**Design**

**Entities**

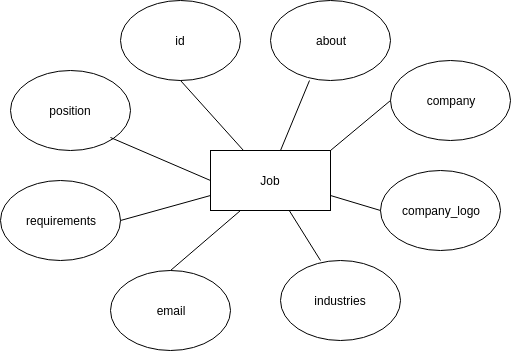
*Job*

This represents the job object in the database.

The attributes are represented as ovals below.

The industries attribute represents another relation to the industries relation.

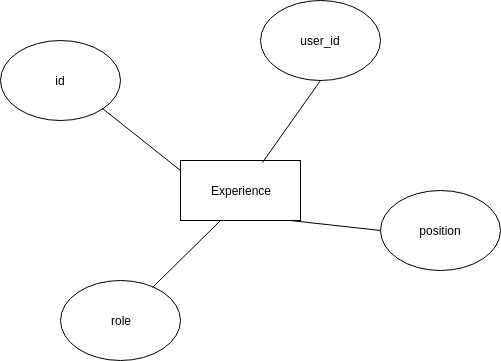
The industries attribute refers to a collection of items from the jobs\_jobs\_industries



Experience

This entity contains details about the experience and the user to whom the experience belongs to.

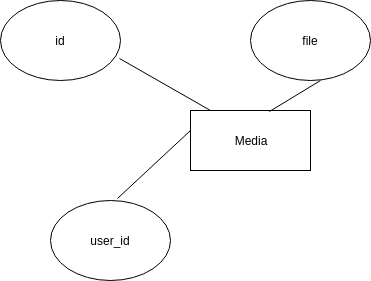
The user is specified in the user\_id



Media

This entity represents documents uploaded by the user to his/her account.

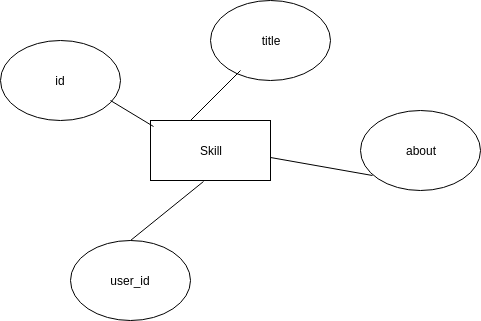
The media entity contains the file to be uploaded and the user\_id. This is the user to whom the uploaded file belongs to.



Skill

This is a skill that a user is proficient with.

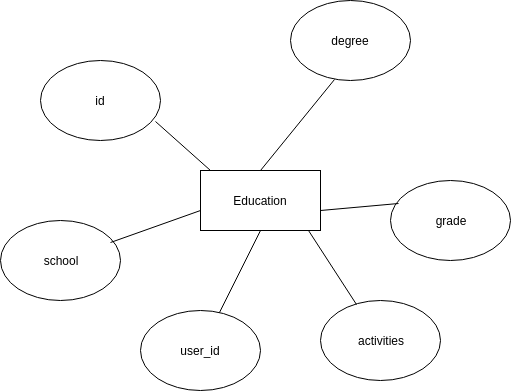
A skill contains details about the skill and also the user\_id of the user to whom the skill belongs to.



Education

This is the education that a user went through.

An education entity contains details about the education and also the user\_id of the user to whom the education belongs to.



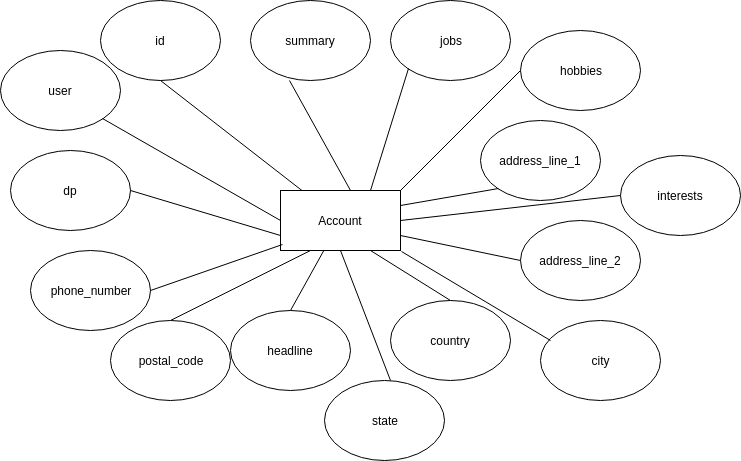
Account

This entity extends the user entity. It is an account for a user.

It contains extra attributes that are not contained in the django ‘user’ entity schema.

It is used to store extra information about a user

The interests of a user are got from the Industries relation in a many to many fashion.

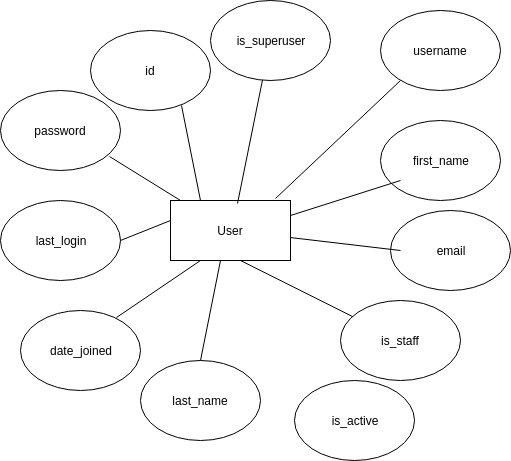


User

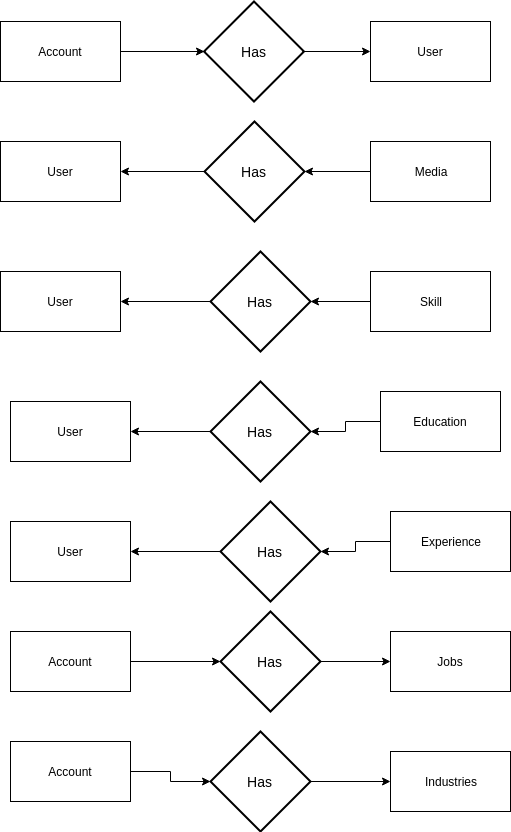
This entity is created by the django framework. Its attributes are predefined and thus in order to add extra attributes one extends the User model.

The following line imports the User model from django framework

from django.contrib.auth.models import User



Entity Relationship Diagrams



Use Case Diagrams

*Shortlisting A job*

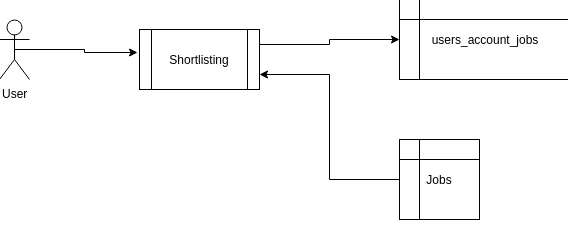
When a user short-lists a job, the system adds the new job into the user\_accounts\_jobs relation which is referred to by the jobs attribute of the account relation.

The user\_account\_jobs relation stores a job and the user to whom it has been shortlisted for.

Two data stores are in use here:

* Jobs – This is used to select the job that has been chosen for shortlisting
* user\_accounts\_jobs – This is used to store the shortlisted job and to whom the job has been shortlisted.

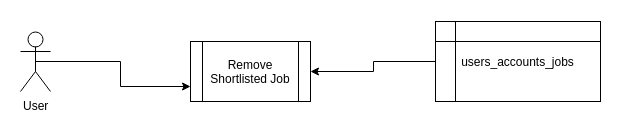
The shortlisted jobs are viewed by accessing the jobs attribute of a user.



*Removing Shortlisted Job*

The selected job is retrieved from the users\_accounts\_jobs relation and it is deleted. The users\_account\_jobs relation does not contain the actual jobs. It just contains records of the ids of jobs and ids of accounts that the jobs match to.

It is a lookup field. This means that deleting records from this relation does not delete the jobs in the Jobs relation.

**

*Uploading Media*

In uploading media, only one data source is used. This is the media data store.

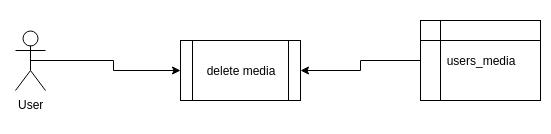
Input consists of the file to be uploaded.

Once the file is uploaded, it is stored in the media data store.



*Deleting Media*

In order to delete a media object, the selected media object is retrieved from the users\_media relation and is deleted.

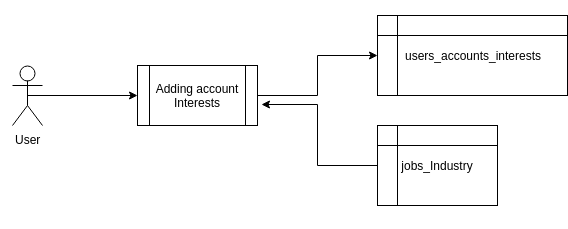


*Adding Interests to Account*

Two data stores are in use here:

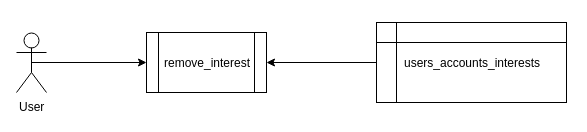
* jobs\_industry – This data store is represented by the entity Industries.
* users\_account\_interests – This data store is referenced to by the interests attribute of the account entity of a user.

An industry is selected from the jobs\_industry relation and its ‘id’ added to the users\_accounts\_interests together with the user\_id to whom the industry has been added as an interest

**

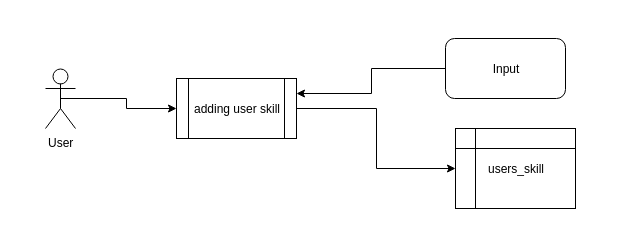
*Removing Interest from Account*

The interest object is selected from the users\_account\_interests table and it is deleted. Take in mind that this does not delete the industry object from the industries table.

**

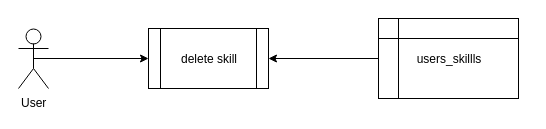
*Adding Skills to User Account*

The process takes in skill information as input and stores it in the users\_skill data store represented by the entity skill.

**

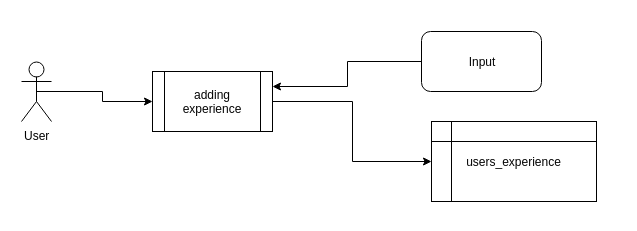
*Delete Skill*

The skill object is selected from the users\_skills(Skills) table and it is deleted

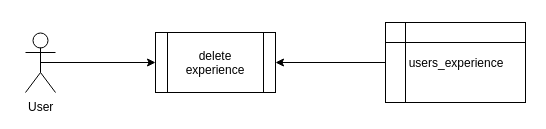
**

*Adding Experience*

Input about the experience is taken in by the process. This information is then stored in the users\_experience relation. Each experience is related to a user via a foreign key.

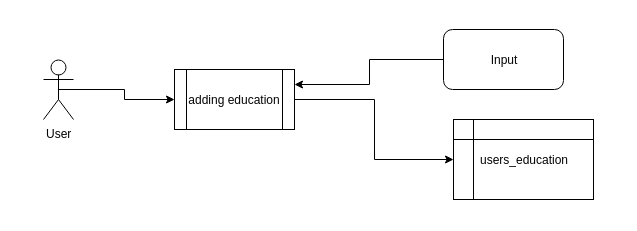
**

*Delete Experience*

The experience object is selected from the users\_experience(Experience) table and the row is deleted

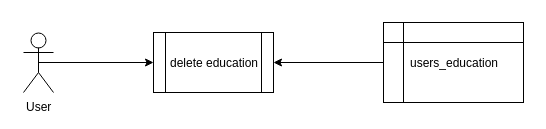
*Adding Education*

Input about the education is taken in by the process. This information is then stored in the users\_education relation. Each education is related to a user via a foreign key.

**

*Delete Education*

The education object is selected from the data store users\_education(Education) and the row is deleted

**

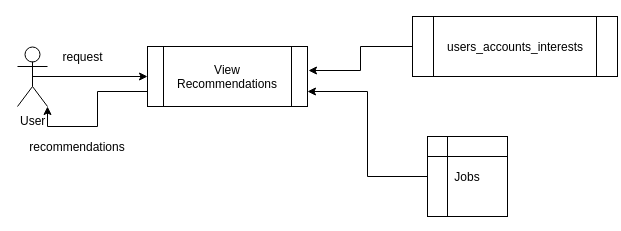
*View Recommendations*

When a user wants to view job recommendations, two data stores are used to process his/her request:

* users\_accounts\_interests – This contains industries that the user is interested in
* Jobs – This contains all the job objects

All jobs that have the same industries as those contained in the users\_accounts\_interests table rows of the specified user are returned to the user as recommendations.

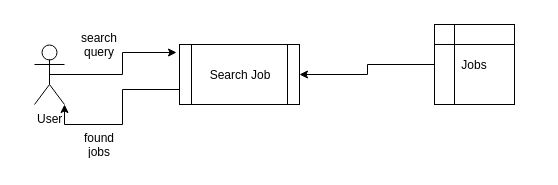
This means that a comparison is made on contents of the attribute ‘industries’ of the job entity and industries contained in rows of the users\_accounts\_interests where the account is the account of the logged in user.

**

*Search Jobs*

The search is made on the ‘Jobs’ table.

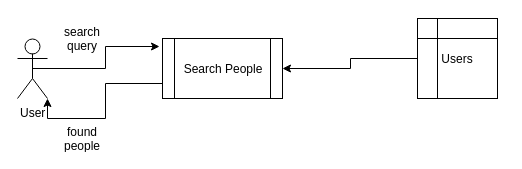
The company and position attributes of the jobs are matched against the string searched for. If matches are found, the jobs are returned to the user.

**

*Search People*

The search is made on the ‘Users’(auth\_user) table.

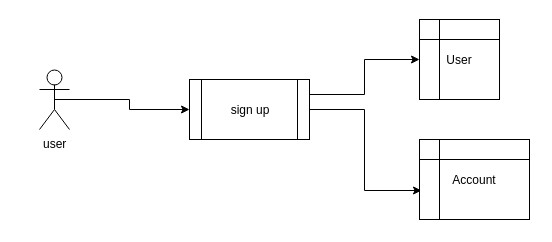
The user-name attribute of the users are matched against the string searched for. If matches are found, the users are returned to the user.

**

*Sign Up*

When a user wants to sign up, a user instance is created in the User relation. Also, an account object is created.

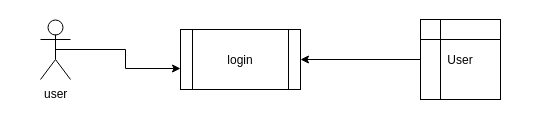
The account object created contains the just created user as the value of the user attribute of account.



*Login*

When a user wants to log in, a user object is searched for in the ‘User’ relation containing the user-name provided. If found the password supplied is matched against the password of the user retrieved from the database.

If they match, the user is logged into the system.

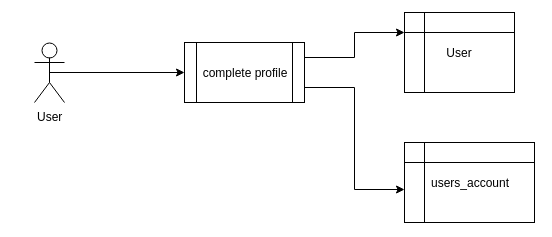


*Complete Profile*

Two data stores are in use here:

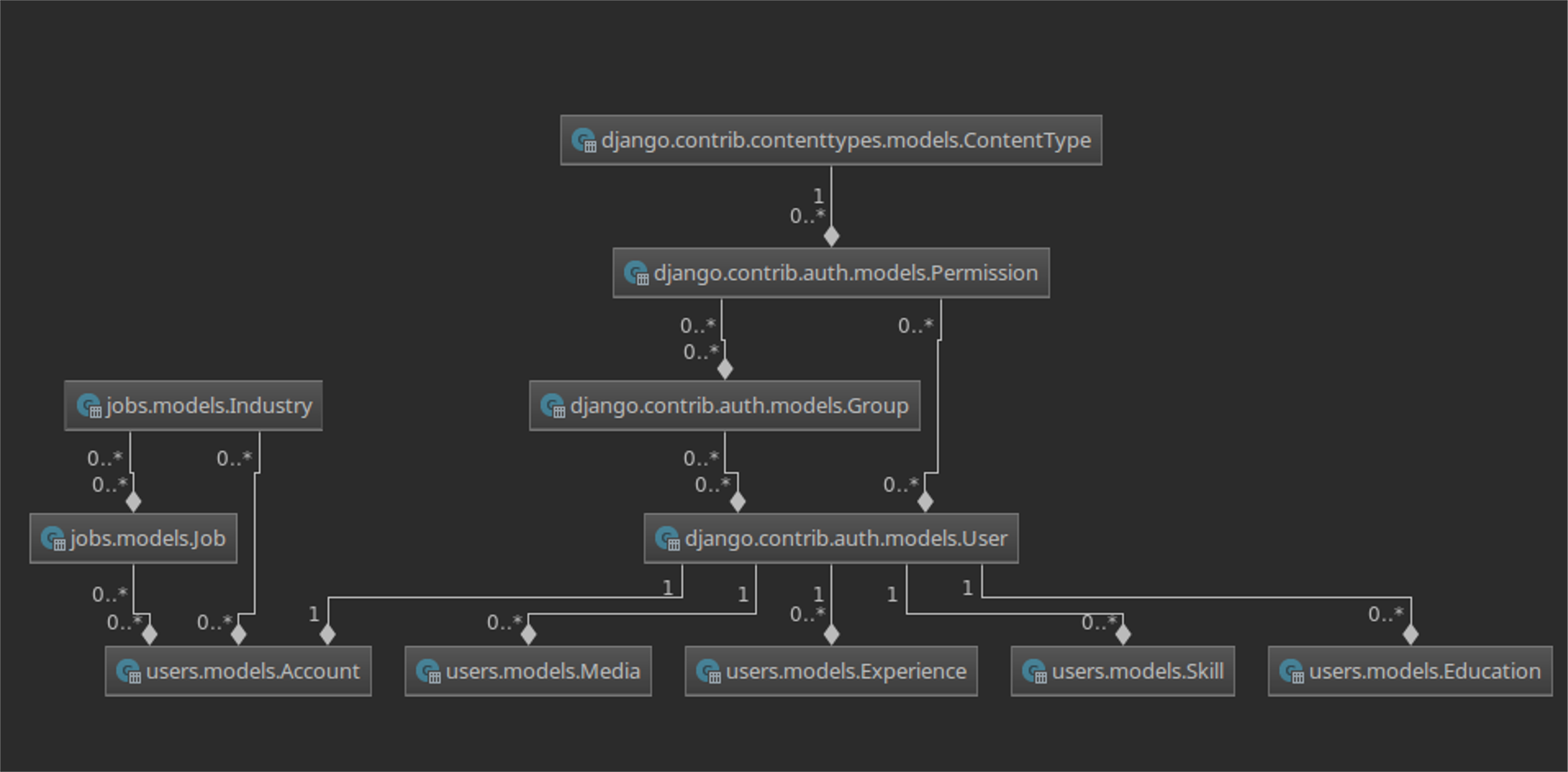
* User – The user object is retrieved from this relation
* users\_account – The account object whose user attribute is the user currently logged in is retrieved from this relation

Once a user completes their profile, the specified attributes belonging to the user object and account object are modified as neccessary



**Conceptual Model**

Below shows how the various tables in the database relate and the relationships.



Relational Database Schema

