**Proiect Android:**

**MyTinder**

Proiect realizat de:

**Gorea Tudor-Andrei**

**Cuprins:**

1. **Introducere și obiective**................................................................
2. **Parte teoretică**…………………………………………………………………………..
   1. *Client*………………………………………………………………………………..
   2. *Server*……………………………………………………………………………….
3. **Proiectare și implementare**……………………………………………………….
   1. *Client*………………………………………………………………………………..
   2. *Server*……………………………………………………………………...……….
   3. *API*........................................................................................
4. **Bibliografie**………………………………………………………………………………...

**Introducere**

Da

**Parte Teoretică**

**Client:**

aaa

**Server:**

aaaa

**Proiectare și Implementare**

**Client:**

aaaa

**Server:**

aaaa

**API:**

Comunicarea dintre client și server este foarte importantă în această aplicație.

În comenzile de mai jos se pot observa **c!** și **s!**, care reprezintă cine a transmis mesajul (c=client, s=server).

Caracterul **#** este folosit pentru a despărții comenzile de parametrii lor și de spațiul liber rămas din socket.

Am marcat cu culori diferite tipurile de dată în care trebuie să fie datele trimise împreună cu comanda astfel:

**comandă**

**șir de caractere (string)**

**valoare numerică (1, 2, 3, .....)**

**tip de dată enum (nope / male / female / other)**

**valoare booleană (true / false)**

API-urile folosite in proiect sunt:

**c!test\_add#** - verifică dacă se pot efectua operații de introducere în baza de date pe un tabel de teste (*test\_table*).

**c!test\_get#** - verifică dacă se pot efectua query-uri pe baza de date pe un tabel de test (*test\_table*).

**c!login\_req#*mail*#*pwd*#** - clientul trimite o cerere de login, alături de mailul și parola (parola sub formă de hash) către server, așteptând unul din răspunsurile:

**s!login\_ok#*mail,pwd,fullname,age,gender*#** - serverul trimite răspuns că totul a decurs în regulă (contul există și parola este bună), alături de toate datele utilizatorului,

**s!login\_err#** - serverul trimite mesaj de eroare dacă contul nu există sau parola nu este bună, sau alte cauze, fară a specifica eroarea.

**c!reg\_req#*mail,pwd,fullname,age,gender*#** - clientul trimite o cerere de înregistrare cont, împreună cu datele de înregistrare (clientul trebuie să se asigure că parola este hash-uită înainte de a fi trimisă), așteptând unul din răspunsurile:

**s!reg\_ok#** - dacă contul a fost adăugat cu succes în baza de date,

**s!reg\_err#** - dacă există deja un cont cu același mail sau apar erori la comunicarea cu baza de date.

**c!getP\_req#*ageMin*#*ageMax*#*sMale*#*sFemale*#*sOther*#** - clientul trimite o cerere că dorește să primească o listă de persoane, care au vârsta între *ageMin* și *ageMax*, iar valorile booleene reprezintă ce genuri să apară în rezultatul căutării.

(ex: sMale=true, sFemale=true, sOther=false - serverul va returna doar persoane cu genul *female* și *nope*(ascuns),

sMale=true, sFemale=false, sOther=true - serverul va returna doar persoanele cu genul *male*, *other*(non-binar, sau orice altceva) și *nope*)

Serverul va trimite unul din răspunsurile:

**s!getP\_inv#** - dacă *ageMax* < *ageMin* sau toate cele 3 valori booleene sunt false,

**s!getP\_err#** - dacă există erori in la baza de date

**s!getP\_ok#*mail1,fullname1,age1,gender1*$*mail2,fullname2,age2,gender2*$...#** – serverul va trimite un mesaj că totul a mers bine, alături de o listă cu un număr maxim de persoane (ex: primele 20 de persoane) care respectă filtrele date. Persoanele din listă vor fi despărțite de caracterul **$**.

**c!like\_set#*mail1*#*mail2*#** - clientul trimite un request la server să salveze în baza de date că user-ul cu mail=*mail1* i-a dat like la user-ul cu mail=*mail2*. Clientul nu așteaptă niciun răspuns de la server în legătură cu această operație.

**Bibliografie:**