اگر یک فرایند طولانی مدت بخشی از گردش کار برنامه شما است، بجای اینکه response را مسدود کنید، باید آن را در پس زمینه، خارج از جریان عادی request/response انجام دهید. و یا ممکن است بخواهید یکسری از کارها به صورت اتوماتیک در دوره زمانی خاصی انجام شوند. در این حالت میتوانید از celery استفاده کنید. در این دوره میبینید که چطور میتوانید در جنگو درخواست‌های زمانبر را توسط celery مدیریت کنید.

=====================================================================================

در این پروژه با استفاده از celery درخواست‌های زمانبر را در جنگو مدیریت میکنیم مثلا زمانی که میخواهیم ایمیل ارسال کنیم میتوانیم این کار را به celery بسپاریم. در این دوره میبینید که چطور میتوانیم با استفاده از celery-beat کارهایی که باید به صورت تکراری در زمانی خاص انجام شوند را مدیریت کنیم. همچنین با ابزار supervisor آشنا میشوید که برای اجرای workerها در پس‌زمینه کاربرد دارد.

http://supervisord.org/

=====================================================================================

در این قسمت celery را داخل جنگو تنظیم میکنیم تا بتوانیم از این دو در کنار همدیگر استفاده کنیم. برای اینکار ابتدا کانفیگ‌های مربوط به celery را در فایل settings.py مینویسیم. سپس در کنار فایل settings.py یک فایل دیگر به نام celery.py ایجاد کرده و یک نمونه جدید از کلاس Celery در آن ایجاد میکنیم. بعد از آن با استفاده از متد autodiscover\_tasks تمام فایل‌هایی که نام آن tasks باشد به صورت اتوماتیک پیدا شده و متدهای داخل آن اجرا میشوند.

<https://docs.celeryproject.org/en/stable/django/first-steps-with-django.html>

تمام configهایی که در جنگو وجود دارد به طور کلی با حروف بزرگ نوشته می‌شود

۱. اولین confilg مربوط به CELERY\_BROKER\_URL میباشد که به این معنی میباشد که brokerی که قرار است celery با آن کار کند کجا است

۲. config مهم بعدی CELERY\_RESULT\_BACKEND میباشد که همان rpc میباشد

به طور کلی توجه کنید که برای اضافه کردن کانفیگ های celery بهتر است که به ابتدای هر کانفیگ \_CELERY اضافه کنید

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

در فایل celery.py ابتدا باید یک virualenv را ست کنید که نام DJANGO\_SETTINGS\_MODULE ثابت میباشد و سپس باید آدرس فایل settings پروژه به آن داده شود (چرا که باید config ها را بخواند)

os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'A.settings')

در‌واقع setdefault یک setی را برای شما ایجاد میکند که کلید آن DJANGO\_SETTINGS\_MODULE و مقدار آن A.settings میباشد

مرحله بعد باید یک instance جدید از کلاس Celery ایجاد کنید که اسم آن را celery\_app قرار میدهیم (پارامتر main را برابر اسم پروژه قرار میدهیم )

مرحله ی بعد این است که config هایی که در settings.py نوشته شده است را به celery\_app بشناسانیم

celery\_app.config\_from\_object('django.conf:settings', namespace='CELERY')

همان‌طور که میدانید در جنگو برای import کردن settings از ( from django.conf import settings ) استفاده می‌شود منظور از   
django.conf:settings همان import میباشد   
در فایل settings کانفیگهای بسیار زیادی وجود دارد حال ما با استفاده از namespace به celery میگوییم آن‌هایی را بخوان که با CELERY شروع می‌شوند

آخرین کاری که باید انجام داد این است که به celery\_app بگوییم که در پروژه تسک هایی که مربوط به تو میباشد را پیدا کن و بیاور به این صورت که ()celery\_app.autodiscover\_tasks این کار را به این صورت انجام میدهد که در تمامی appهای جنگویی که وجود دارد به دنبال فایل tasks.py میگردد   
میتوانیم به وسیله related\_name نام tasks.py را عوض کنیم

celery\_app.autodiscover\_tasks(related\_name=”mesal”)

اکنون به دنبال mesal.py میگرد

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

اکنون باید به جنگو بفهمانیم به محض اینکه اجرا شدی celeryهم اجرا کن برای این کار باید در فایل \_\_init\_\_ خط زیر را قرار دهیم

from .celery import celery\_app

=====================================================================================

shared\_task

taskهایی که می نویسید احتمالاً در برنامه های reusable خواهد بود و برنامه های reusable نمی توانند به خود پروژه بستگی داشته باشند ، بنابراین شما نمی توانید نمونه برنامه خود را مستقیماً import کنید. دکوراتورshared\_task به شما امکان می دهد بدون داشتن نمونه برنامه‌ای‌‌ خاص، taskهایی را ایجاد کنید.

تا قبل وقتی میخواستیم فانکشنی را به celery بشناسانیم از نام instance سلری یک decorator ایجاد میکردیم (در این پروژه   
celery\_app.task) و از این دکوریتور استفاده میکردیم اما در جنگو لازم نیست به این صورت عمل کنیم تنها کافی است   
shared\_task را import کنید و از این کوریتور استفاده کنید (دقت کنید که حتماً باید از دکوریتور shared\_task استفاده کنید و نباید از دکوریتور task استفاده شود)

from celery import shared\_task

@shared\_task

def adding(x, y):  
 ….

در بعضی از مواقع به شما celery به شما ارور میدهد که پکیج furure وجود ندارد شما باید با دستور pip install future این پکیج را نصب کنید

توجه کنید که بعد از هر تغییر در پروژه باید workerتان را ریستارت کنید

celery -A A worker -l info

=====================================================================================

یکسری کارها هستند که شما میخواهید آن‌ها را در بازه های زمانی مختلف به صورت تکراری آن‌ها را انجام دهید مثلاً ما یک کاری داریم که قرار است که هر 20 ثانیه یکبار انجام شود در این مورد شما میتوانید از periodic tasks ها استفاده کنید

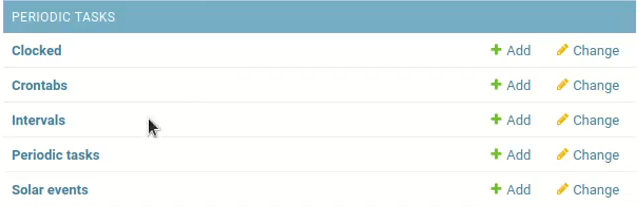
[https://docs.celeryproject.org/en/stable/userguide/periodic-tasks.html#periodic-tasks](https://docs.celeryproject.org/en/stable/userguide/periodic-tasks.html" \l "periodic-tasks)

دو روش نسبتاً متفاوتی وجود دارد یکی برای پایتون و دیگری برای جنگو هرچند که روش پایتون هم میتوانید در جنگو استفاده کنید اما برای جنگو پکیجی وجود دارد که سختی کار را به شدت کم میکند

برای جنگو پکیجی وجود دارد به نام django-celery-beat که میتوانید این کار را انجام دهید

[https://docs.celeryproject.org/en/stable/userguide/periodic-tasks.html#id8](https://docs.celeryproject.org/en/stable/userguide/periodic-tasks.html" \l "id8)

ابتدا شما باید django-celery-beat را به installed apps ها اضافه کنید بعد از اضافه کردن حتماً باید دستور  
python manage.py migrate را اجرا کنید بعد از آن داخل admin پنل به شما طبق عکس زیر امکاناتی را میدهد

  
که هر کدام از آن‌ها امکاناتی جهت اجرای تسک ها در بازه ی زمانی خاصی را میدهد ( به جز periodic tasks که تسک را باید انتخاب کنید )

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

در جنگو برای ارسال ایمیل میتوانید از طریق زیر عمل کنید :

from django.core.mail import send\_mail

send\_mail('Subject', 'message', 'from', ['to',….])

send\_mail('Test', 'This is a test email for django-celery-beat', 'amirbig44@gmail.com', ['mongard.amir@gmail.com'])

بعد از این فانکشن مورد نظرتان را نوشتید و داخل ادمین پنل تغییرات مورد نظرتان را انتخاب کردید لازم است دستور زیر را اجرا کنید

$ celery -A A beat -l INFO --scheduler django\_celery\_beat.schedulers:DatabaseScheduler

توجه داشته باشید که حتماً یک worker هم باید اجرا داشته باشید که تسک هایی که celery-beat داخل صف فرستاده است اجرا شوند

=====================================================================================

Supervisor

Supervisor یک سیستم کاربر/سرور است که به کاربران خود اجازه می دهد تا تعدادی از فرایندها را روی سیستم عاملهای مشابه یونیکس نظارت و کنترل کنند. این برنامه برخی از اهداف مشابه برنامه هایی مانند launchd ، Daemontools و Runit را به اشتراک می گذارد. این منظور برای کنترل فرآیندهای مربوط به یک پروژه یا کاربر استفاده می شود و به معنای شروع مانند هر برنامه دیگری در زمان بوت شدن است.

http://supervisord.org/

به زبانی دیگر به شما اجازه میدهد تسک هایتان را در پس زمینه اجرا کنید

برای اجرای supervisor باید ۷ مرحله را طی کنید که در فایل supervisor\_command.txt وجود دارد

service supervisor status

تمامی config های supervisor در پوشه ی زیر قرار میگیرد

*etc*/supervisor/conf.d