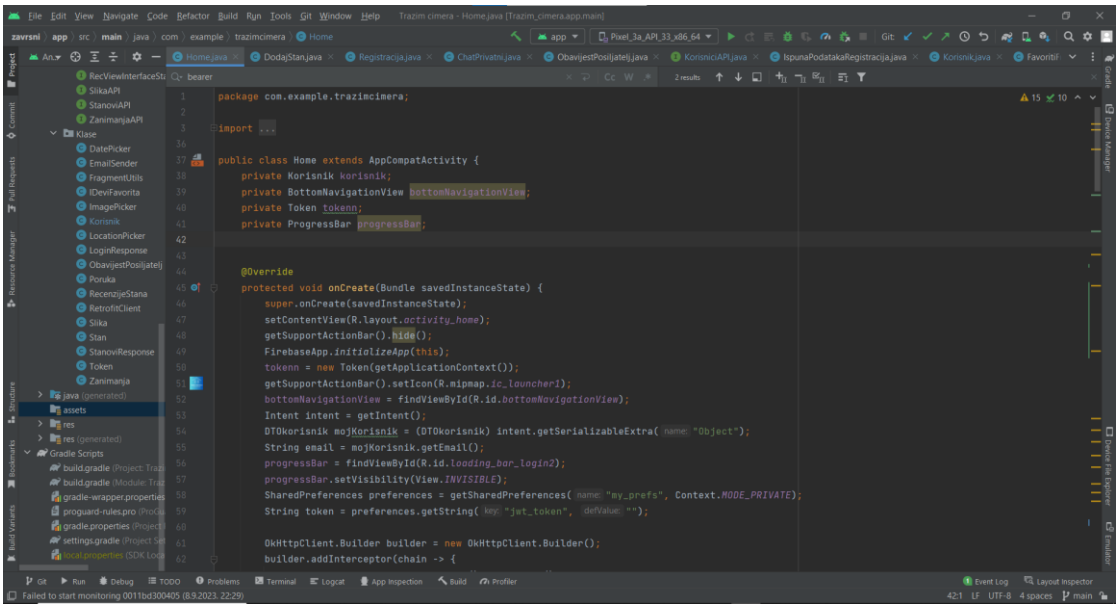
2 Programske tehnologije i alati

2.1 Razvojna okruženja

Razvojna okruženja su softverski alati koji programerima pružaju sveobuhvatan skup funkcionalnosti za razvoj, testiranje i održavanje softvera. Ona kombiniraju alate poput uređivača koda, debuggera, kompajlera, testera, integracije s bazama podataka i drugih korisnih funkcionalnosti. Ta okruženja omogućavaju programerima učinkovito pisanje, uređivanje i testiranje koda, olakšavajući proces razvoja aplikacija.

2.1.1 Android Studio

Android Studio (slika 1) je integrirano razvojno okruženje (IDE) namijenjeno razvoju Android aplikacija. Izradio ga je Google i programerima pruža mnoge prednosti. Neke od ključnih prednosti Android Studija uključuju: fleksibilan sustav izrade temeljen na Gradle-u, C++ i NDK podrška, Ugrađeni debugger za ispravljanje pogrešaka koji pomaže u otkrivanju i ispravljanju pogrešaka koda, čime se olakšava testiranje i poboljšava kvaliteta aplikacija, Intuitivno korisničko sučelje za dizajniranje korisničkog sučelja aplikacija, omogućuje programerima da vizualno organiziraju i prilagode izgled aplikacija, integracija s Gitom za upravljanje verzijama koda i olakšavanje suradnje s drugim članovima tima, ugrađena podrška za Google Cloud Platform, što olakšava integraciju Google Cloud Messaging i App Engine



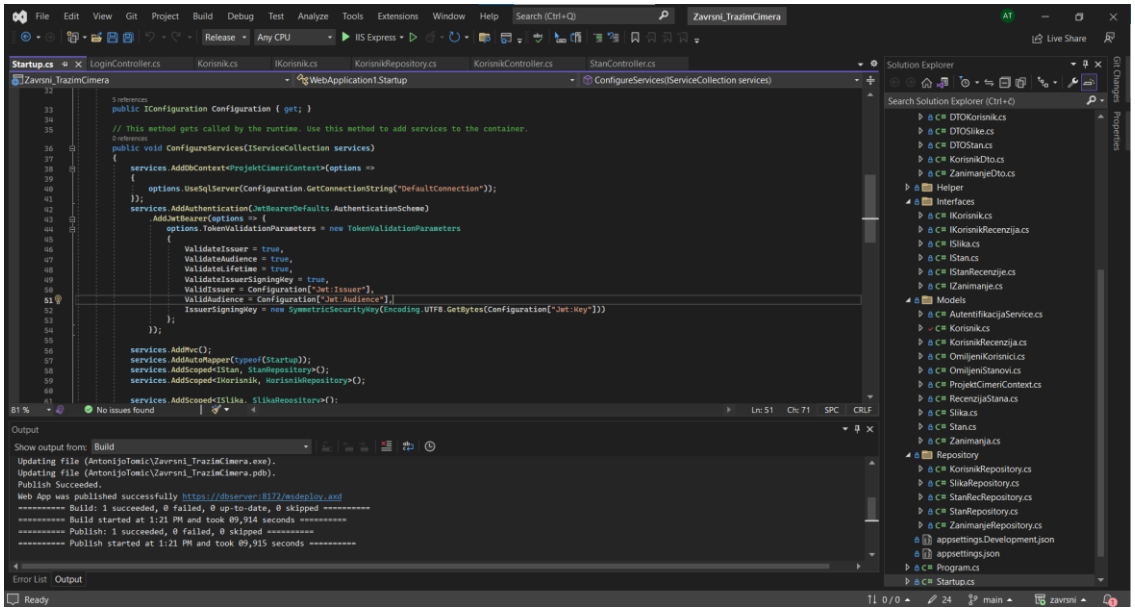
Slika 1. Izgled alata Android studio

Android Studio (slika 1) je prvi put objavljen 2013. godine kao zamjena za prethodni alat za razvoj Android aplikacija, Eclipse. Od tada se redovito ažurira i poboljšava kako bi podržao najnovije verzije platforme Android i pružio bolje iskustvo za programere.

2.2 Visual Studio

Visual Studio (slika 2) je razvojno okruženje (IDE) koje je razvio Microsoft. Prva verzija Visual Studija je objavljena 1997. godine. Glavne prednosti Visual studia su:

- Široka jezična podrška: Visual Studio podržava širok raspon programskih jezika, uključujući C#, Visual Basic, C++, F#, Python i mnoge druge, što olakšava razvoj različitih vrsta aplikacija.
- Integracija s Microsoftovim ekosustavom: Visual Studio integriran je s Microsoftovom platformom, što omogućuje jednostavnu integraciju s drugim alatima i uslugama, kao što su Azure cloud, Team Foundation Server i Git.
- Napredno otklanjanje pogrešaka: IDE pruža moćne alate za otklanjanje pogrešaka, uključujući izvođenje korak po korak, prijelomne točke, praćenje varijabilnih vrijednosti i još mnogo toga, što olakšava pronalaženje i ispravljanje grešaka u vašem kodu.
- Bogat skup alata: Visual Studio dolazi sa širokim rasponom alata za upravljanje projektima, dizajn korisničkog sučelja, testiranje i optimizaciju performansi aplikacija.



Slika 2. Izgled alata Visual studio

- Velika korisnička zajednica: Visual Studio ima veliku i aktivnu korisničku zajednicu koja pruža podršku, dijeli resurse i doprinosi širenju znanja o razvoju aplikacija.

2.3 Java

Java za Android je programski jezik koji se koristi za razvoj mobilnih aplikacija za Android platformu. Java je popularna zbog svoje jednostavnosti, čitljivosti koda i platformne neovisnosti. Razvoj Android aplikacija u Javi omogućava programerima pristup velikoj zajednici i brojnim alatima koji olakšavaju proces razvoja. Također, Java za Android koristi Android API-je, što omogućava pristup različitim funkcionalnostima uređaja kao što su kamera, senzori i mrežno povezivanje.

Programeri koriste razvojna okruženja poput Android Studio kako bi napisali, testirali i izgradili aplikacije. Ova okruženja pružaju bogat skup alata i podršku za razvoj Android aplikacija. Java za Android je važan jezik za razvoj mobilnih aplikacija, jer omogućava programerima da kreiraju visokokvalitetne, funkcionalne i interaktivne aplikacije za Android platformu.

2.4 XML

XML (engl. Extensible Markup Language) je jezik označavanja koji se često koristi u Android razvoju za definiranje korisničkog sučelja (UI) i konfiguracijskih datoteka.

U kontekstu Androida, XML se često koristi za definiranje izgleda korisničkog sučelja pomoću XML datoteka koje se nazivaju "layout" datoteke. Layout datoteke opisuju strukturu i izgled elemenata korisničkog sučelja, kao što su gumbi, tekstualna polja, slike itd. Koristeći XML, možete definirati hijerarhiju elemenata, postaviti njihove atribute (npr. veličinu, boju, raspored) i definirati interakciju između njih.

Isječak programskog koda 1. Primjer gumba u xml-u

```
1. <Button
2.     android:id="@+id/myButton"
3.     android:layout_width="wrap_content"h
4.     android:layout_height="wrap_content"
5.     android:text="Click me!" />
```

2.5 C#

C# je moderni programski jezik razvijen od strane Microsofta koji podržava objektno orijentirano programiranje (OOP) i ima bogatu biblioteku i okvire za razvoj različitih vrsta aplikacija. On je tipiziran jezik s automatskim prikupljanjem smeća (Garbage Collection) za upravljanje memorijom. C# također podržava LINQ (Language Integrated Query), što omogućava programerima jednostavno i izražajno rukovanje podacima. Ovaj jezik je posebno popularan u razvoju web aplikacija, desktop aplikacija i mobilnih aplikacija putem različitih .NET platformi.

2.5.1 ASP.NET Core Web API (3.1)

ASP.NET Core Web API je dio .NET Core platforme i okvir za razvoj web API-ja. Ovaj okvir omogućava programerima da izgrade skalabilne, performantne i sigurne web API-je koje mogu koristiti klijenti kao što su web aplikacije, mobilne aplikacije ili drugi servisi. Ovaj framework nudi visoke performanse i čistu organizaciju koda, pružajući programerima alate za dokumentaciju i testiranje. Također, nudi naprednu podršku za autentikaciju i autorizaciju kako bi se osigurala sigurnost API resursa. ASP.NET Core Web API je idealan izbor za razvoj RESTful API-ja i aplikacija koje zahtijevaju pouzdanu komunikaciju između različitih sistema i platformi..

Kontroleri su klase u okviru ASP.NET Core Web API-ja koje obavljaju logiku i obradu zahtjeva od klijenta. Kontroleri su odgovorni za prihvaćanje HTTP zahtjeva, izvršavanje odgovarajućih akcija i generiranje odgovora.

Kontroleri u ASP.NET Core Web API-ju obično sadrže javne metode koje se nazivaju akcijama (actions). Svaka akcija odgovara određenom HTTP zahtjevu (GET, POST, PUT, DELETE itd.). Na primjer, može postojati akcija "GetAll" koja se izvršava kada se šalje GET zahtjev za dohvaćanje svih resursa, ili akcija "Create" koja se izvršava za POST zahtjev za stvaranje novog resursa.

Isječak programskog koda 2. Primjer kontrolera za dohvaćanje ratinga korisnika

```
1.  1.        [HttpGet("{korisnikID}/rating")]
2.  2.         [ProducesResponseType(200, Type = typeof(decimal))]
3.  3.         [ProducesResponseType(400)]
4.  4.         public IActionResult GetKorisnikRating(int korisnikID)
5.  5.         {
6.  6.
7.  7.             if (!_KorisnikRepository.KorisnikExists(korisnikID))
8.  8.                 return NotFound();
```

```
 9.  9.
10. 10.             var rating = _KorisnikRepository.GetKorisnikRating(korisnikID);
11. 11.
12. 12.             if (!ModelState.IsValid)
13. 13.                 return BadRequest(ModelState);
14. 14.
15. 15.             return Ok(rating);}
```

Modeli u ASP.NET Core Web API-ju su klase koje predstavljaju podatke ili entitete s kojima aplikacija radi. Ovi modeli definiraju strukturu podataka koje API čita, koristi za obradu zahtjeva ili vraća kao odgovor. Modeli se često koriste za provjeru podataka koji se unose ili primaju putem HTTP zahtjeva. Također se mogu koristiti za automatsko mapiranje podataka iz baze podataka u oblik pogodan za API i obratno. Modeli igraju ključnu ulogu u procesu prijenosa podataka i omogućuju jasnu organizaciju i strukturu podataka u aplikaciji.

Rute u ASP.NET Core Web API-ju definiraju kako će se dolazni HTTP zahtjevi usmjeriti na odgovarajuće akcije u kontrolerima. One određuju URL putanju i metodu HTTP za svaku akciju.

2.6 Postman

Postman je popularan alat za testiranje i razvoj API-ja (engl. (Application Programming Interface)). Pruža korisnicima intuitivno sučelje koje omogućava slanje HTTP zahtjeva na različite rute te primanje odgovora od poslužitelja. Jedna od ključnih značajki Postmana je mogućnost slanja različitih vrsta HTTP zahtjeva kao što su POST, PUT, GET, DELETE. Također se mogu prilagoditi parametri tijela zahtjeva, zaglavlja i razne druge opcije kako bi se testirale i simulirali različiti scenariji komunikacije s API-jem

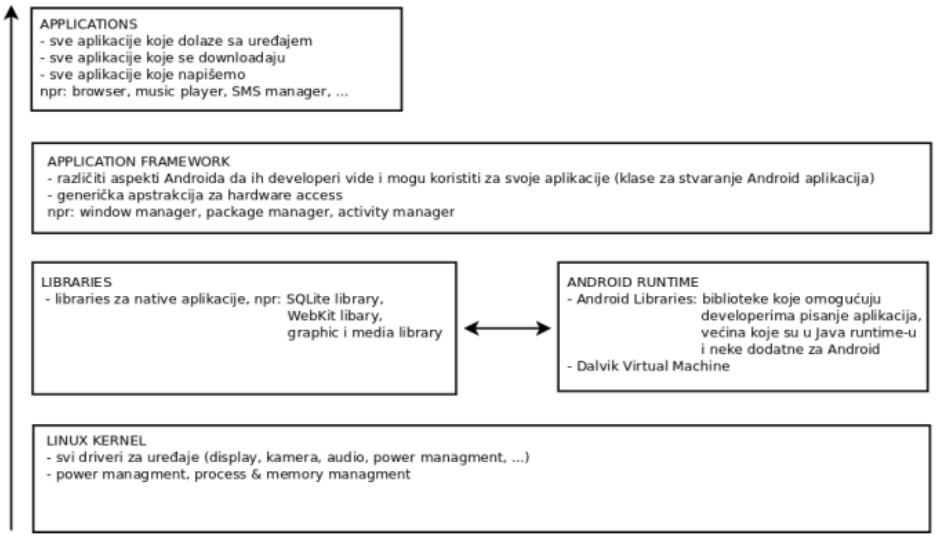
3 Android operacijski sustav

Android je otvoreni operacijski sustav za mobilne uređaje, kao što su pametni telefoni i tablet računala, američke tvrtke Google Inc. temeljen na jezgri Linux i drugom softveru otvorenog kôda. Nadalje, Google je razvio i Android TV za pametne televizore, Android Auto za automobile i Wear OS za pametne satove.

Android omogućuje korisnicima pristup velikom broju aplikacija putem Google Play trgovine. Korisnici mogu preuzeti i instalirati različite vrste aplikacija, uključujući igre, alate za produktivnost, društvene mreže, multimedijski sadržaj i mnoge druge. Također, Android pruža integrirane usluge i funkcionalnosti kao što su sinkronizacija s Google računom, višenamjenski sustav obavijesti, podrška za više korisničkih računa, Google Now osobni asistent, podrška za različite vrste veza (Wi-Fi, mobilni podaci, Bluetooth itd.) i mnoge druge funkcije.

3.1 Android arhitektura

Android arhitektura je složen sustav koji omogućava izvršavanje Android aplikacija. Ključni dijelovi uključuju Linux jezgru za upravljanje hardverom, Android Runtime za izvršavanje aplikacija, korisničko sučelje za interakciju s korisnikom te okvir za aplikacije za razvoj. Također, Android ima sustav za upravljanje aplikacijama, sigurnosni sloj i podršku za različite API-je i okvire za razvoj. Ova arhitektura omogućava izvođenje različitih aplikacija na Android uređajima.



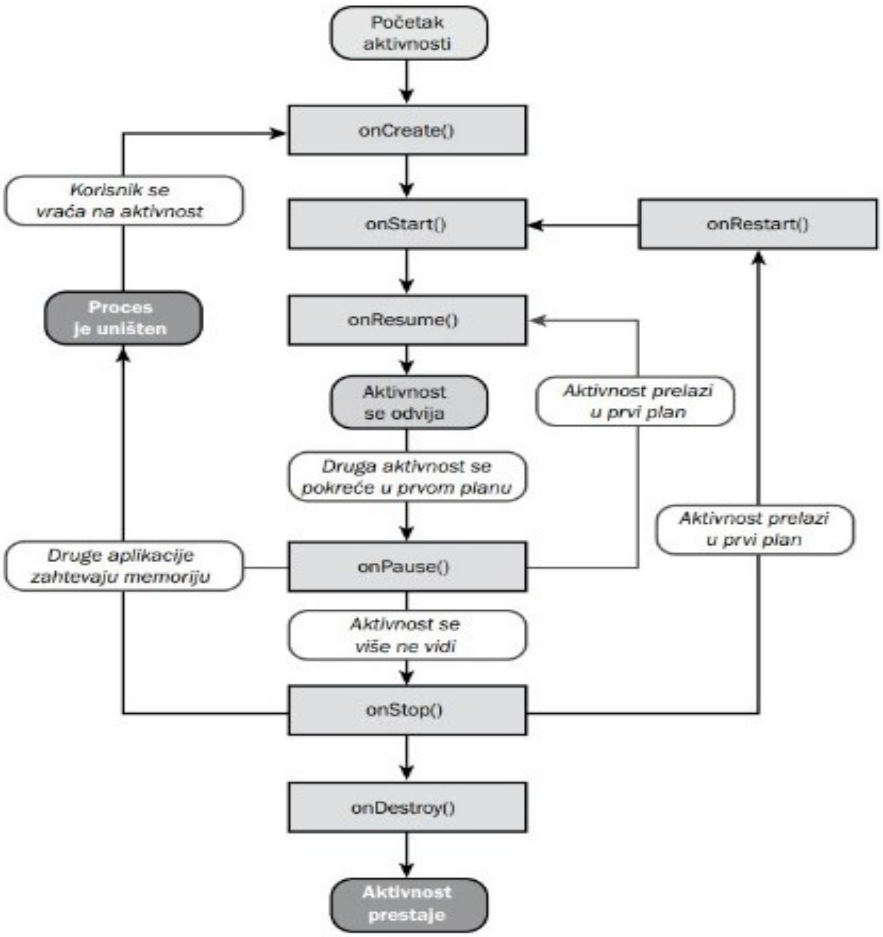
Slika 3. Arhitektura Android sustava prikazana po slojevima [10]

Arhitektura android sustava [10] (slika 3) sastoji se od 5 djelova koji su smješteni u 4 sloja :

- Linux Kernel
- Libraries, Android Run Time (Biblioteke, Android vrijeme rada)
- Application Framework (Aplikacijski okvir)
- Applications (Aplikacije)

3.2 Životni ciklus

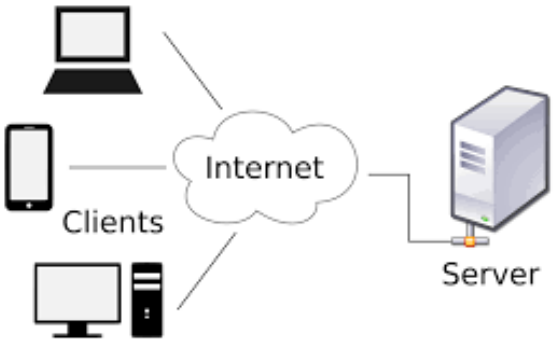
Android životni ciklus (slika 4) odnosi se na seriju događaja i stanja kroz koje prolazi Android aplikacija tijekom svog vremena izvršavanja. Razumijevanje životnog ciklusa ključno je za pravilno upravljanje resursima i interakcijom s korisnikom.



Slika 4. Životni ciklus android aplikacije [11]

3.3 Klijent – poslužitelj

Klijent-poslužitelj arhitektura[12] (slika 5) je jedna od najčešće korištenih arhitektonskih obrazaca. Ova arhitektura se temelji na podjeli uloga i odgovornosti između klijentskog računalnog uređaja (klijent) i poslužiteljskog sustava (poslužitelja). Glavna prednost ovakve strukture je razdvajanje poslovne logike i korisničkog sučelja. Poslužitelj je odgovoran za obradu poslovne logike, pristupanje podacima i izvođenje zahtjevnim operacijama. Klijent s druge strane je odgovoran za prikazivanje korisničkog sučelja, prikazivanje podataka i interakciju s korisnikom. Također klijent-poslužitelj arhitektura omogućava skalabilnost, fleksibilnost i lakše održavanje sustava. Komunikacija između klijenta i poslužitelja obično se odvija preko mreže koristeći protokol kao što je HTTP.



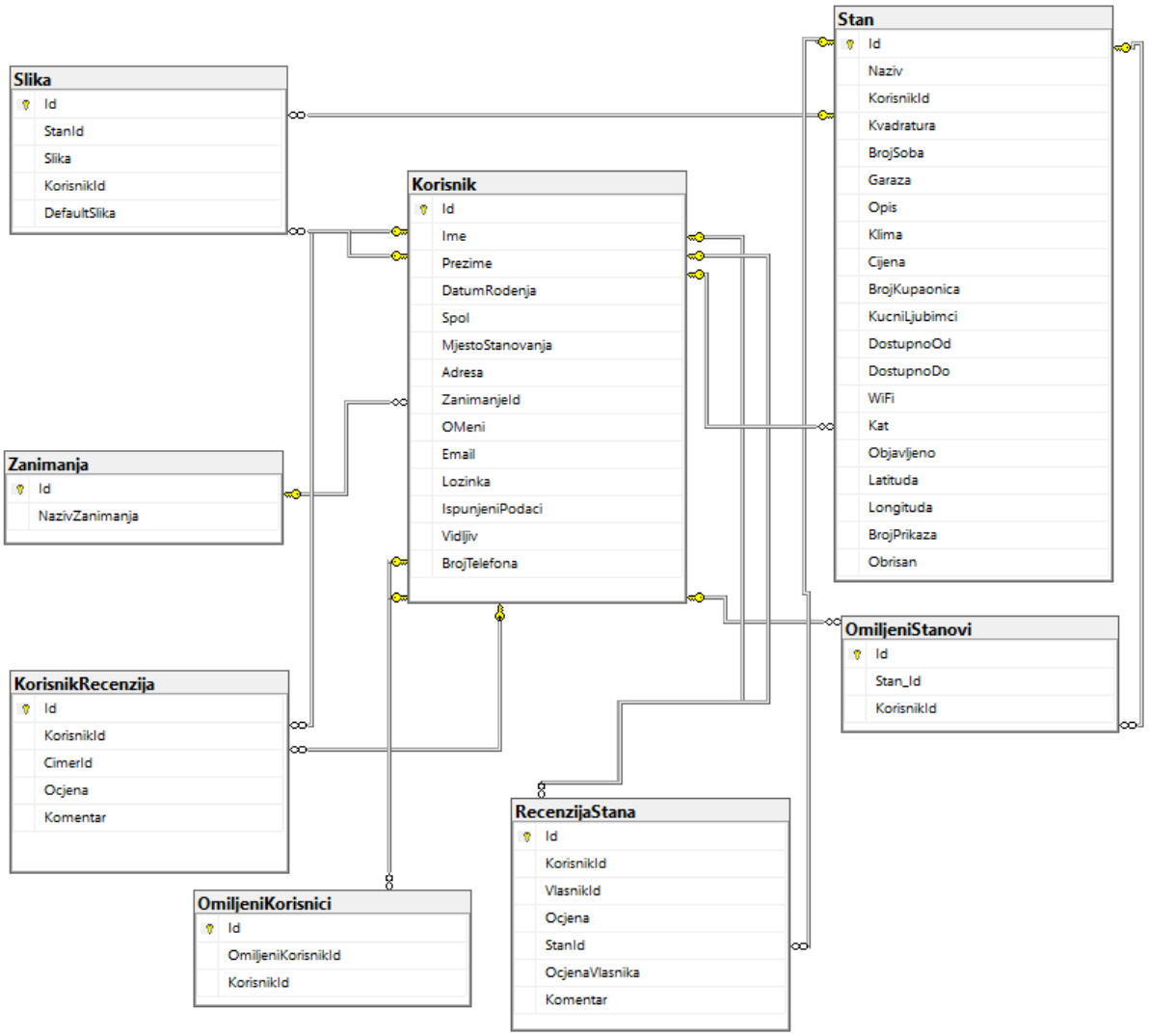
Slika 5. Klijent - poslužitelj arhitektura [12]

Klijent je računalni entitet ili korisnik koji inicira zahtjeve prema poslužitelju. U kontekstu klijent-poslužitelj arhitekture, klijent je entitet koji koristi usluge ili resurse koje pruža poslužitelj.

Poslužitelj (server) je računalni entitet ili sustav koji prima zahtjeve od klijenata i pruža tražene resurse ili usluge. U kontekstu klijent-poslužitelj arhitekture, poslužitelj je odgovoran za obradu zahtjeva i generiranje odgovora koji se šalju natrag klijentu.

4 Baza podataka

Baza podataka je organizirana kolekcija podataka koja je dizajnirana za efikasno pohranjivanje, upravljanje i pristup podacima. Baza podataka omogućava strukturiranu pohranu podataka, što olakšava upravljanje i manipulaciju podacima u organizirani način. Baze podataka koriste se za čuvanje različitih vrsta podataka, kao što su tekstualni podaci, brojevi, slike, zvukovi ili bilo koji drugi oblik digitalnih informacija. One omogućavaju upite, ažuriranje, brisanje i upisivanje podataka u skladu s definiranim pravilima i ograničenjima.



Slika 6. Dijagram baze podataka projekta

Dijagram baze podataka je vizualna reprezentacija strukture i relacija između entiteta u bazi podataka. Ključni elementi digrama baze podataka uključuju: tablice, polja, ključeve, veze,

tipove podataka. U dijagramu baze podataka aplikacije (slika 6) postoje 8 tablica, a dvije najbitnije tablice su tablica „Korisnik“ i „Stan“, te dvije tablice imaju najviše veza i polja.

4.1 Firebase

Firebase je platforma za razvoj mobilnih i web aplikacija koja pruža skup alata i usluga za olakšavanje razvoja, upravljanja i skaliranja aplikacija. Ova platforma omogućuje razvoj mobilnih aplikacija za Android, iOS i web koristeći zajedničku backend infrastrukturu.

Firebase pruža različite usluge koje su ključne za razvoj aplikacija, uključujući: Realtime bazu podataka: Ovo je cloud-based baza podataka koja omogućuje sinkronizaciju podataka u stvarnom vremenu između različitih klijentskih uređaja, Autentikacija: Firebase pruža jednostavnu integraciju za provjeru autentičnosti korisnika putem različitih metoda prijave, uključujući e-poštu i lozinku, društvene mreže poput Google i Facebook, te sustave za upravljanje identitetom poput Firebase Authentication, Hosting: Firebase omogućuje jednostavno pohranjivanje statičkih web stranica i web aplikacija na brzim i sigurnim Firebase Hosting serverima, Cloud Functions: Ova usluga omogućuje izvršavanje prilagođenog koda na serveru bez potrebe za upravljanjem vlastitim backend infrastrukturom. Koristeći Firebase Cloud Functions, možete reagirati na događaje, obrađivati podatke i izvršavati operacije na serveru.

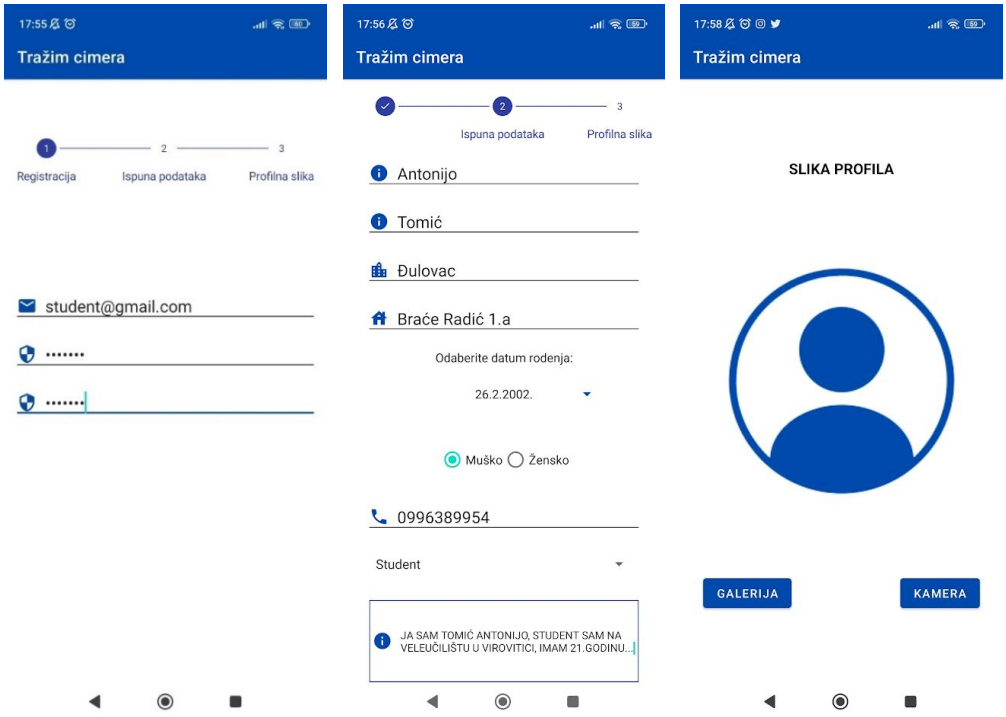
U projektu se koristi Realtime za funkcionalnost slanja i prikaza poruka u stvarnom vremenu.

5 Android aplikacija „Tražim cimera ili stan“

5.1 Osnovne funkcionalnosti registracije i prijave

Kako bi se aplikacija mogla koristiti obavezna je registracija odnosno kreiranje računa. Registracija se odvija u 3 koraka:

- Unos emaila i lozinke
- Unos osobnih podataka
- Dodavanje slike profila



Slika 7. Unos podataka kod registracije

Za postavljanje slike profila postoje 2 načina:

- Odabir fotografije iz galerije
- Uslikati putem kamere

Prilikom prijave u aplikaciju, dohvaćaju se uneseni email i lozinka te se kreira novi objekt klase DTOkorisnik. Kreira se OkHttpClient.Builder objekt s Interceptorom koji dodaje zaglavlje "Content-Type" u HTTP zahtjevu. Inicijalizira se novi Retrofit objekt za komunikaciju s API-jem, dodaje se „slušatelj" odgovora na HTTP zahtjev te ukoliko je prijava uspješna (200) sprema se JWT token u SharedPreferences i otvara se nova aktivnost Home, u suprotnom se ispiše određena poruka.

Isječak programskog koda 3. Primjer funkcije za prijavu (Java)

```
 private void loginUser() {
    String email = Email.getText().toString();
    String password = Lozinka.getText().toString();
    DTOkorisnik loginRequest = new DTOkorisnik(email, password);

    OkHttpClient.Builder builder = new OkHttpClient.Builder();
    builder.addInterceptor(chain -> {
        Request request = chain.request().newBuilder()
                .addHeader("Content-Type", "application/json")
                .build();
        return chain.proceed(request);
    });

    Retrofit retrofit2 = new Retrofit.Builder()
            .baseUrl("http://193.198.57.183:7071/api/")
            .client(builder.build())
            .addConverterFactory(GsonConverterFactory.create())
            .build();

    ApiService apiInterface = retrofit2.create(ApiService.class);

    Call<LoginResponse> call = apiInterface.login(loginRequest);

    call.enqueue(new Callback<LoginResponse>() {
        @Override         public void onResponse(Call<LoginResponse> call,
Response<LoginResponse> response) {
            if (response.isSuccessful()) {
                Intent intent = new Intent(Login.this, Home.class);
                token = new Token(getApplicationContext());
                token.spremiToken(response.body().getToken());
                intent.putExtra("Object", loginRequest);
                startActivity(intent);

            } else {
                Toast.makeText(Login.this, "Invalid email or password",
Toast.LENGTH_SHORT).show();
                progressBar.setVisibility(View.INVISIBLE);
            }
        }

        @Override         public void onFailure(Call<LoginResponse> call, Throwable t) {
            Toast.makeText(Login.this, "Network error", Toast.LENGTH_SHORT).show();
            progressBar.setVisibility(View.INVISIBLE);
        }
    });
}
```

Isječak programskog koda 4.Definicija login metode u ApiService sučelju

```
public interface ApiService {
    @POST("Login")
    Call<LoginResponse> login(@Body DTOkorisnik dtokorisnik);
}
```

Isječak programskog koda 5.Kreiranje JWT tokena u c#

```
 1. [HttpPost]
 2.         public IActionResult Login([FromBody] DTOKorisnik userLogin)
 3.         {
 4.             var user = Authenticate(userLogin);
 5.
 6.             if (user != null)
 7.             {
 8.                 var token = Generate(user);
 9.                 return Ok(token);
10.             }
11.
12.             return NotFound("User not found");
13.         }
14.
15.         private string Generate(Korisnik user)
16.         {
17.             var securityKey = new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(_config["Jwt:Key"]));
18.             var credentials = new SigningCredentials(securityKey,
SecurityAlgorithms.HmacSha256);
19.
20.             var claims = new[]
21.             {
22.                 new Claim(ClaimTypes.NameIdentifier, user.Id.ToString()),
23.                 new Claim("email", user.Email),
24.                 new Claim(ClaimTypes.GivenName, user.Ime),
25.                 new Claim(ClaimTypes.Surname, user.Prezime)
26.
27.             };
28.
29.             var token = new JwtSecurityToken(_config["Jwt:Issuer"],
30.               _config["Jwt:Audience"],
31.               claims,
32.               expires: DateTime.Now.AddMinutes(15),
33.               signingCredentials: credentials);
34.             var tokenString = new JwtSecurityTokenHandler().WriteToken(token);
35.             var tokenJson = JsonConvert.SerializeObject(new { token = tokenString });
36.             return tokenJson;
37.         }
38.
39.         private Korisnik Authenticate(DTOKorisnik userLogin)
40.         {
41.             var korisnik = _KorisnikRepository.GetKorisnici().FirstOrDefault(o => o.Email ==
userLogin.Email);
42.
43.             if (korisnik != null)
44.             {
45.
46.                 string hashedPassword = HashPassword(userLogin.Lozinka);
47.                 if (hashedPassword == korisnik.Lozinka)
48.                 {
49.                     return korisnik;
50.                 }
51.             }
52.
53.             return null;
54.         }
```

Potrebno je još dodati JWT:Key te sljedeći kod u Startup klasi

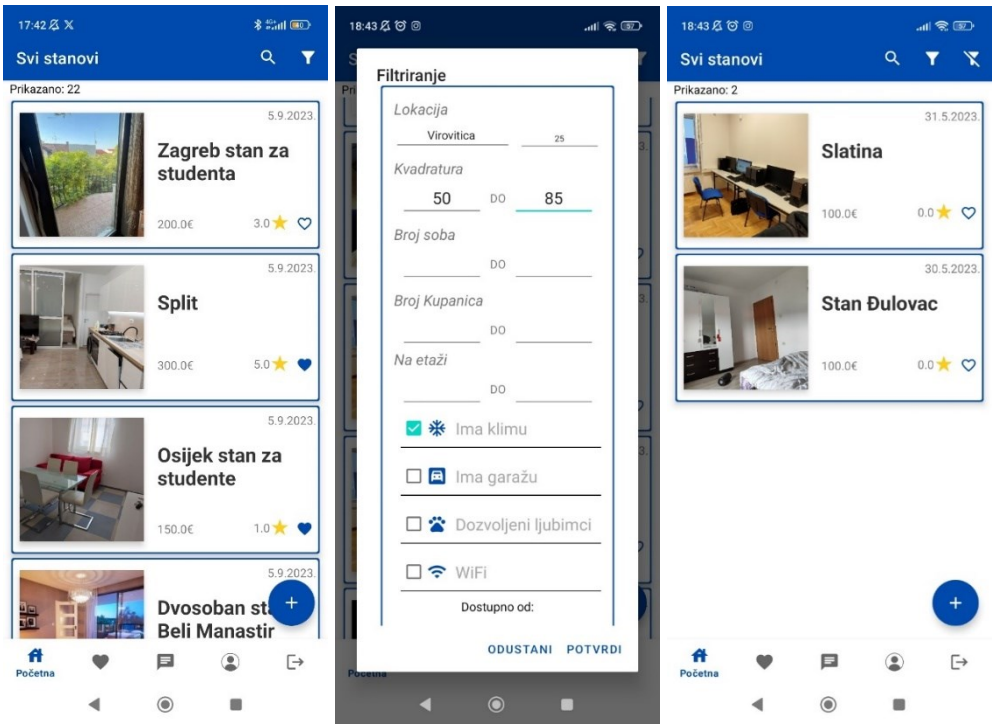
Isječak programskog koda 6.Konfiguracija autentikacije s JWT tokenom

```
services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)
                .AddJwtBearer(options => {
                    options.TokenValidationParameters = new TokenValidationParameters
                    {
                        ValidateIssuer = true,
                        ValidateAudience = true,
                        ValidateLifetime = true,
                        ValidateIssuerSigningKey = true,
                        ValidIssuer = Configuration["Jwt:Issuer"],
                        ValidAudience = Configuration["Jwt:Audience"],
                        IssuerSigningKey = new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(Configuration["Jwt:Key"]))
                    };
                });

```

5.2 Prikaz stanova

U ovom pregledu prikazani su svi dostupni stanovi koji su trenutno objavljeni. Korisnici imaju mogućnost jednostavne pretrage putem tražilice koja se nalazi u gornjoj navigacijskoj traci. Osim toga, mogu dodatno filtrirati rezultate putem gumba za filtriranje koji se također nalazi u toj traci. Klikom na taj gumb, otvara se prozor s poljima za unos filtera, omogućujući korisnicima da preciznije pronađu željeni stan filterima kao što su lokacija, kvadratura, cijena i slično.

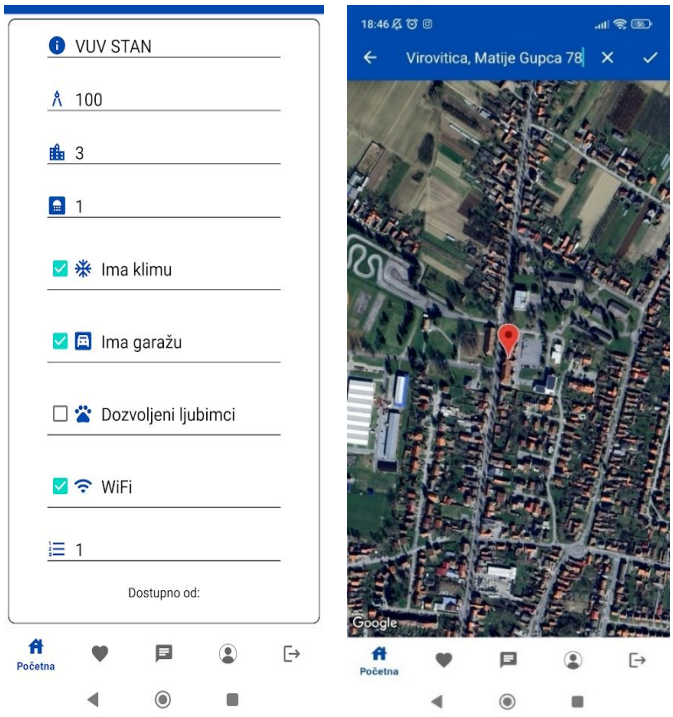


Slika 8. Prikaz i filtriranje stanova

Svaki stan na ovom pregledu također ima vidljivu ocjenu koja predstavlja aritmetičku sredinu svih ocjena koje su korisnici dodijelili tom stanu. Ovo omogućava potencijalnim najmoprimcima da brže procijene zadovoljstvo drugih korisnika koji su boravili u tom stanu. Korisnici također imaju opciju dodavanja stanova u svoj popis omiljenih stanova ili njihovo uklanjanje iz popisa pritiskom na gumb u obliku srca. Ova značajka olakšava korisnicima praćenje i upravljanje stanovima koji ih posebno zanimaju.

5.3 Dodavanje stanova

Na pritisku „plutajućeg gumba“ s desne strane, otvara se fragment za dodavanje stanova. U njemu se nalaze unosi za sve informacije stana (naslov, kvadratura, broj soba, broj kupaonica, potvrdni okviri za klimu, garažu, kućne ljubimce i WiFi, cijena) te Google mapa preko koje odabiremo lokaciju stana tako što upišemo adresu ili jednostavno nađemo i označimo „markerom“

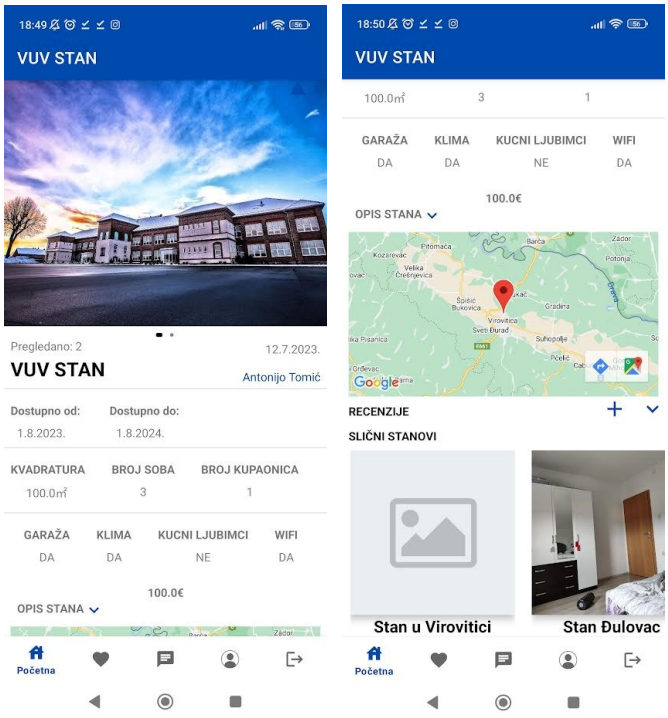


Slika 9. Unos informacija stana i postavljanje lokacije

Nakon unosa podataka imamo mogućnost dodavanja slika ukoliko želimo, ta opcija nije obavezna, a unos slika može biti višestruk.

5.4 Prikaz stana

Kada korisnik odabere određeni stan na platformi, otvara se nova stranica posvećena tom stanu. Na toj stranici korisnici mogu pronaći osnovne informacije poput kvadrature, lokacije i cijene stana. Također će vidjeti broj pregleda stana, što može sugerirati popularnost stana. Galerija slika omogućuje korisnicima uvid stana. Osim slika, stranica sadrži i opis stana s informacijama o broju soba, broju kupaonica i drugim relevantnim karakteristikama. Korisnici mogu pronaći profil stanodavca te ga dodatno kontaktirati u vezi stana. Datum objave stana pomaže korisnicima da prate dostupnost stana. Recenzije i komentari drugih korisnika pružaju dodatne uvide o iskustvima s tim stanom. Na kraju, prikaz sličnih stanova olakšava korisnicima pronalazak alternativnih opcija ako trenutni stan ne zadovoljava njihove potrebe.

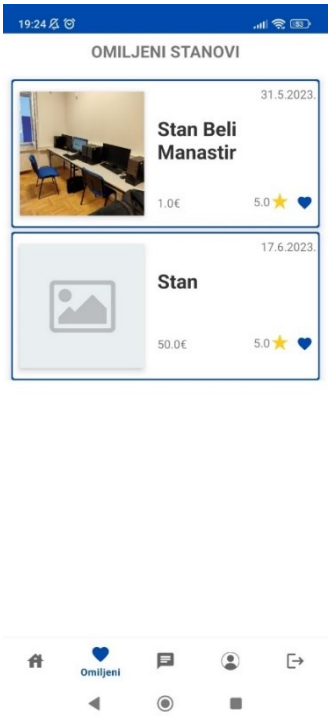


Slika 10. Detaljni prikaz stana

Stan je moguće prijaviti putem maila ukoliko korisnik smatra da je neprikladan ili recenzirati i ostaviti komentar.

5.5 Prikaz omiljenih stanova

Na odabir „Omiljeni“ u donjoj navigacijskoj traci otvara nam se stranica sa stanovima koje smo označili kao omiljene. Putem ove opcije korisnik može lakše doći do stanova koji su mu se svidjeli prilikom pretrage ili filtriranjem. Na ovoj stranici, korisnici mogu pregledati sve stanove koje su dodali u omiljene što olakšava usporedbu i donošenje konačne odluke oko odabira stana. Pritiskom na gumb u obliku srca, korisnik može jednostavno i brzo ukloniti stan iz omiljenih ukoliko je promijenio mišljenje.

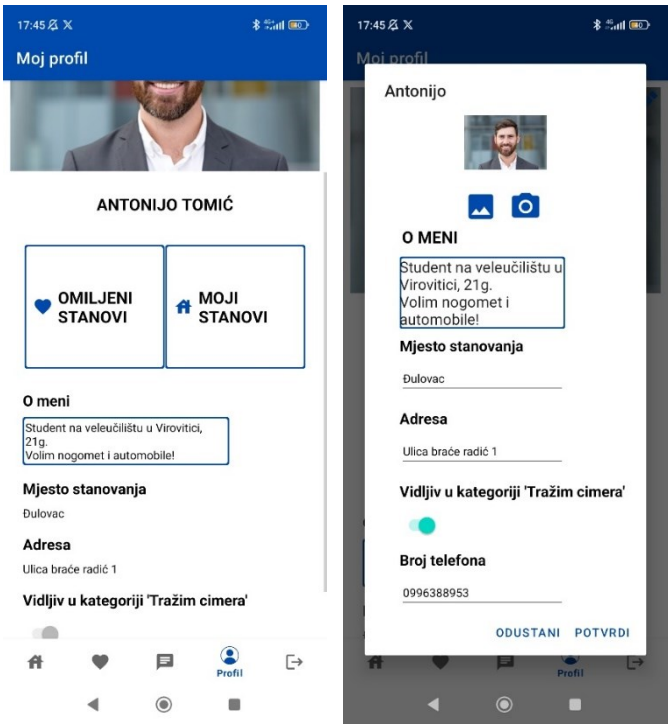


Slika 11. Prikaz omiljenih stanova

5.6 Prikaz profila

Na odabir „Profil“ u donjoj alatnoj traci otvara se stranica s prikazom detalja njegovog profila. Na ovoj stranici korisnik može pregledati svoje osobne informacije i ukoliko je potrebno, urediti ih kako bi prilagodili svoj profil najnovijim podacima o sebi.

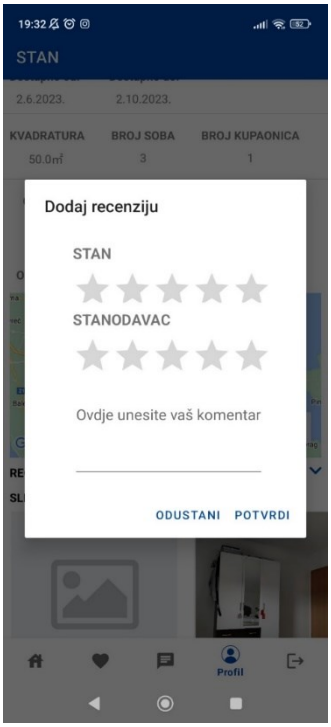
Pritiskom na „omiljeni stanovi“ odvest će nas na prethodno prikazanu funkcionalnost, dok pritiskom na „moji stanovi“ odvest će nas na prikaz stanova koje je korisnik objavio što olakšava mogućnost ažuriranja stanova ili brisanja.



Slika 12. Prikaz profila i uređivanje profila

5.7 Dodavanje recenzija i komentara stanu

Prilikom pritiska gumba za dodavanje recenzija otvara se „prozor“ za unos recenzija.

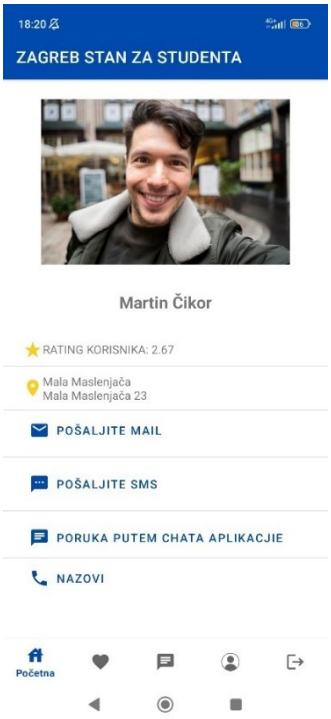


Slika 13. Recenziranje stana i stanodavca

Korisniku se pruža mogućnost ocjenjivanja stana i stanodavca ocjenjivanjem od 1 do 5. Osim ocjene, korisnik ima i neobaveznu mogućnost ostavljanja komentara, komentar može sadržavati dodatne informacije, dojmove ili iskustva koje želi podijeliti s ostalim korisnicima o stanu ili stanodavcu.

5.8 Prikaz profila vlasnika stana

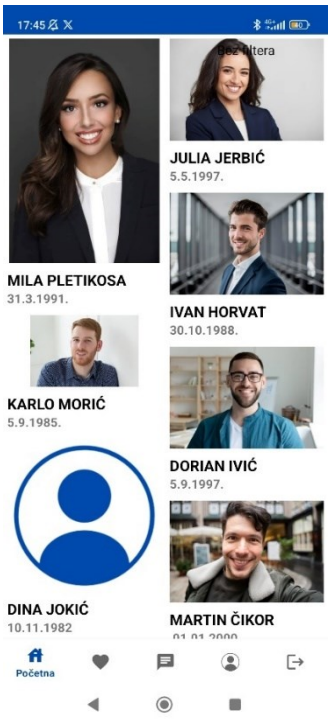
Prilikom pritiska imena vlasnika stana u detaljnom prikazu stana otvara se nova stranica s informacijama vlasnika, profilna slika, njegova prosječna ocjena koju dobije recenziranjem te kontakt opcije s mogućnošću kontaktiranjem putem Gmaila, SMS poruke, telefonom ili sustavom za razmjenjivanje poruka implementiranog u aplikaciji.



Slika 14. Prikaz profila stanodavca

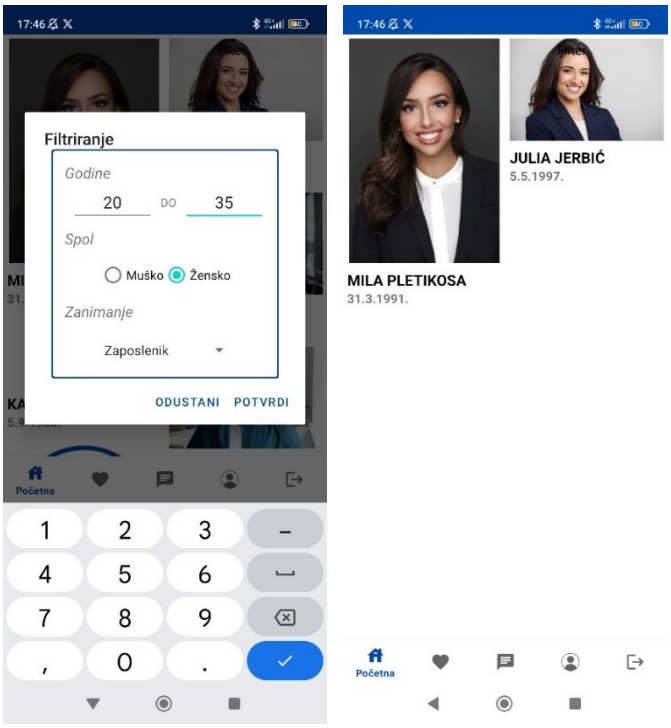
5.9 Prikaz dostupnih cimera i filtriranje

Odabirom prikaza dostupnih cimera otvara se stranica u kojoj se prikazuju dostupni cimeri u „Staggered gridu“ s najosnovnijim informacija poput imenom i prezimenom, profilnom slikom te datumom rođenja, ukoliko korisnik nema profilnu sliku postavlja se zadana slika.



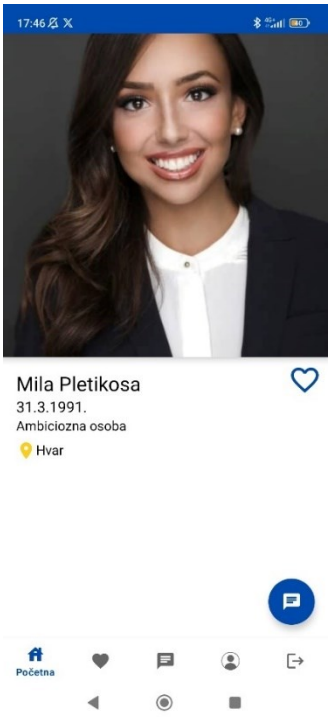
Slika 15. Prikaz dostupnih cimera

Korisnik ima mogućnost filtriranja korisnika kako bi suzio izbor cimera koji najbolje odgovaraju njegovim preferencijama. Pritiskom na gumb za filtriranje u gorem desnom kutu otvara se novi „prozor“ za unos filtera. Unosi filtera uključuju raspon godina, spola i zanimanja gdje korisnik može postaviti raspon godina te tako prikazao točno određenu dobnu skupinu koju želi npr. svojih godina ili svojeg spola i zanimanja. Nakon što korisnik unese određene filtere i primjeni ih, prikazat će mu se korisnici koji odgovaraju točno tim unesenim kriterijima što korisniku olakšava pronalazak potencijalnog cimera za zajedničko dijeljenje stana.



Slika 16. Prikaz i filtriranje korisnika

Prilikom odabira korisnika prikazuje se detaljniji prikaz profila, pritiskom na fotografiju otvara se prikaz slike u cijelom zaslonu, a pritiskom na opisu korisnika prikaže se cijeli opis. Postoji i mogućnost kontaktiranja putem aplikacije na pritisak plutajućeg gumba u donjem desnom kutu te dodavanje korisnika u omiljene pritiskom gumba u obliku srca.

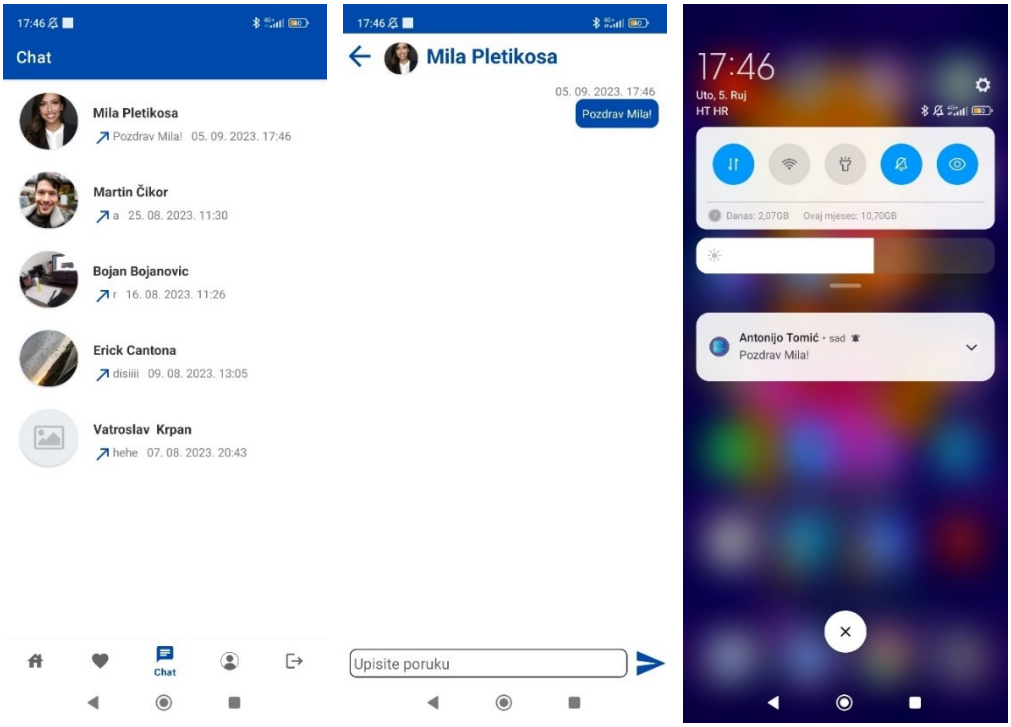


Slika 17. Prikaz korisnika

5.10 Funkcije slanja i primanja poruka

Pritiskom na „Chat“ u donjoj navigacijskoj traci otvara se stranica s korisnicima s kojima je korisnik ranije razmjenjivao poruke u aplikaciji, prikazuje se profilna slika korisnika, njegovo ime, zadnja poruka te vrijeme kada je poruka poslana ili primljena. Ukoliko pritisnemo korisnika prikazuju se sve poruke koje su međusobno poslane s tim korisnikom.

Kada korisnik pošalje poruku primatelju stigne plutajuća obavijest s imenom pošiljatelja i sadržajem poruke.



Slika 18. Prikaz chata i obavijesti

6 Zaključak

Izrada ovog rada proizašla je kombinacijom Java programskog jezika i REST API arhitekture. Java je objektno orijentirani programski jezik koji je iznimno popularan u Android programiranju, REST API omogućuje komunikaciju između mobilne aplikacije i udaljenih servera putem HTTP zahtjeva. Uz to aplikacija posjeduje i bazu podataka gdje se spremaju podaci generirani u aplikaciji, aplikacija također koristi i Firebase platformu u svrsi izvođenja funkcionalnosti slanja i primanja poruka u stvarnom vremenu te Firebase Cloud Messaging za slanje plutajućih obavijesti.

Svrha izrađene aplikacije je olakšati korisnicima proces pronalaska odgovarajućeg smještaja ili cimera putem niza funkcionalnosti. Funkcionalnosti aplikacije uključuju mogućnost filtriranja, pretraživanja, pregleda detalja, ocjenjivanja i komentiranja oglasa, te omogućuju komunikaciju između korisnika i stanodavaca ili potencijalnih cimera putem funkcije za slanje i primanje poruka unutar aplikacije. Osim toga, aplikacija omogućuje korisnicima da sami objave svoje oglase za stanove ili cimere, što je sveobuhvatno rješenje za potrebe pronalaska i ponude smještaja.

Aplikaciju očekuje dodatno ažuriranje i proširenje kako bi se poboljšala funkcionalnost i korisničko iskustvo, te eventualna objava na Google platformi, odnosno Trgovini Play, kako bi bila dostupna širem krugu ljudi.

7 Literatura

[1] Meet Android Studio, Pristupljeno: 9. kolovoza 2023. [Online]. Dostupno na: https://developer.android.com/studio/intro

[2] Android programming for Beginners (John Horton), Pristupljeno: 9. kolovoza 2023., [Knjiga]

[3] The Activity Lifecycle, Pristupljeno: 10. kolovoza 2023. [Online]. Dostupno na: https://developer.android.com/guide/components/activities/activity-lifecycle

[4] Java: ABeginner's Guide (Herbert Schildt), Pristupljeno: 11. kolovoza 2023., [Knjiga]

[5] Entity Framework Core, Pristupljeno 12.Kolovoza 2023. [Online]. Dostupno na: https://learn.microsoft.com/en-us/ef/core/

[6] Pro Entity Framework Core (Adam Freeman), Pristupljeno 14.Kolovoza 2023. [Knjiga].

[7] What is Firebase?, Pristupljeno 17. kolovoza 2023. [Online]. Dostupno na: https://firebase.google.com/

[8] Mastering Firebase for Android Development (Ashok Kumar), Pristupljeno 18.Kolovoza 2023. [Knjiga].

[9] Firebase Cloud Messaging: Pristupljeno 20, kolovoza 2023. [Online]. Dostupno na: https://firebase.google.com/docs/cloud-messaging

[10] Arhitektura Androida, Pristupljeno 14.Kolovoza 2023, Dostupno na: https://web.math.pmf.unizg.hr/~karaga/android/android_skripta.pdf

[11] Životni Ciklus Androida, Pristupljeno 14.Kolovoza 2023, Dostupno na: https://www.webprogramiranje.org/aktivnost-u-okviru-android-operativnog-sistema/

[12] Client-Server model, Pristupljeno 17.Kolovoza 2023., Dostupno na: https://en.wikipedia.org/wiki/Client%E2%80%93server_model

Popis slika

Veleučilište u Virovitici

OBRAZAC 5

IZJAVA O AUTORSTVU

Ja, Antonijo Tomić

izjavljujem da sam autor/ica završnog/diplomskog rada pod nazivom

"Java programski jezik i REST API za razvoj aplikacija za Android operacijski sustav"

Svojim vlastoručnim potpisom jamčim sljedeće:

- da je predani završni/diplomski rad isključivo rezultat mog vlastitog rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu, a što pokazuju korištene bilješke i bibliografija,
- da su radovi i mišljenja drugih autora/ica, koje sam u svom radu koristio/la, jasno navedeni i označeni u tekstu te u popisu literature,
- da sam u radu poštivao/la pravila znanstvenog i akademskog rada.

Potpis studenta/ice

Antonijo Tomić

Veleučilište u Virovitici

OBRAZAC 6

ODOBRENJE ZA OBJAVLJIVANJE ZAVRŠNOG/DIPLOMSKOG RADA U DIGITALNOM REPOZITORIJU

Ja Antonijo Tomić

dajem odobrenje za objavljivanje mog autorskog završnog/diplomskog rada u javno dostupnom digitalnom repozitoriju Veleučilišta u Virovitici sadržanom u Dabar (Digitalni akademski arhivi i repozitoriji) te u javnoj internetskoj bazi završnih radova Nacionalne i sveučilišne knjižnice bez vremenskog ograničenja i novčane nadoknade, a u skladu s odredbama članka 58. stavka 5., odnosno članka 59. stavka 4. Zakona o visokom obrazovanju i znanstvenoj djelatnosti (NN 119/22).
Potvrđujem da je za pohranu dostavljena završna verzija obranjenog i dovršenog završnog/diplomskog rada. Ovom izjavom, kao autor navedenog rada dajem odobrenje i da se moj rad, bez naknade, trajno javno objavi i besplatno učini dostupnim na sljedeći način:

a) Rad u otvorenom pristupu
b) Rad dostupan nakon: 18.9.2023. (upisati datum nakon kojeg želite da rad bude dostupan)
c) Pristup svim korisnicima iz sustava znanosti i visokog obrazovanja RH
d) Pristup korisnicima matične ustanove
e) Rad nije dostupan (*u slučaju potrebe dodatnog ograničavanja pristupa Vašem završnom/diplomskom radu, podnosi se pisani obrazloženi zahtjev*).

Potpis studenta/ice
Antonijo Tomić

U Virovitici, 8.9.2023.