|  |  |
| --- | --- |
|  | FitTracker Ontwerp Document |
|  |  |
|  | Wenders,Sem S.W.C.  DB02U |

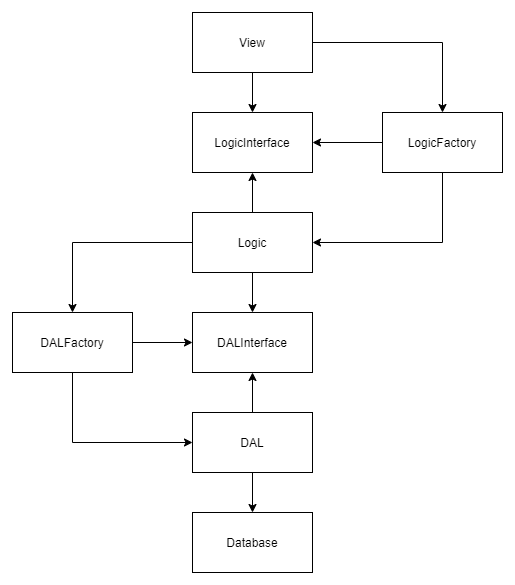
Contents

[Architectuur 2](#_Toc54599944)

[Klassendiagram 3](#_Toc54599945)

[Databaseontwerp 4](#_Toc54599946)

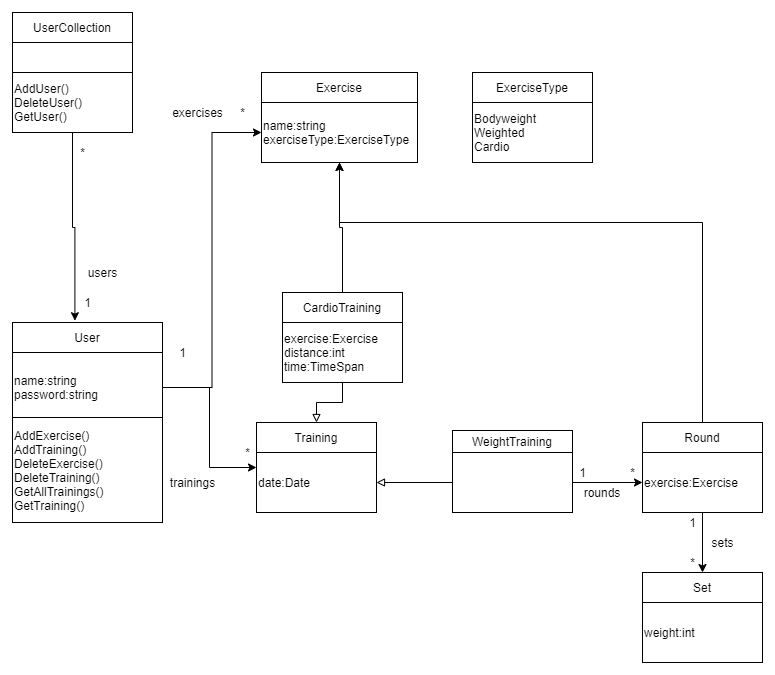
# Architectuur



In de database worden alle gebruikers en trainingen opgeslagen. Deze database kan worden bereikt met de Data Access Layer. In mijn project zal ik voor de view gebruik maken van een MVC applicatie. Omdat de Logic, View en DAL lagen allemaal onafhankelijk van elkaar zijn is de applicatie makkelijk te onderhouden. Tussen elke laag zit een factory pattern. De interfaces functioneren als een contract tussen de lagen. Ze spreken af welke methodes er zijn, maar voor de rest weten de lagen niks van elkaar. De factories maken objecten aan zodat een laag de methodes kan gebruiken.

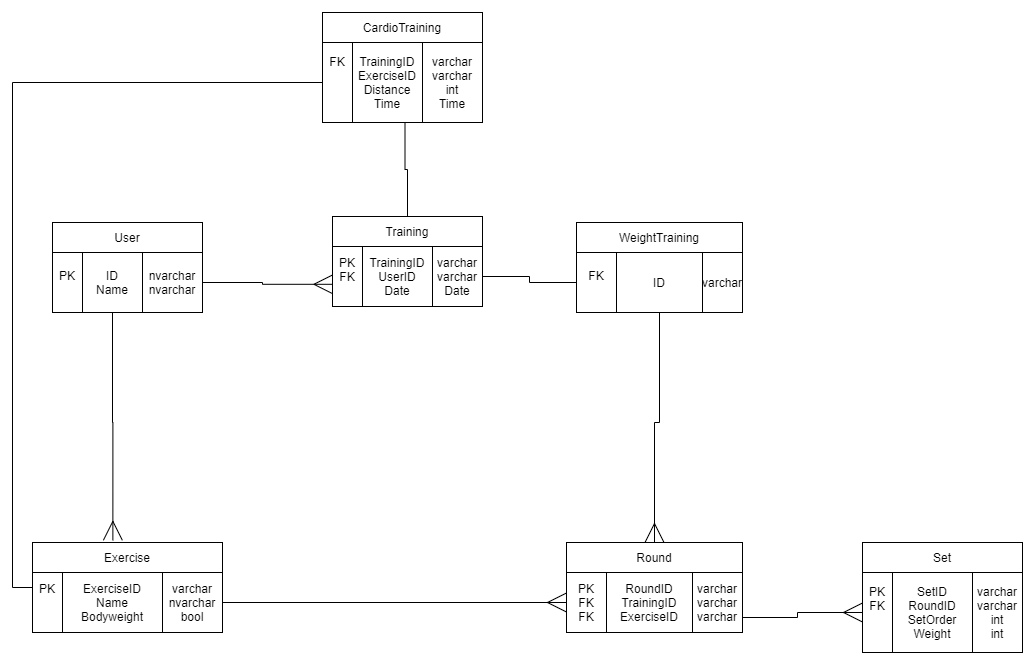
Als ik de applicatie zou uitbreiden, zou ik een API maken die de methodes van de logic laag oproept. Dit wordt heel makkelijk gemaakt door de factory pattern.

# Klassendiagram



Elke gebruiker heeft een ID en naam. Hij/zij kan een oefening verwijderen en toevoegen, een training verwijderen en toevoegen en zijn of haar trainingen opvragen. Elke oefening heeft een naam en een enum die beschrijft wat voor een type oefening het is. Er zijn op het moment drie typen oefeningen; cardio, gewichten, lichaamsgewicht. De cardio oefeningen kunnen voor de cardiotrainingen worden gebruikt en de gewichten en lichaamsgewicht oefeningen kunnen voor de krachttraining worden gebruikt. Een cardiotraining heeft een oefening, afstand en tijdsduur. Een krachttraining heeft een lijst met rondes. Elke ronde heeft een oefening en een lijst met sets. Elke set heeft een gewicht. Mocht ik later een nieuwe oefening of training willen toevoegen, dan is dat redelijk makkelijk te doen.

# Databaseontwerp



Elke gebruiker heeft een aparte ID. Dit is een primary key en hiermee kan dus onderscheid worden gemaakt tussen verschillende gebruikers. Elke training heeft een eigen ID en een UserID. Het UserID kan gebruikt worden om de juiste gebruiker te vinden die de training heeft uitgevoerd. Ik heb op internet opgezet op wat voor een manieren je inheritance in een database kunt implementeren en uiteindelijk heb ik voor de *Table-Per-Type* manier gekozen. Dit heb ik gedaan omdat ik een tabel wil hebben met alle trainingen, zodat ik op de homepagina makkelijk alle trainingen van een gebruiker kan weergeven. Een cardiotraining heeft een ID, ExerciseID, afstand en tijdsduur. Een gewichtentraining heeft een foreign key ID van de training tabel. De tabel Round heeft dezelfde foreign key, een ExerciseID en een eigen ID. De tabel Set heeft een primary key SetID, foreign key RoundID, een SetOrder om bij te houden de hoeveelste set het is en een integer Weight om het gewicht van de set op te slaan.