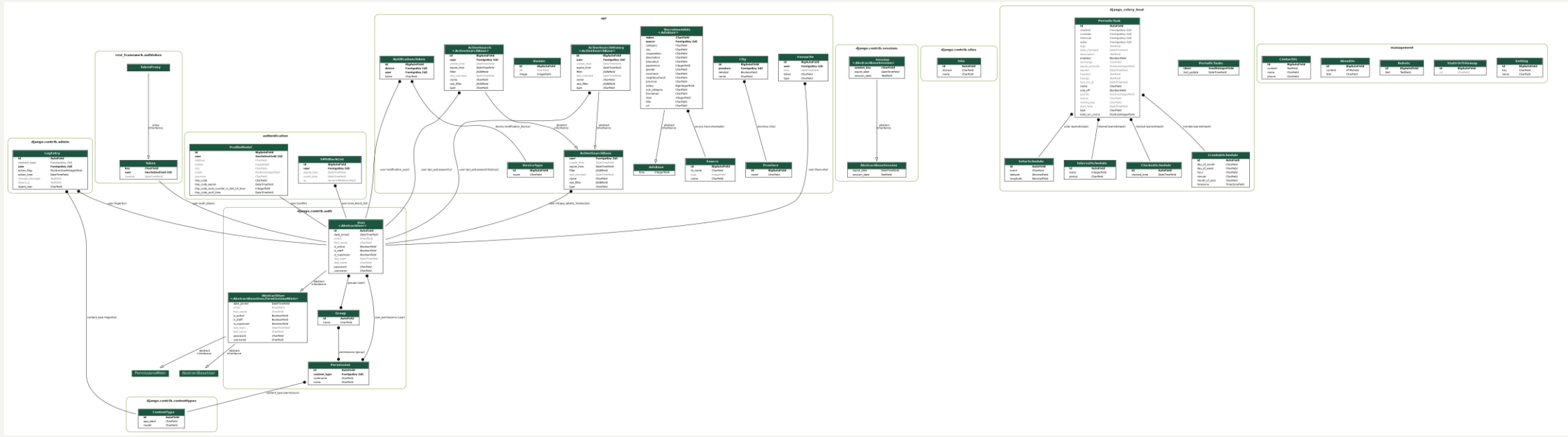
JObJOO DataBase

visit JOBJOO here

https://developerszaris.ir/

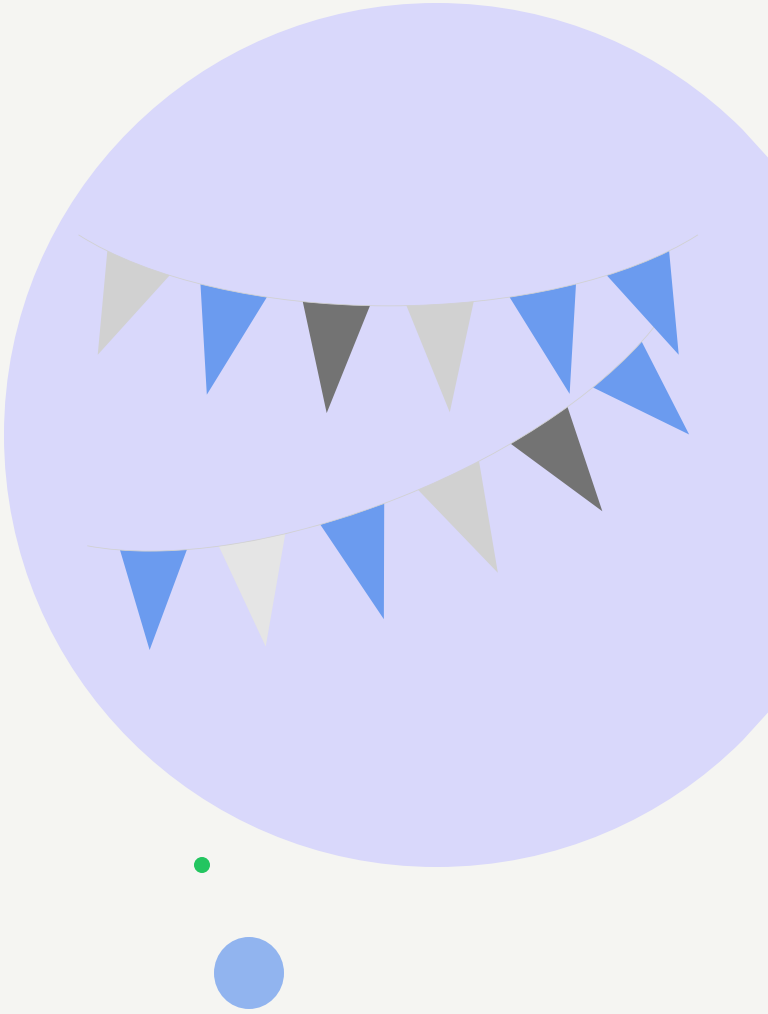
JObJOO Data base Er model



Celery - Distributed Task Queue

In broad terms, the reason why we use async tasks queues is because we want to answer quickly to our users. The simplest use case for it is to delegate long lasting CPU jobs. But the most popular reason people use async tasks is probably to execute external API calls. Whenever you depend on external services, you no longer have control over how long things will take to be ready. It might also be the case that they will never be ready, since the system might be down or broken. Another good use for async tasks is to prepare and cache result values. You can also use them to spread bulk database insertions over time. This can help you avoid DDoS'ing your own database. Cron jobs are yet another good example of things you can do with them.

There are many tools available to manage async tasks in Python. RQ seems to be getting some attention lately, but Celery is the all time champion so far.



Using celery in jobjoo

You can find codes related to celery in backend/api/task.py

We used celery to increase our site performance

```python
from requests import post
from datetime import datetime
from api.services import recruiment_services
import json

from searchengine.celery import app


@app.task
def call_sheypoor_spider():
    url = "http://localhost:7000/schedule.json" #scrapy port in server is 7000
    try:
        response = post(url, data={
            'project': 'crawler',
            'spider': 'sheypoor',
            'category': 'recruitment'
        })
    except:
        return
    json_data = json.loads(response.content)

    print(json_data)

@app.task
def call_divar_spider():
    url = "http://localhost:7000/schedule.json" #scrapy port in server is 7000
    try:
        response = post(url, data={
            'project': 'crawler',
            'spider': 'divar',
            'category': 'jobs'
        })
    except:
        return
    json_data = json.loads(response.content)

    print(json_data)


@app.task
def recruiment_service(function, *args, **kwargs):
    class_method = getattr(recruiment_services, function)
    return class_method(recruiment_services(), *args, **kwargs)
```