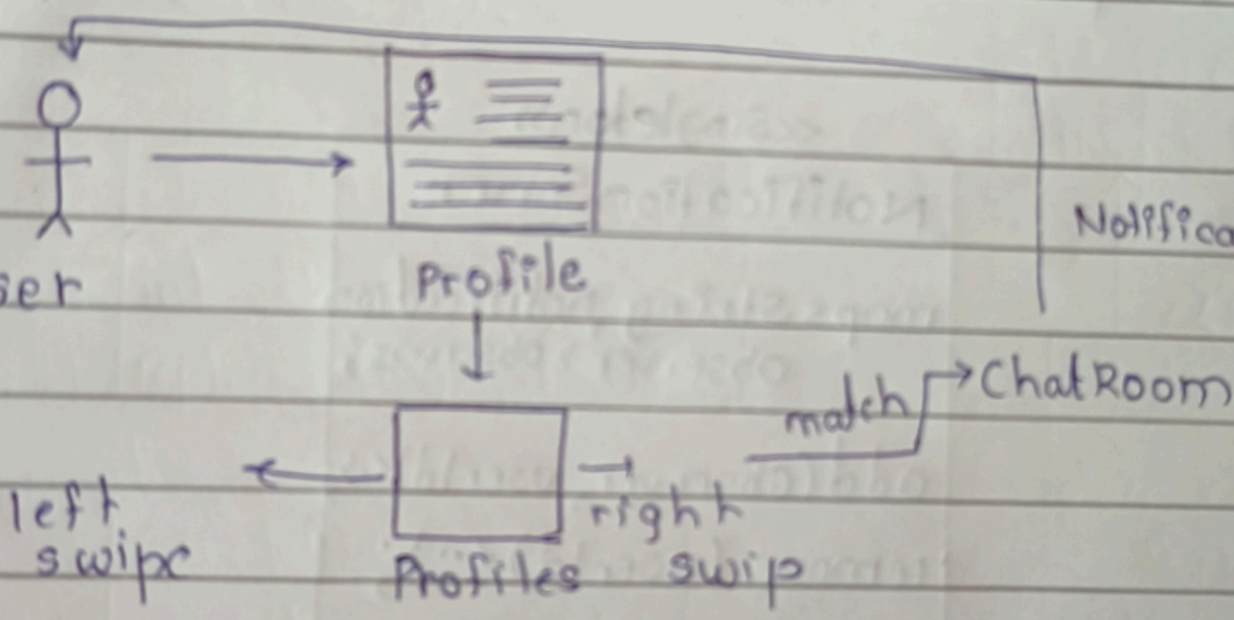
Lecture 27 : Building Tinder

Requirement Gathering and Analysis

→ ① User can swipe left/right to a profile
→ ② User can setup his/her own profile
→ ③ User can set his/her preferences
→ ④ Once there is a match they can chat in a chatroom.
→ ⑤ User can see all the profiles ~~one~~ near their location (nearby can be based on different strategy).
→ ⑥ User should get notified where there is a match or receive a new message.
→ ⑦ User matching should be based on several factors & scores (like interest match, location match, etc).

Happy Flow



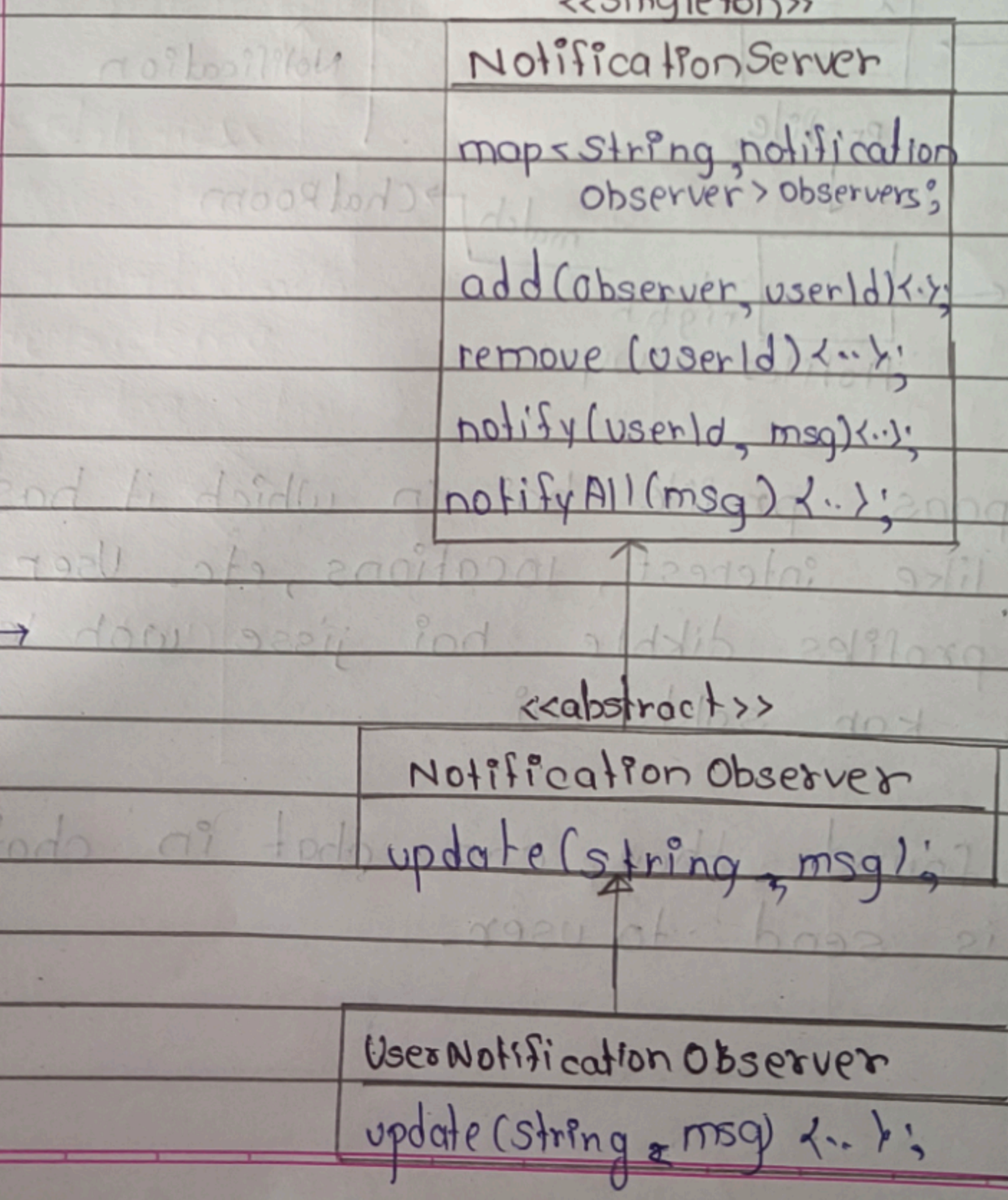
→ User ke paas profile hai in which it has his information like interest, locations, etc. User ko baaot saare aur profiles dikhte hai jisse woh left or right swipe kar sakte hai.

→ If match found they can chat in chatroom & notification is send to user.

→ Abhi tak hum bottom up Approach use kar rahe the but ab hum Top down Approach use karenge coz in this application we have to make many small objects like user, profile, preferences, match, interest, etc. Agar humne pehle inhe banaya hai toh humme inhe combine krne mein dikkat aayegi.
∴ Top-down approach but its upto you tumhe kya use karna hai

UML Diagram details

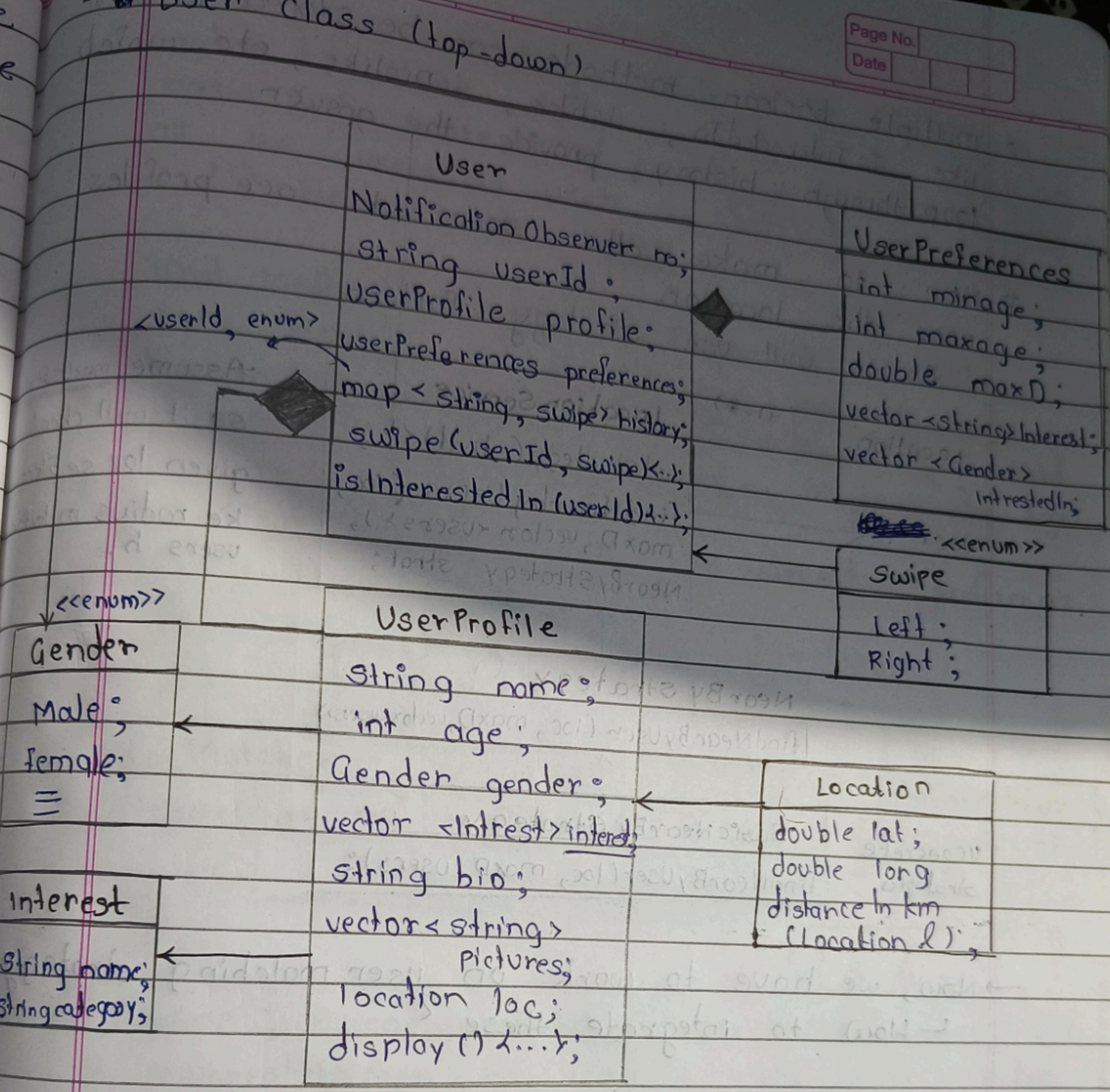
→ Hum sbse pehle notification ~~system~~ server banayge to handle notification. Hum notifications ke liye observer pattern use karege.



47] Verb
48] Preposition
49] Noun

27] Identify Patterns

User Class (top-down)



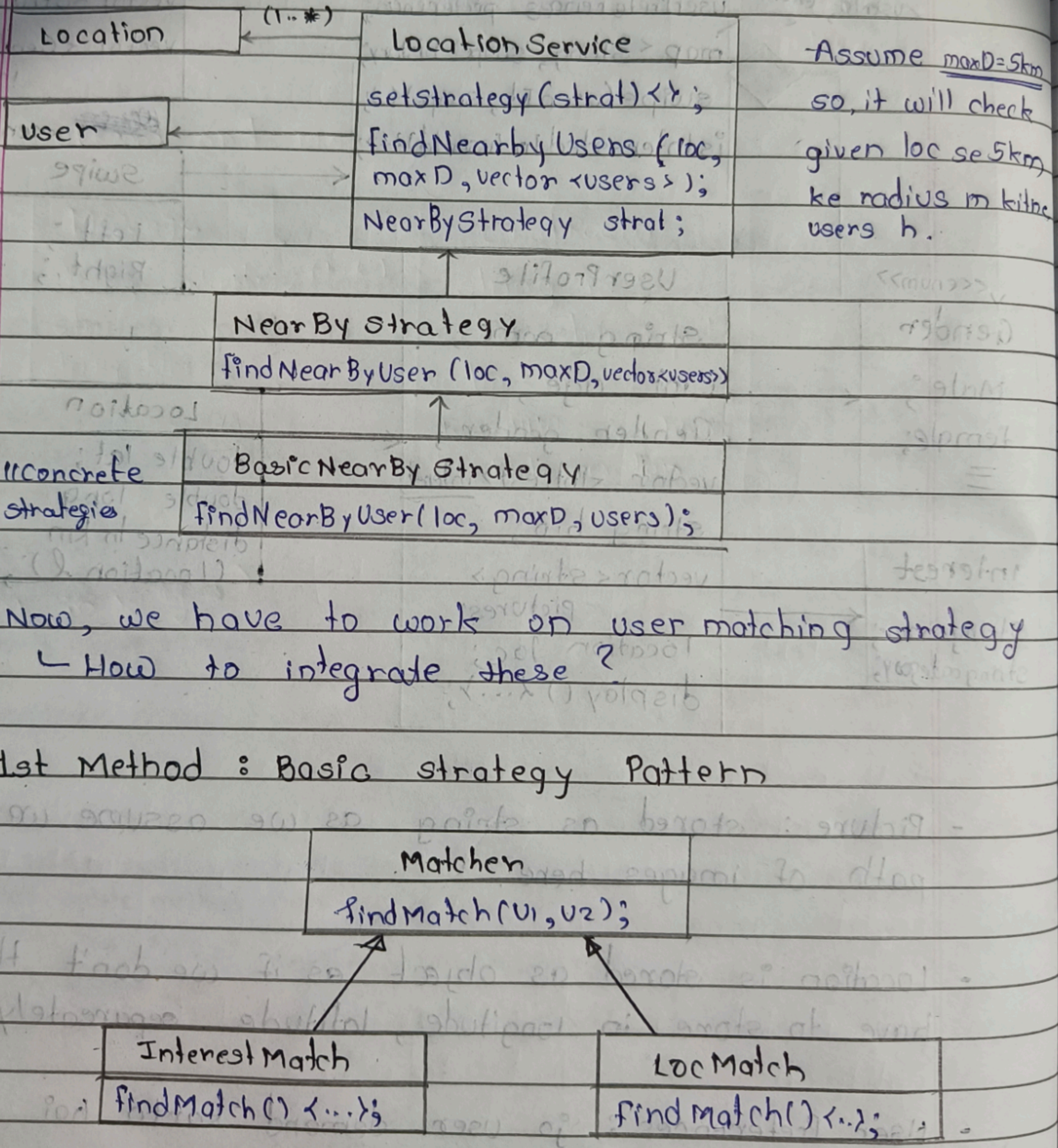
- Picture : stored as string as we assume we store path of images here.

- Location is stored as object as if we don't that we have to store in longitude, latitude separately

- User Preferences : jo user demand karta hai dusre user se.

- multiple boolean methods can be stored in user like isInterestedIn, isLike, isDislike, etc. which loop through history, provide the answer

Now, to make other users to see profiles near the location
↳ we will use strategy Pattern



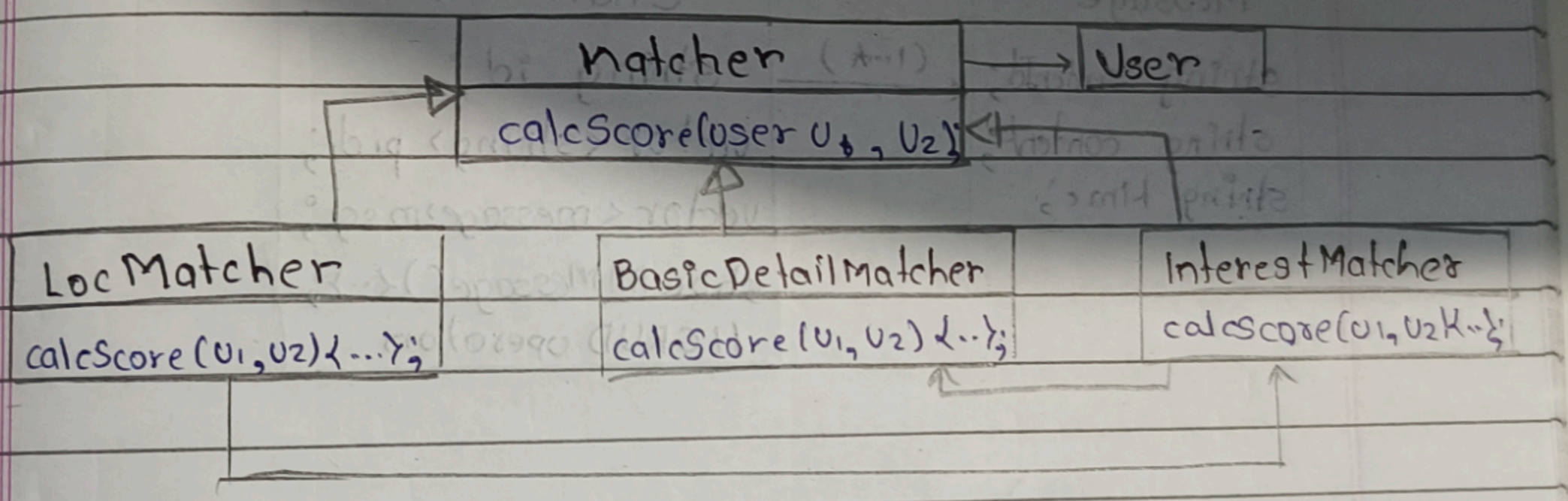
→ Now, we have to work on user matching strategy
↳ How to integrate these?

1st Method : Basic strategy Pattern

- Here, hum matching strategy ke base par particular match kar rahe but we have to match on basis of score and provide score found from all the users to user & then it will select.
(L) We will use COR

IInd Method : Chain of Responsibility Pattern
same strategy use krege par concrete classes ke har match mein dusro ko reference dedenge → Hard reference ⇒ tightly couple denge ⇒ same flow hoga

Ex: first location match ⟶ interest → preferences

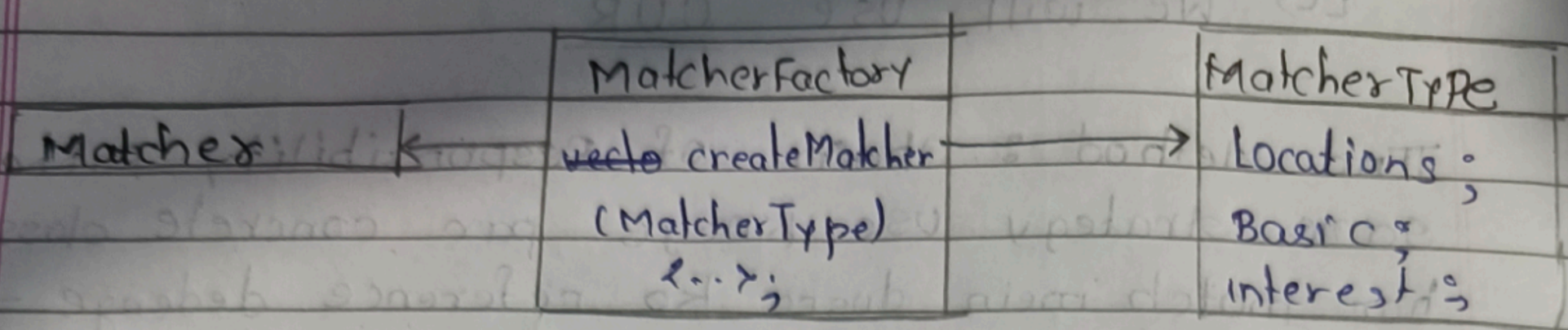


- Assume ; Humne locMatcher ka object banaya → to vo InterestMatcher ko kahega uska score nikalne ke liye ⟶ phir InterestMatch basic detail matcher ka bolega ki apna score nikal ke laa...

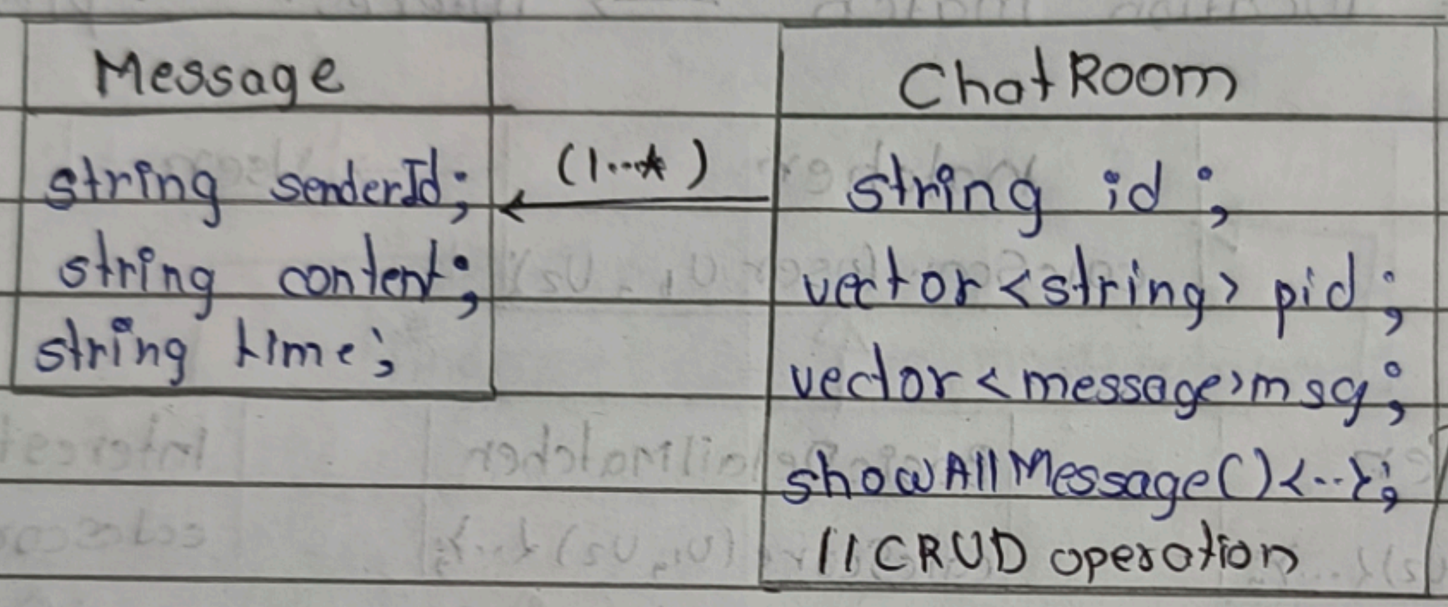
BasicDetailMatcher ⟶ 50 → InterestMatcher → 60 →
LocMatcher ⟶ 70

Then it will return to client.

- We will create Factory which will give object of any of these : matchers based on enum.



- ChatRoom and Messages



(full UML in images)