**فاز های پروژه my gis platform**

****

**فاز اول:** **طراحی معماری و ساختار اولیه پروژه**

****

چرا تنظیمات رو اینطوری چند بخشی کردیم (base, dev, prod)؟

| **محیط** | **توضیح** |
| --- | --- |
| **Development (dev)** | وقتی داریم کد می‌زنیم، دیباگ می‌کنیم، تست می‌گیریم |
| **Production (prod)** | وقتی پروژه رو دیپلوی کردیم و کاربران واقعی استفاده می‌کنن |
| **Base** | تنظیمات مشترک بین همه (پایه پروژه) |

Dev.py



Prod.py



طریقه اجرا برای لوکال:

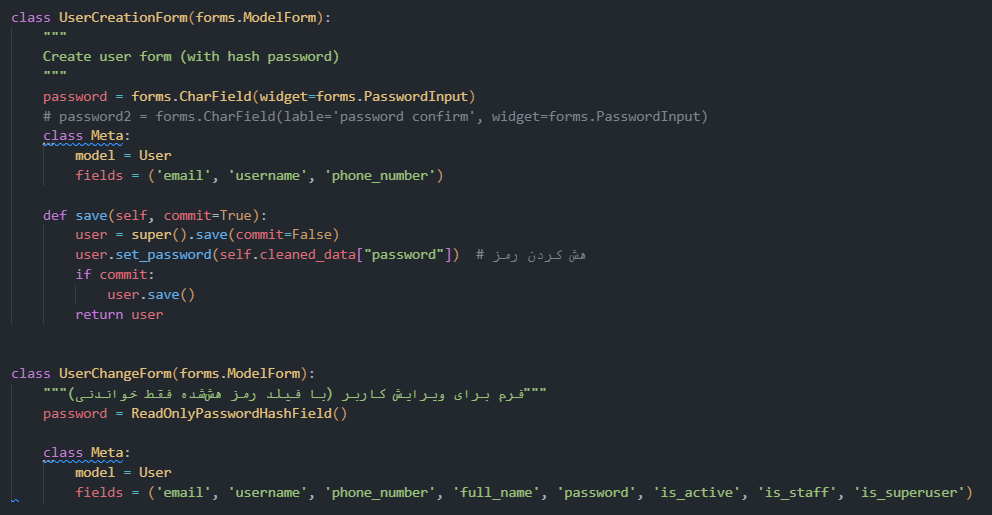
python manage.py runserver --settings=config.settings.dev

اگه خواستیم migrations بکنیم باید این حرکتو بزنیم

python manage.py makemigrations users --settings=config.settings.dev

نکته: وقتی از AbstractBaseUser برای ساخت یوزر استفاده میکنیم اگه کاربر توسط ادمین ساخته بشه پسوردش هش نمیشه برای برطرف کردن این مشکل باید از فرم استفاده کنیم و یک فرم برای ادمین بنویسیم و درادمین فراخوانیش کنیم.

Forms.py



Admin.py



کاستوم یوزری که ساختم  


نکته در abstractuser ما username را به صورت unique داریم ولی email یکتا نیست

نکته: چه زمانی از get\_user\_model استفاده کنیم و چه زمانی از settings.AUTH\_USER\_MODEL استفاده کنیم؟  
وقتی یک فیلد داشته باشیم که به User فارن کی شده بهتره از settings.AUTH\_USER\_MODEL استفاده کنیم و در بقیه موارد از get\_user\_model استفاده میکنیم

نکته:

iexact یعنی: برابری بدون حساسیت به حروف بزرگ و کوچک.

مثلاً:

EMAIL@EXAMPLE.COM با email@example.com یکی در نظر گرفته میشه.

بهتر از exact هست چون گاهی کاربر ایمیل رو با حروف بزرگ می‌نویسه.



filter() همیشه یک QuerySet برمی‌گردونه، حتی اگر فقط یک نتیجه باشه.

.first() باعث میشه فقط اولین نتیجه (یا None) رو بگیریم.

فرق بین get , filter در اینه که get همیشه یک شی برمیگردونه چه نتیجه نداشته باشه چه چنتا نتیجه داشته باشه excep میفرسته ولی فیلتر یک کوئری ست میفرسته.

user.check\_password(password) چیکار میکنه؟

این خط بررسی می‌کنه که:

رمز عبوری که کاربر وارد کرده (password)

با رمز هش‌شده‌ای که در دیتابیس ذخیره شده مطابقت داره یا نه.

پس:

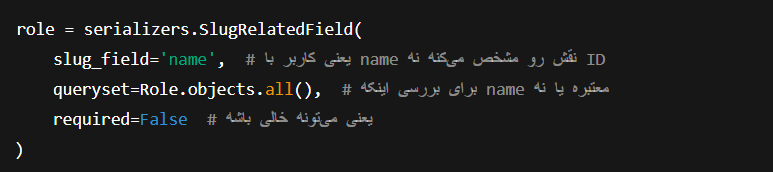
نیازی نیست رمز رو خودت هش کنی یا مقایسه‌ی دستی کنی.

بدون این، نمی‌تونی رمز رو درست چک کنی چون در دیتابیس به صورت هش‌شده ذخیره شده.



hasattr(user, 'role'): آیا یوزر اصلاً فیلدی به اسم role داره؟

خب ما داریم با کلاس کار میکنیم و hasattr یک تابع برای کلاس است



SlugRelatedField چی کار می‌کنه؟

به جای اینکه کاربر id نقش رو بده، از یک فیلد مثل name استفاده می‌کنه

**مثل پست که بخوایم کتگوری مشخص کنیم بجای ای دی میاد اسم کتگوری میده.**

****

partial=True ینی چی؟

ینی میتوان فقط فیلدهای خاصی را بدون نیاز به ارائه همه فیلدها به‌روزرسانی کرد و اکثرا در سریالایزر ها در put , patch استفاده میشه.

رای آپلود profile\_image در متد PUT، باید مطمئن بشید که درخواست شما از نوع multipart/form-data باشه، چون فایل‌های تصویری (یا هر نوع فایل) نمی‌تونن در قالب JSON ارسال بشن.

در چه مواقعی context={'request': request} رو به سریالایزر می‌فرستیم؟

مفهوم context:

در Django REST Framework (DRF)، context یک دیکشنری اختیاریه که می‌تونی به سریالایزر پاس بدی تا اطلاعات اضافی (مثل شیء درخواست، کاربر فعلی، یا تنظیمات خاص) رو در اختیار سریالایزر قرار بده.

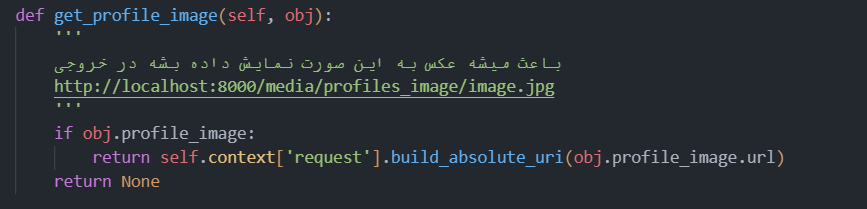
context={'request': request} معمولاً شیء درخواست HTTP (request) رو به سریالایزر می‌فرسته، که شامل اطلاعاتی مثل کاربر لاگین‌شده، هدرها، یا URL پایه سروره.



هر وقت بخوایم اطلاعات request.user در سریالایزر دسترسی داشته باشیم این کار میکنیم مثلا:

1- اگر سریالایزر نیاز به اطلاعات کاربر لاگین‌شده داشته باشه (مثلاً برای اعتبارسنجی یا محدود کردن دسترسی)، می‌تونی request.user رو از طریق context دریافت کنی.

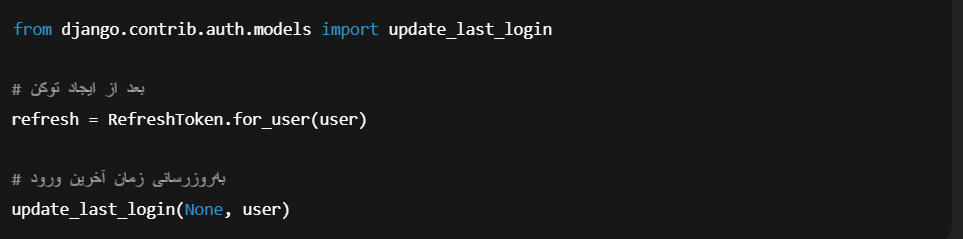
2- اگر بخوای URL کامل یک فایل (مثل تصویر پروفایل) رو در خروجی JSON تولید کنی، نیاز به request داری تا از متد build\_absolute\_uri استفاده کنی.



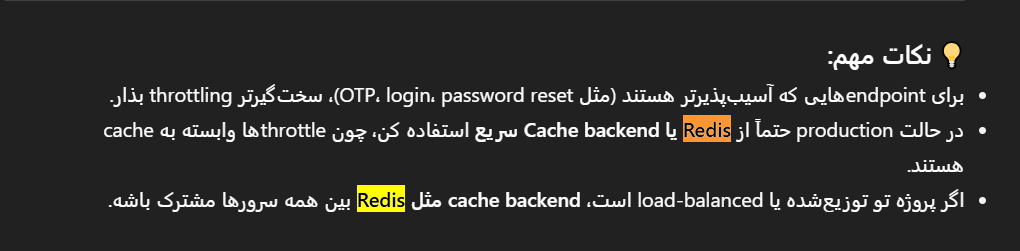
طرز بک اپ گرفتن از دیتابیس

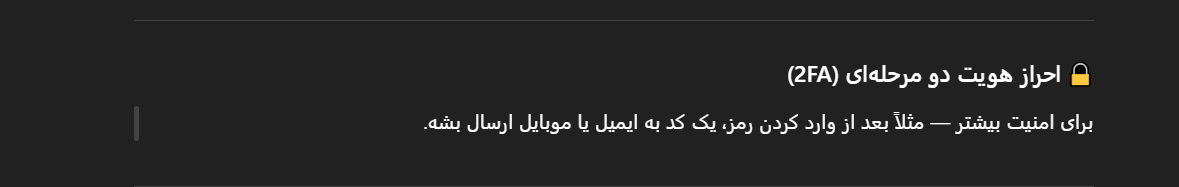
python manage.py dumpdata > backup.json --settings=config.settings.dev

اگه از from rest\_framework\_simplejwt.views import TokenObtainPairView در url برای ساخت توکن به صورت مستقیم استفاده کنیم last login خودش اپدیت میشه ولی چون توی این پروژه کاستوم شده پس باید بعد هر refresh صداش بزنیم



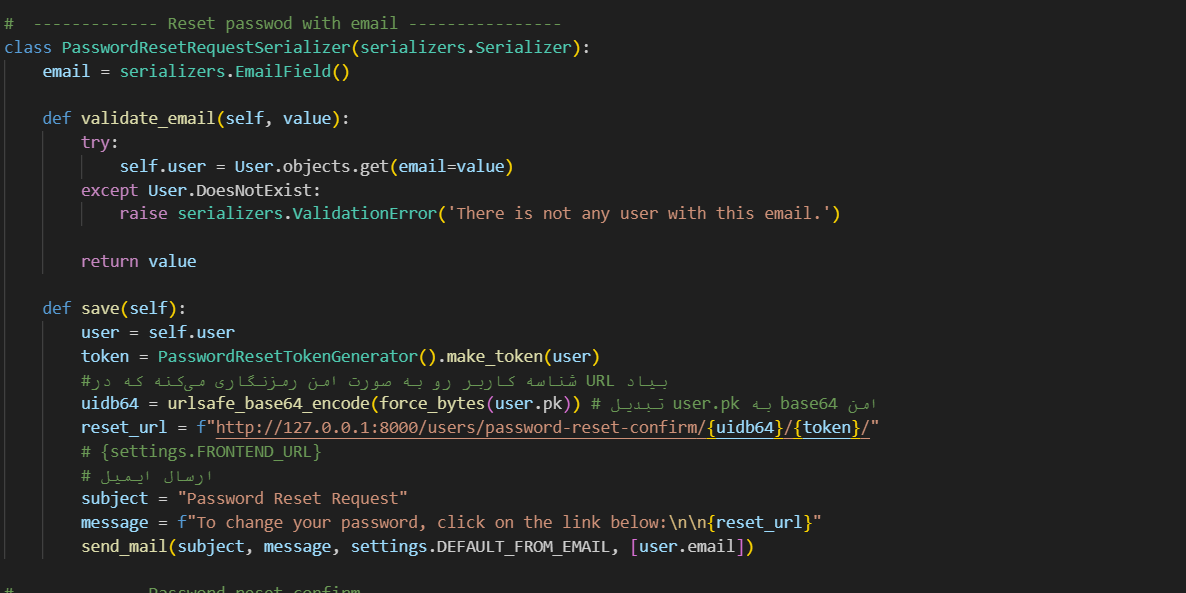








فراموشی رمز عبور با ایمیل: passwordReset



با استفاده از passwordresettokengnerator یک توکن امن ساخته میشه

و با استفاده ازین دیگه نیازی به ساخت مدل برای توکن ها نداریم چون PasswordResetTokenGenerator به‌صورت داخلی با استفاده از اطلاعات کاربر (مثل last\_login و password) توکن را به‌صورت امن و قابل اعتبارسنجی تولید می‌کنه.

یعنی:

این توکن نیاز به ذخیره در دیتابیس نداره.

بک‌اند با توجه به uid و token دریافتی، بررسی می‌کنه که آیا توکن هنوز معتبره یا نه.

به‌صورت پیش‌فرض، اگر کاربر پسوردشو عوض کرده باشه یا لاگین کرده باشه، توکن دیگه معتبر نخواهد بود.

اعتبار کد تا 3 روز است و با کد زیر در ستینگ میتوان زمان ان را تغییر داد.

PASSWORD\_RESET\_TIMEOUT = 3600

وقتی کاربر لاگین میکنه اگه 2FA فعال باشه ما میدونیم کاربر کیه پس وقتی میخوام کد رو براش ارسال کنیم دیگه شماره موبایل یا ایمیلشو نمیگیریم که مطابقت بدیم با کد بلکه یوزر رو از طریق user\_id که توی درخواست برای لاگین گرفتیم پیداش میکنیم اینکار امنیتی تره و همچنین هر چقدر مقدار برگشتی کمتر به اطلاعات حساس (مثل ایمیل، موبایل) وابسته باشه، امنیت بیشتره. چون ممکنه در لاگ‌ها یا توسط ابزارهای مانیتورینگ لو بره.

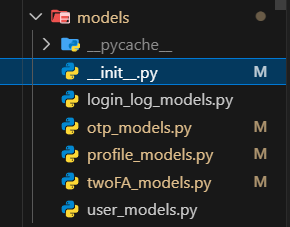
نکته مهم: کاربر ممکنه هنوز در سیستم ثبت‌نام کامل نکرده باشه ولی شماره موبایلش رو وارد کرده باشه.

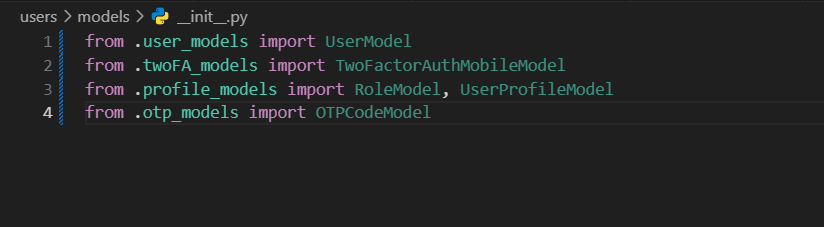
در جنگو، **.exclude()** روشی است که در QuerySetها برای بازیابی اشیاء که با معیارهای مشخص شده مطابقت ندارند، استفاده می‌شود. این روش از نظر منطقی برعکس .filter() است. در حالی که .filter() اشیاء را انتخاب می‌کند که یک شرط را برآورده می‌کنند، .exclude() آنهایی را انتخاب می‌کند که این شرط را برآورده نمی‌کنند.

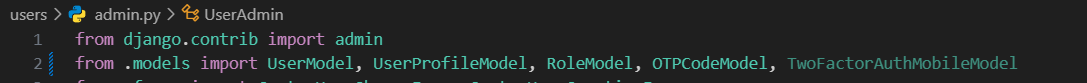


داره میگه کاربر رو براساس ایمیل پیدا کن به شرطی که کاربر پیدا شده کاربر حال حاضر نباشد

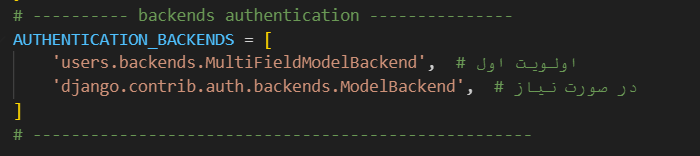
اگه ما برای model, view, serializer و ... میایم پوشه های جدا میسازیم و در هر پوشه فایل \_\_init\_\_.py میسازیم  
باید درن این فایل همه model ها و ... را تعریف کنیم و سپس اگه جای دیگه تعریف شدن باید به صورت زیر صدا بزنیمشون



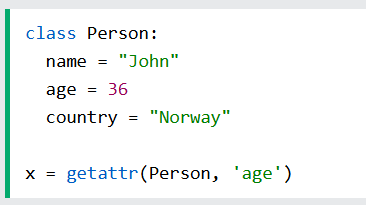
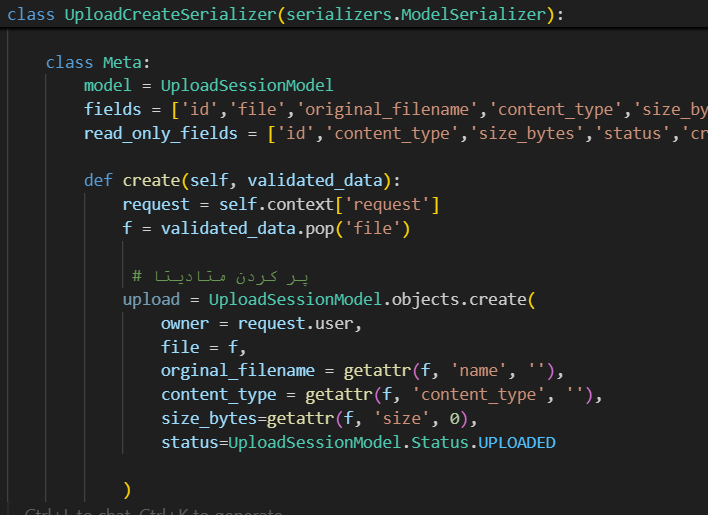




اگه از backend استفاده کرده باشیم برای احراز هویت باید در ستینگ معرفیش کنیم



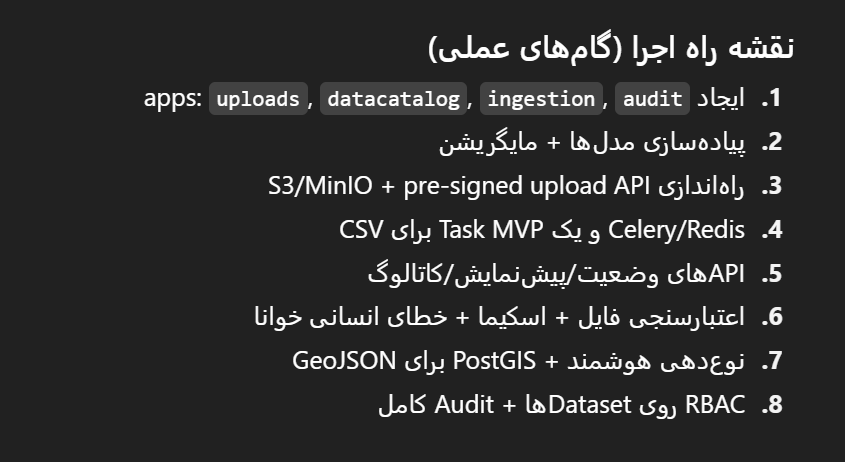
Getattr():

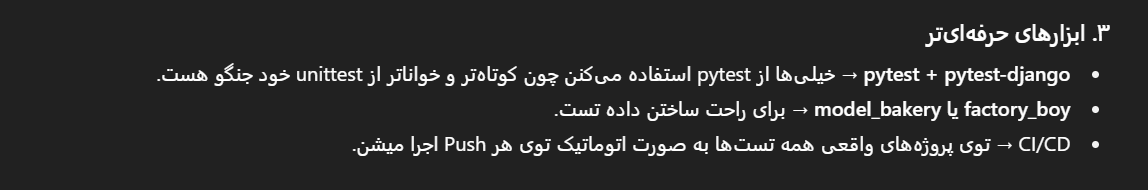
 

از فایل که همون f هست(کاربر ارسال کرده) بیا name و content\_type و sizeرو بگیر و بریز توی مدل مربوطه.

تفاوت بین set\_password و make\_password:

**فاز دوم:** **داده‌های داخلی (Uploaded)**



****