**Models in Django**

* In Django**,** A model is a class that represents table or collection in our DB, and where every attribute of the class is a field of the table or collection. Models are defined in the app/models.py (in our example: myapp/models.py).
* Each model class maps to a single table in the database.
* Django provides a big in-built support for database operations. Django provides one inbuilt database **sqlite3**.
* Django can provide support for other databases also like oracle,mysql, postgresql, etc.
* Model is defined in Models.py file. This file can contain multiple models.

**Step 1:**

you need to tell Django that you’re going to use them by registering the application name in the INSTALLED\_APPS list in the settings.py of the project:(Register app into the INSTALLED\_APPS inside settings.py file.

A screenshot of a computer

Description automatically generated

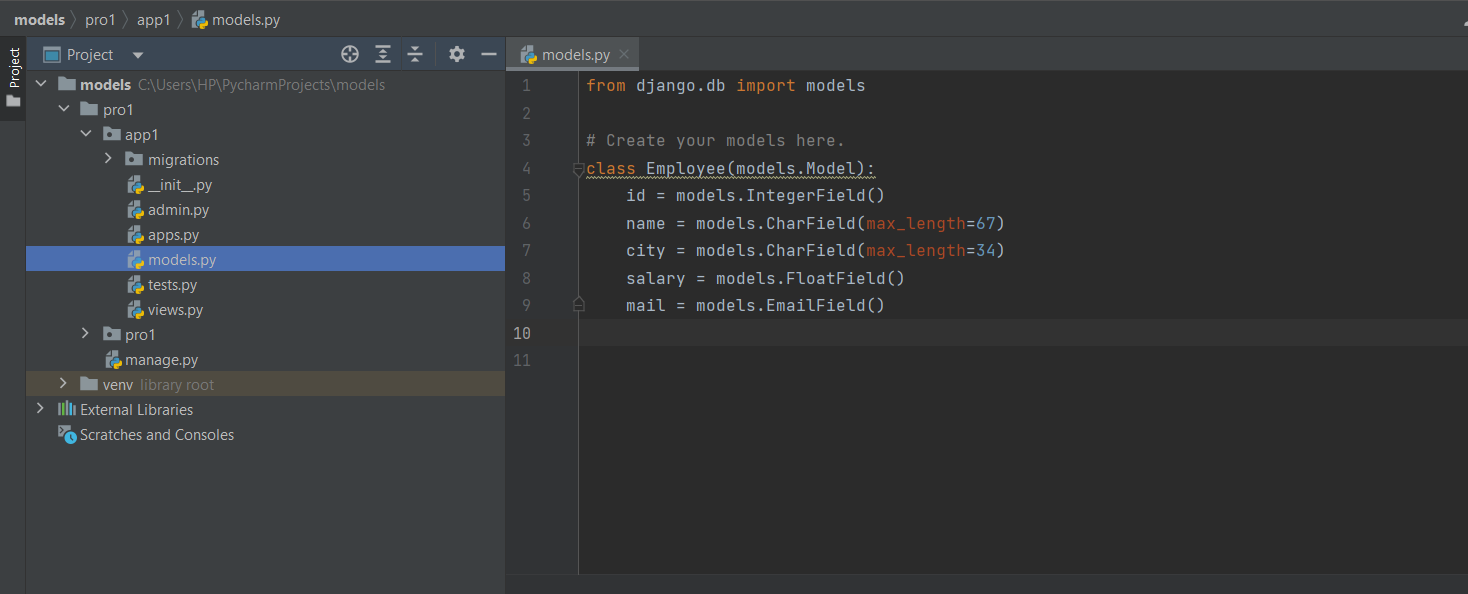
**Step 2: Define Models**

Open models.py file, Inside your app directory, locate the models.py file and define your models using Python classes. The fields are specified as class attributes and each attribute maps to a database column.

**<ProName> →AppName →models.py file**

**Fields and Options:**

* Field Types: Django provides various field types (CharField, IntegerField, DateTimeField, ForeignKey, etc.) that define the type of data each field will hold.
* Field Options: You can set various options for each field, like max\_length for CharField or null and blank for allowing null values or empty strings.
* Define Relationships (if needed):If your model has relationships with other models (one-to-one, one-to-many, many-to-many), you can define them using ForeignKey, OneToOneField, or ManyToManyField.

****

**we are creating a model Employee which has Five fields id, name, city, mail and salary.**

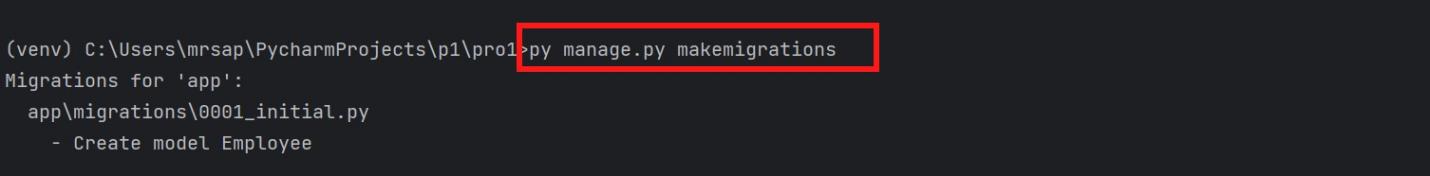
**Model Class:**

* A Model is a Python class which contains database information.
* A Model is a single, definitive source of information about our data. It contains fields and behavior of the data what we are storing.
* Each model maps to one database table.
* Every model is a Python class which is the child class of (django.db.models.Model)
* Each attribute of the model represents a database field.
* We have to write all model classes inside **‘models.py’ file.**

**Step 3:** **Make Migrations**

After defining your models, generate Django migrations using the following command:

**py manage.py makemigrations**



**The makemigrations in django the command is used to create database migration files based on the changes you’ve made to your models.(** python code gets converted into SQL**)**

**what is a migration file?**

A migration file contains Python code that represents the changes to the database schema, such as creating new tables, altering existing tables, or adding new fields and you can see the created file after running the command.

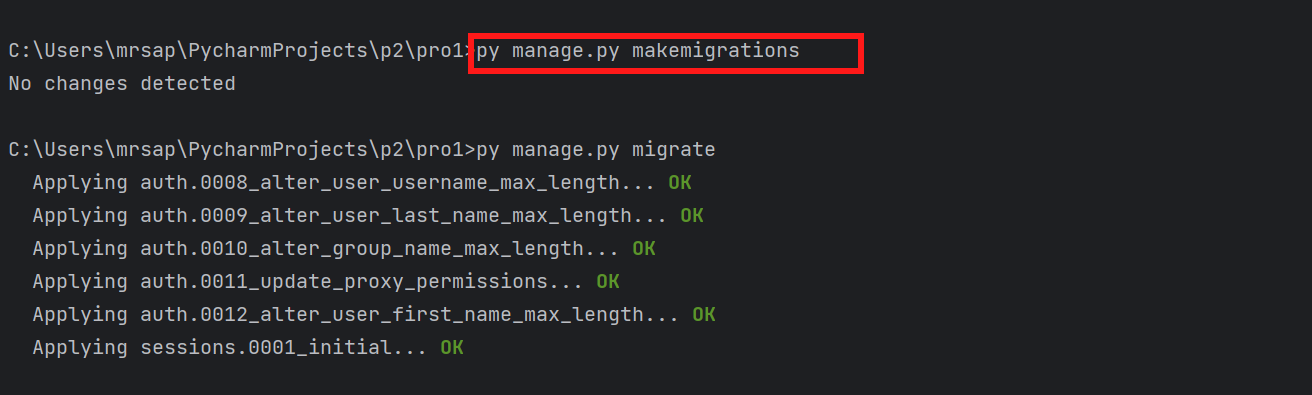
When you make changes to your models, such as adding new fields, changing field types, or even creating new models, running makemigrations analyzes these changes and generates migration files that capture the changes in a human-readable format. These migration files are stored in your app’s migrations directory.

**step 4: Apply Migrations:**

Apply the generated migrations to create or update your database schema:

**py manage.py migrate**

Execute the SQL code



* This command creates tables, modifies columns, adds indexes, and performs any other database-related operations needed to reflect the changes you’ve made.
* The **migrate** command takes care of the order in which migrations are applied, ensuring that dependencies between migrations are satisfied.

Once your models are defined and migrations are applied, you can use these models in your views, serializers, forms, etc., to interact with the database.

**Database Configuration:**

* Django by **default provides sqlite3 database**. If we want to use this database,we are not required to do any configurations.
* The default sqllite3 configurations in settings.py file are declared as follows.

A computer screen with green text

Description automatically generated

If we don't want sqlite3 database then we have to configure our own database with the following parameters.

1) ENGINE: Name of Database engine

2) NAME: Database Name

3) USER: Database Login user name

4) PASSWORD: Database Login password

5) HOST: The Machine on which database server is running

6) PORT: The port number on which database server is running.

**Configuration of MySQL Database**:

* First we have to create our own logical database in the MySQL.

**<mysql> create database <dbname>;**

* We have to install mysqlclient using pip :

**pip install mysqlclient**

A screen shot of a computer

Description automatically generated

* Go to the settings.py file change database type.

