**Create a user model manager:**

\*Create Super User command – comes with Django CLI

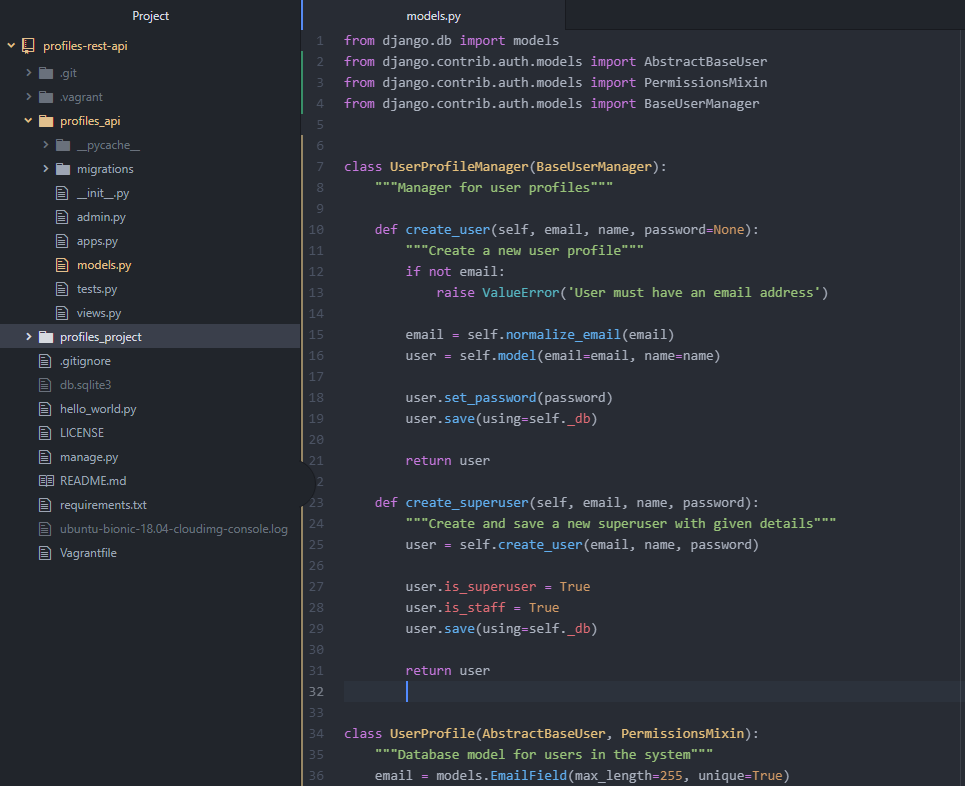
– to add super users to the system

A super user is an administrator user that has full control and will be able to access the Django admin and see all the models in the database.

Since we have customized our user model, we need to tell Django how to interact with this user model to create users (by default it expects a username and password but we have replaced it with email and password)

So we will create a custom manager that will handle creating users with email and password (instead of username and password)

We are going to inherit from BaseUserManager - default manager model in Django



1. password=None - As the Django password checking system works, it won’t accept a none password since it should be a hash(#).

2. Create a check if an email address has been provided. In case the email is an empty string or null value, an error is raised.

3. normalize\_email - it makes the second half email address (domain name) all lowercase.

Since domain name is case insensitive. First half of email is case sensitive.

4. Create our user model.

It creates a new model object and sets the email and the name.

5. We need to use set\_password function which comes with our user model as a part of the AbstractBaseUser.

- this function encrypts the password. Makes sure the password is converted to hash and never stored as plain text in the database

6. user.save(using=self.\_db)

Save the user model.

Standard is to specify the database we want to use.

Django supports multiple databases and we are going to use only one database in this case.

But we mentioned this line so that we can support multiple databases in the future.

7. create\_superuser func – create a new user using create\_user(parameters) func and also assign it a super user status.

8. user.is\_superuser = True

This is not specified in our UserProfile model because it is created automatically by PermissionsMixin.

Save the changes.