DOCUMENTAȚIA PROIECTULUI

**My Property**

STUDENȚI

Andrei-Alexandru Samulescu

Alexandru Edves

**Cuprins**

½ - 1 pagina despre prezentarea proiectului, ce isi propune, ce probleme rezolva, solutii

½ - 1 pagina despre tehnologiile folosite

½ - 1 pagina despre partea de “backend” - request-uri folosite

1-3 pagini despre arhitectura (componentele principale, navigare, baza de date etc.)

1-2 pagini despre cum poate fi utilizata aplicatia: tipuri de utilizatori, ce vede fiecare, autentificare etc

½ - 1 pagina concluzii

1.Prezentare generala

Este o aplicatie care ofera posibilitatea unui client sa vizualizeze proprietatile stocate in baza de date, pe care ulterior le poate inchiria daca acestea sunt disponibile.A doua functionalitate este posibilitatea de a crea un cont de proprietar, functie care permite listarea unor proprietati spre inchiriere.

Aplicatia noastra faciliteaza comunicarea intre chiriasi si proprietari.De fiecare data cand un proprietar adauga o propritate noua pentru a fi inchiriata, aceasta devine instant vizibila pentru toti chiriasii.

Cu ajutorul interfetei prietenoase, clientul poate sa vizualizeze deataliile complete ale fiecarei proprietati, inclusiv descireri, proprietar, adresa si daca este ocupata sau nu.

Indiferent daca cautati o proprietate pentru o vacanta, un sejur de afaceri sau o perioada de tranzit, aplicatia noastra de inchiriere a proprietatilor va ofera o modalitate convenabila de a gasi locuinta perfecta.

2.Tehnologii folosite

* **Android studio**
* SQL lite
* Java
* .Net & C#

Aplicatia este dezvoltata in Android Studio, iar limbajul folosit este Java.

Folosim o baza de date implementata in SqlLite, in care stocam toate informatiile despre fiecare user care poate sa fie client sau proprietar si de asemenea stocam si informatiile despre proprietati.

Proiectul contine si un API cu operatii CRUD(Create Read Update Delete) pe fiecare tabel din baza de date.Acesta este implementat in .Net in limbajul C#.

3.Backend

Am proiectat un backend functional pentru toate elementele specifice din baza noastra de date locala, urmand a fi implementata si o conexiune intre backend si frontend.Pentru backend s-a folosit ASP.NET, Web Api, un framework care ne ajuta in a “simula” un server.Pentru arhitectura backend-ului s-au folosit modele cu scopul de a primi si transmite date(ex: User, UserDTO) .S-au folosit controllere pentru a putea distribui in mod uniform toate actiunile.

Exemple de http request-uri folosite:

* HttpGet-returneaza datele in functie de ce avem nevoie(client, proprietar, proprietati)
* HttpPost-adauga o inregistrare noua in baza de date
* HttpPut-modifica o inregistrare existenta din baza de date
* HttpDelete-sterge o inregistrare existenta din baza de date

4.Arhitectura

Pentru a putea organiza eficient toate datele, ne-am hotarat sa folosim modele pentru a putea transmite datele si de a le manipula mai usor. Urmatoarele modele sunt: User, Client, Owner, Property.

User: are scopul de a accesa datele din aplicatie intr-un mod mai sigur si mai eficient. Acesta este alcatui din

* ID : (int) AutoIncrement, pentru a putea avea pentru fiecare utilizator un id unic.
* Username: (string), pentru a putea securiza prin intermediul unei chei.
* Password: (string), pentru a putea securiza prin intermediul unei chei secrete
* IsOwner: (bool), pentru a putea pe viitor sa separam Ownerul de Client.

Owner: are specializarea de a putea crea noi Proprietati. Acesta contine:

* ID: care are acelasi scop ca User-ul.
* Name: Prenumele Owner-ului.
* Surname: Numele Owner-ului.

Client: are specializare de a putea observa intreaga tabela a Proprietatilor. Tabela contine:

* ID: care are acelasi scop ca User-ul.
* Name: Prenumele Owner-ului.
* Surname: Numele Owner-ului.

Property: este un model care poate sa fie vizualizata atat de catre un client cat si de un proprietar. Aceasta tabela contine:

* Name : Nume resedintei.
* OwnerName: Numele proprietarului.
* Description: Descrierea proprietatii.
* Status: Statusul proprietatii.
* Price: Pretul locuintei.
* Address: Adresa stradala
* Type: Tipul acesteia (Casa / Apartament)

Aplicatia ruleaza pe 3 mari componente: MainActivity, ClientActivity si OwnerActivity.

MainActivity are rolul de register in aplicatie. Aici se regasesc urmatoarele: Fragmentul pentru Login, Signup, newAccount.

* Signup se ocupa pe partea de creare al utilizatorului.
* Login se ocupa pe partea de cautare al utilizatorului. Aceasta cautare o facem prin intermediul unui find in baza de date cu un username si parola (atat username-ul cat si parola sunt construite astfel incat sa nu avem conflicte).
* NewAccount se ocupa pe separarea utilizatorului de rolul acestuia. In cazul in care utilizatorul este nou pe aplicatie, acesta poate sa isi faca un cont specializat pe domeniul ales (Client / Owner).

In plus, am folosit un fragment special care are rolul de a sterge toate informatiile din baza de data, cu scopul de a putea verifica posibilele conflicte in cazul stergerilor.

ClientActivity se ocupa pe specializarea utilizatorilor clienti. Acestia sunt lipsiti de orice putere, avand unica posibiliate de a interactiona cu baza de date prin intermediul recycle-view-ului care isi ia datele din tabela Property. Este alcatuita din 2 fragmente: MainPage si PropertyDetail.

OwnerActivity se ocupa pe specializarea proprietarului. Acestia pot sa isi creeze proprietati ca pe urma sa fie vizualizate de catre posibilii clienti. Este alcatuita din 3 fragmente: MainPage, PropertyDetail, AddProperty.

Pentru a putea naviga usor prin fragmente, am folosit Navigation Component. Un mod usor de a naviga, prin intermediul unor eventuri. Am folosit in toate cele 3 activitati.

Pentru a putea stoca datele eficient, am folosit SQLite, o baza de date locala care poate sa retina toate activitatile din aplicatie.

5.Utilizare

5.1 Tipuri de utilizatori

* Client-are posibilitatea de a inchiria o locuinta
* Proprietar-are posibilitatea de a adauga o locuinta spre inchiriere

5.2 Perspective utilizatori

* Client-dupa ce se autentifica, acesta poate vizualiza toate locuintele care sunt adaugate in baza de date si pot sa aleaga sa inchirieze o locuinta daca aceasta nu este ocupata.
* Proprietar-dupa ce se autentifica, isi poate vizualiza proprietatile pe care le detine si poate sa adauge altele noi daca doreste.

5.3 Autentificare

* SignUp

Aceasta optiune este apelata de un utilizator care nu are cont momentan in aplicatie si doreste sa isi creeze unul.Acesta trebuie sa isi aleaga rolul, client sau proprietar, apoi sa isi aleaga un username si o parola cu care se va autentifica ulterior.

* LogIn

In aceata faza a autentificarii, utilizatorul se foloseste de credentialele alese in faza de “sign up” pentru a-si accesa contul.Daca credentialele se potrivesc, apoi daca este prima autentificare a utilizatorului, acesta trebuie sa isi introduca numele si prenumele.

6.Concluzii

In concluzie, aplicatia noastra de inchiriere a proprietatilor ofera o solutie practica si conveenabila pentru cei care cauta locuinte temporare.Cu ajutorul acestei aplicatii, utiilizatorii pot gasi cu usurinta proprietati disponibile pentru inchiriere, avand acces l ainformatii detaliate.

Aplicatia noastra isi propune sa aduca impreuna chiriasii si proprietarii intr-un mod transparent si eficient.Indiferent ca sunteti in cautarea unei locuinte pentru o scurta sedere sau pentru o perioada mai lunga, aplicatia noastra va pune la dispozitie o gama variata de optiuni, astfel incat sa puteti gasi proprietatea perfecta care sa va satisfaca nevoile si preferintele.