Problem:

- Designing a recommendation engine that will recommend a 21 day workout plan based on a user's attributes like age, physical condition, gender etc.

Approach:

- Right now there is no dataset that the engine can work with; it is a cold start problem. Therefore it is important to know the item (routine) very well.
- What we plan to do is make a Routine entity with different attributes like goal it helps achieve, age it is suitable for, gender, activity levels, bmi, bmr.

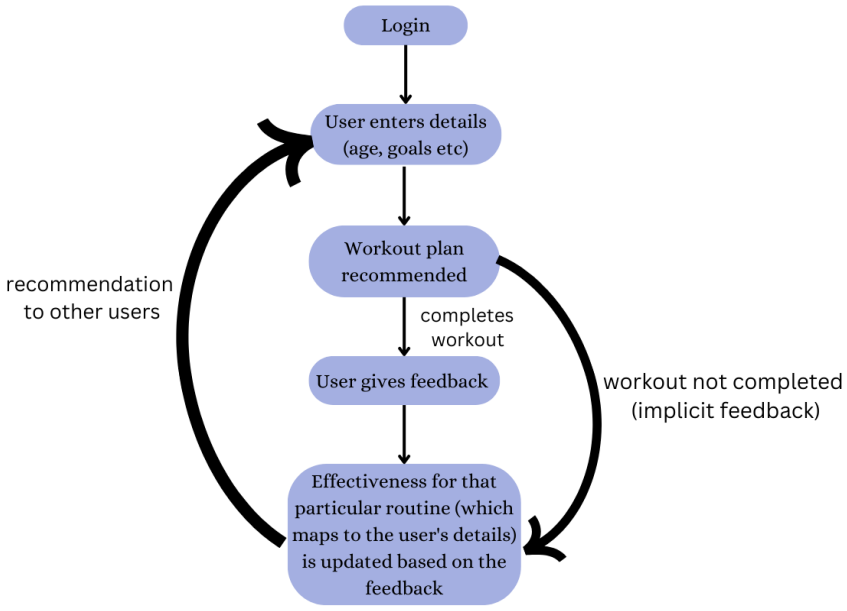
ROUTINE

Name	Goal	Age	Gender	Activity Levels	BMI	Effectiveness
R1	Weight loss	x	M	Moderate	….	
R2	Strength training	y	F	Sedentary	….	
R3	General Fitness	z	F	Intense	….	

.
.
.

Data collection:

- The data will be collected explicitly: Users will give a satisfaction rating after completing the 21 day workout. Also implicit data can be collected by looking at whether a user completes the 21 day workout or not. Not completing might indicate that the routine was too difficult, or not interesting enough etc.



Model:
We have these factors:

- ➔ Age groups are divided into 3 - Young, Middle aged and Old.
- ➔ Goals are also of 3 kinds - Losing Weight (Mostly aerobic exercises), Strength training (Mostly anaerobic exercises) and General fitness.
- ➔ Previous activity level 3 kinds - Moderate, Sedentary, Intense.
- ➔ BMI 4 kinds- Underweight, Healthy, Overweight, Obese.
- ➔ 2 Genders - Male and Female.

For each combination of these (216 ways) we have to have a routine.

However, the table cannot be filled in a random way because exercises that comprise the routine are dependent on the goal the user wants to achieve. Cannot make guinea pigs out of users. This is the problem we are currently facing.

One solution is that we research and fill all 216 entries for Routine manually (or get the help of fitness coaches and experts to do that).

Then the system can learn online and take every user feedback into account. Something here to keep in mind will be how fast the system adapts to changing data. If the learning rate is high, then the system will adapt rapidly to new data, but also tend to quickly forget the old data. Conversely if the learning rate is low the system will learn more slowly but will also be less sensitive to noise in the new data.