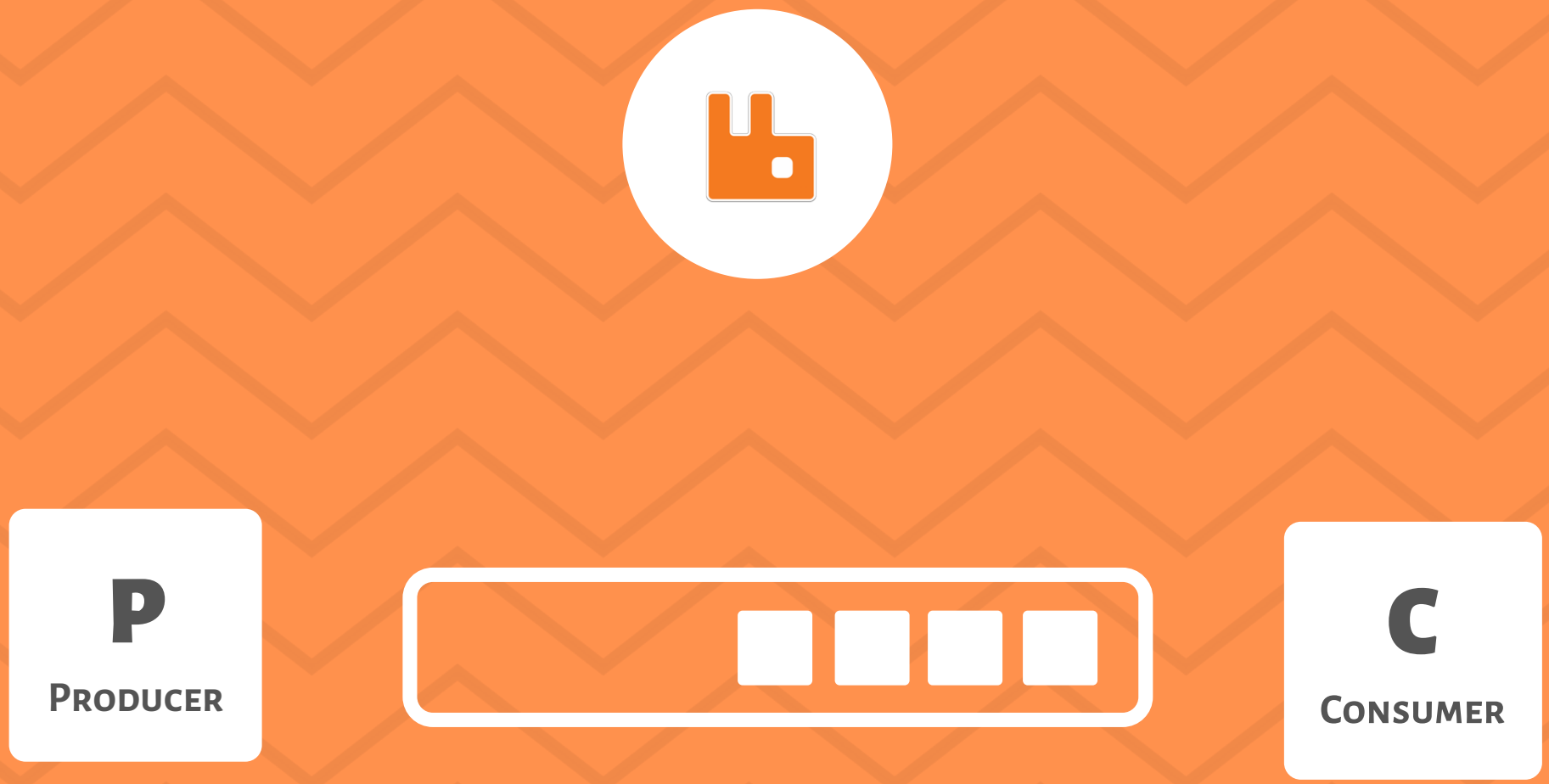
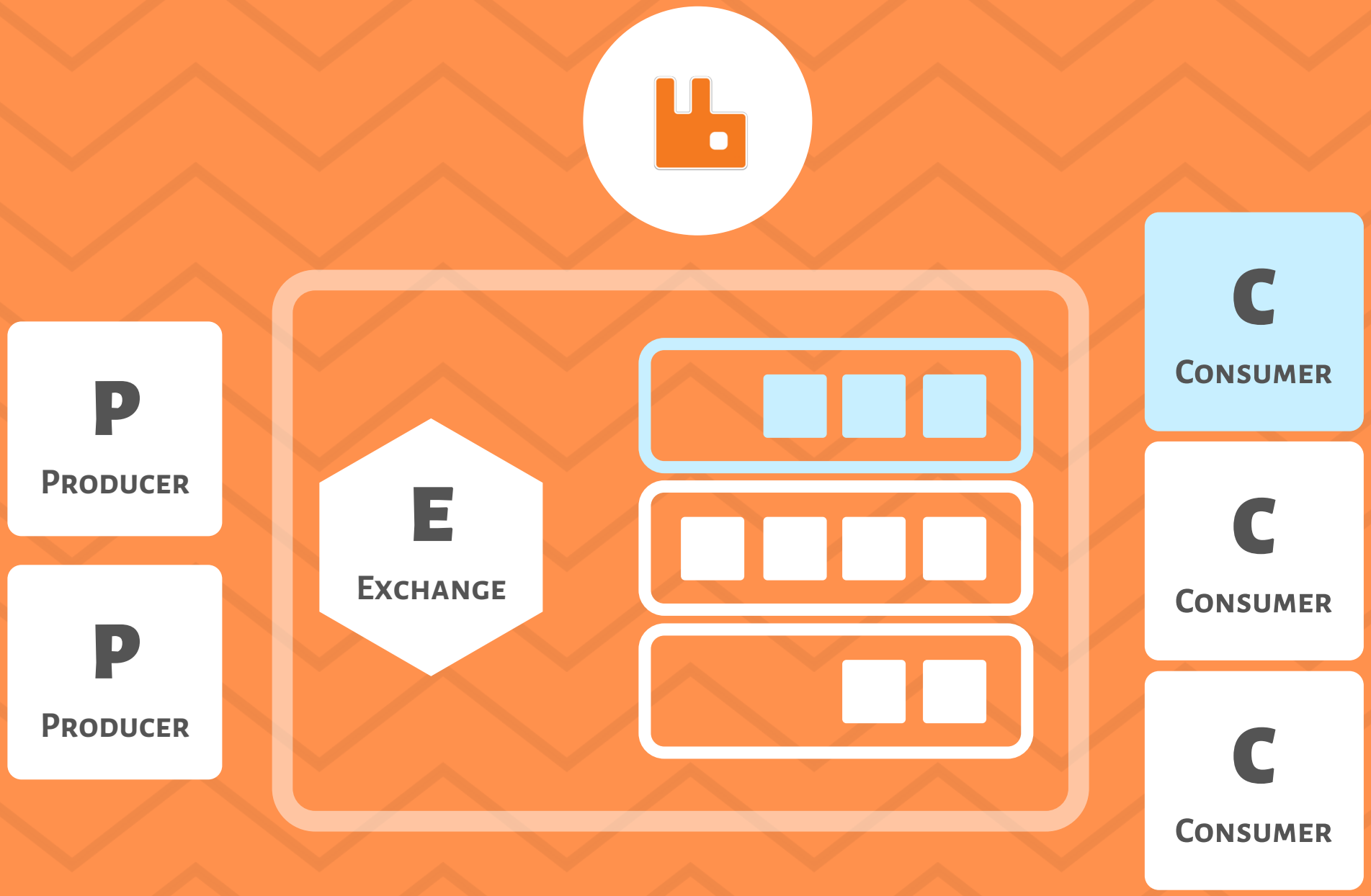
RabbitMQ + Celery

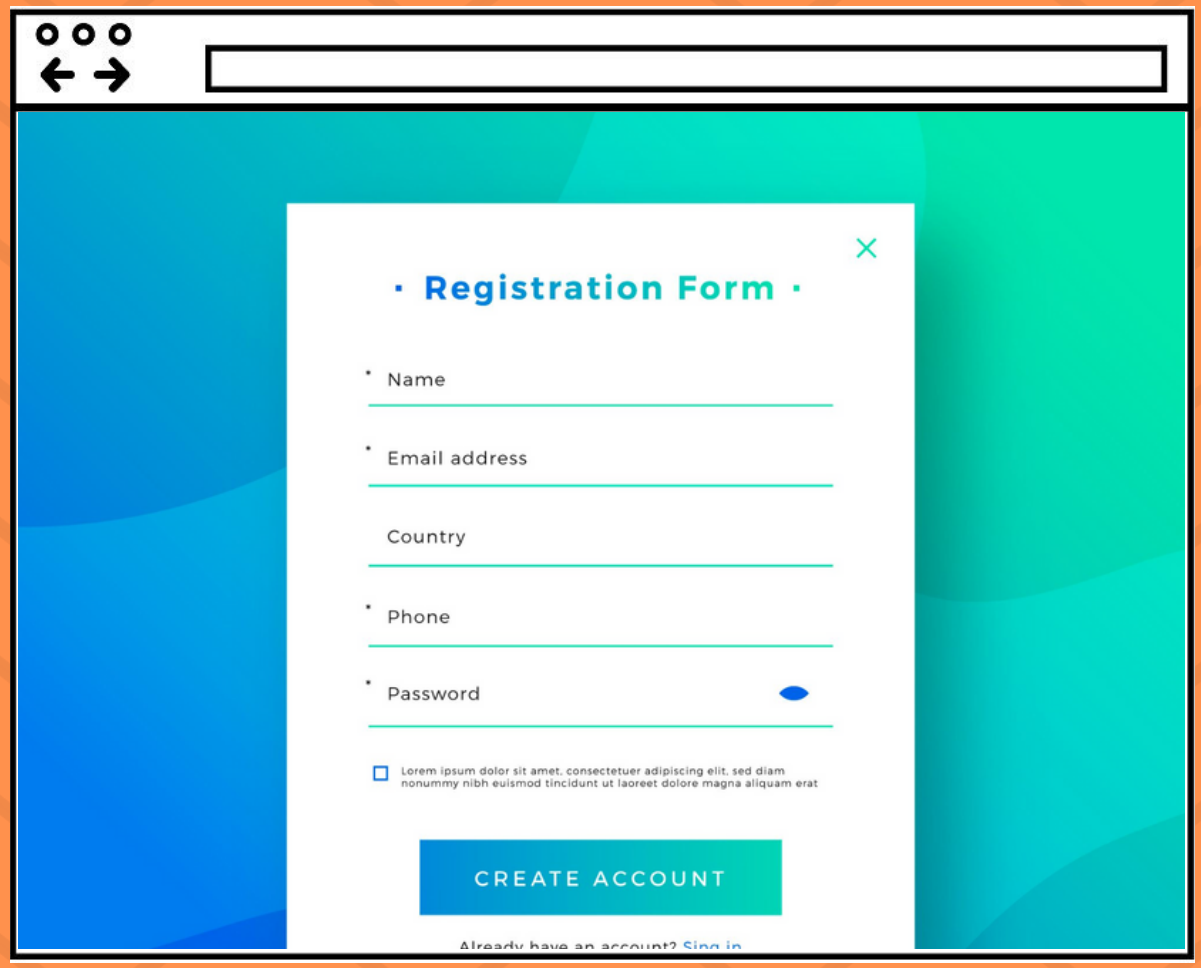
@ALEMANGUI


P
Producer
C
Consumer

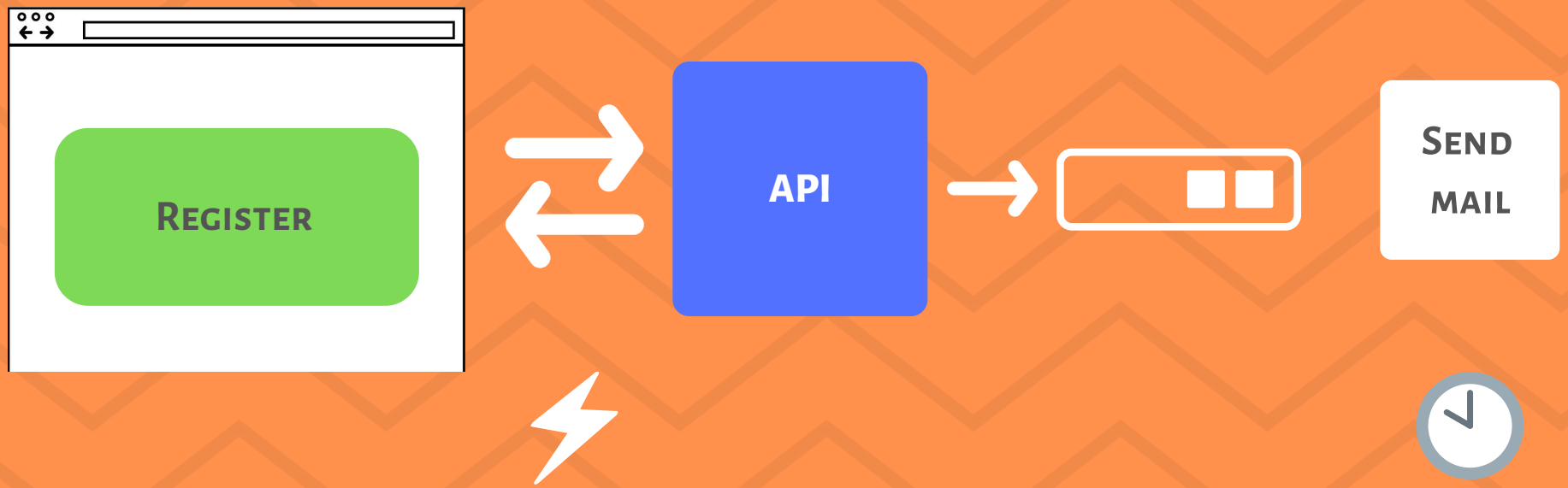

P
Producer
P
Producer
E
Exchange
C
Consumer
C
Consumer
C
Consumer


Montre le
CODE

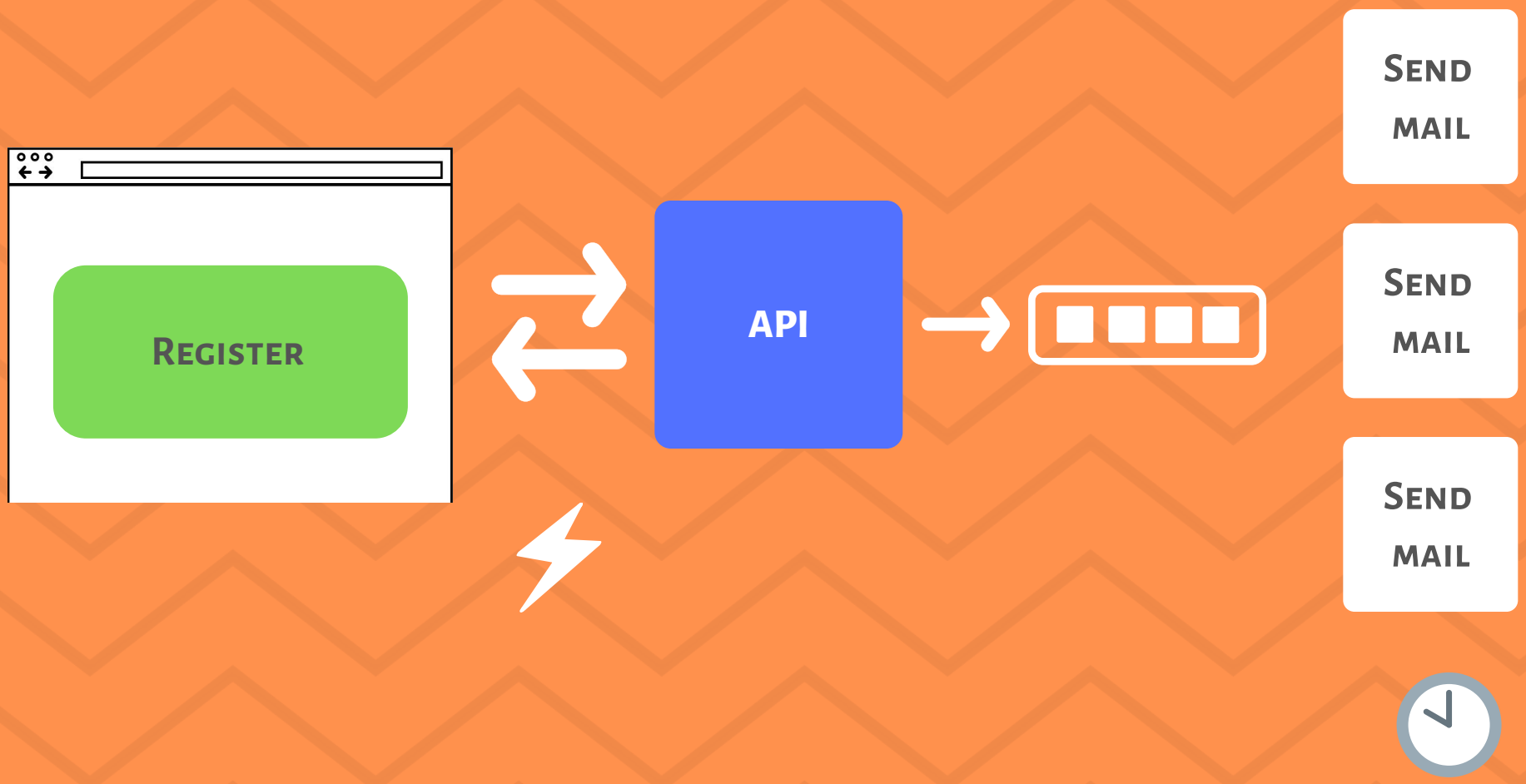
Pourquoi ?


Registration Form
Name
Email address
Country
Phone
Password
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
CREATE ACCOUNT

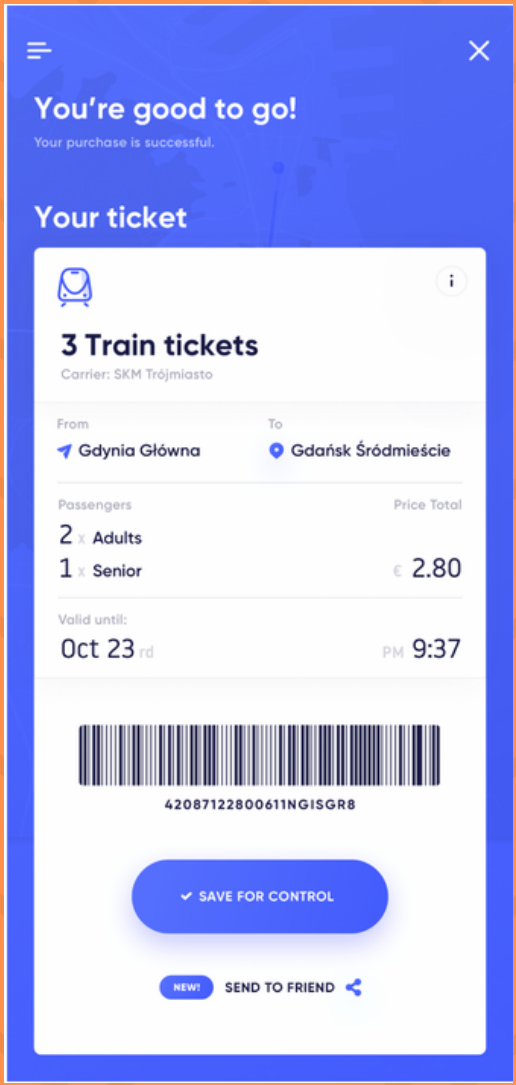
Pourquoi ?



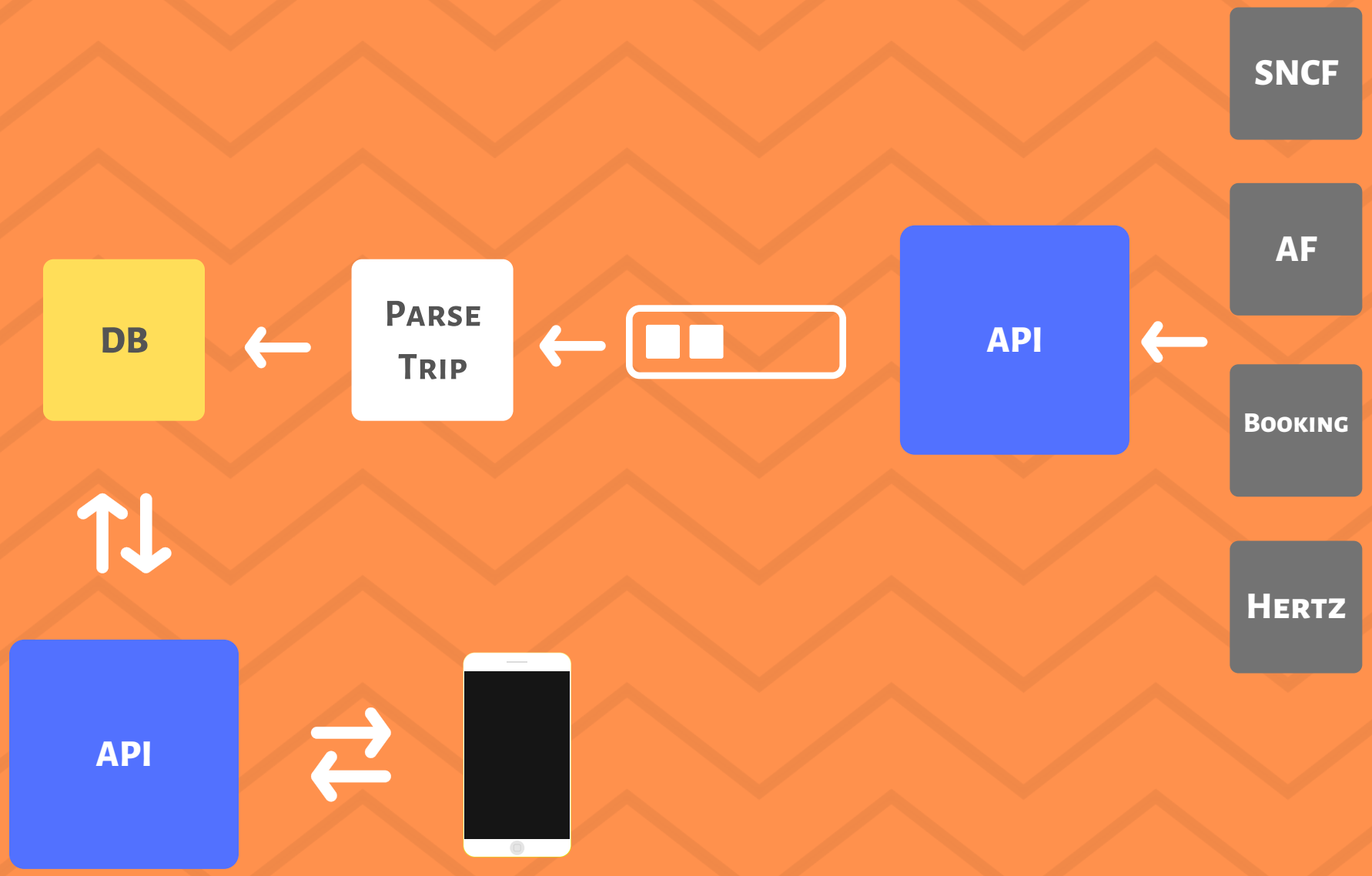
Pourquoi ?



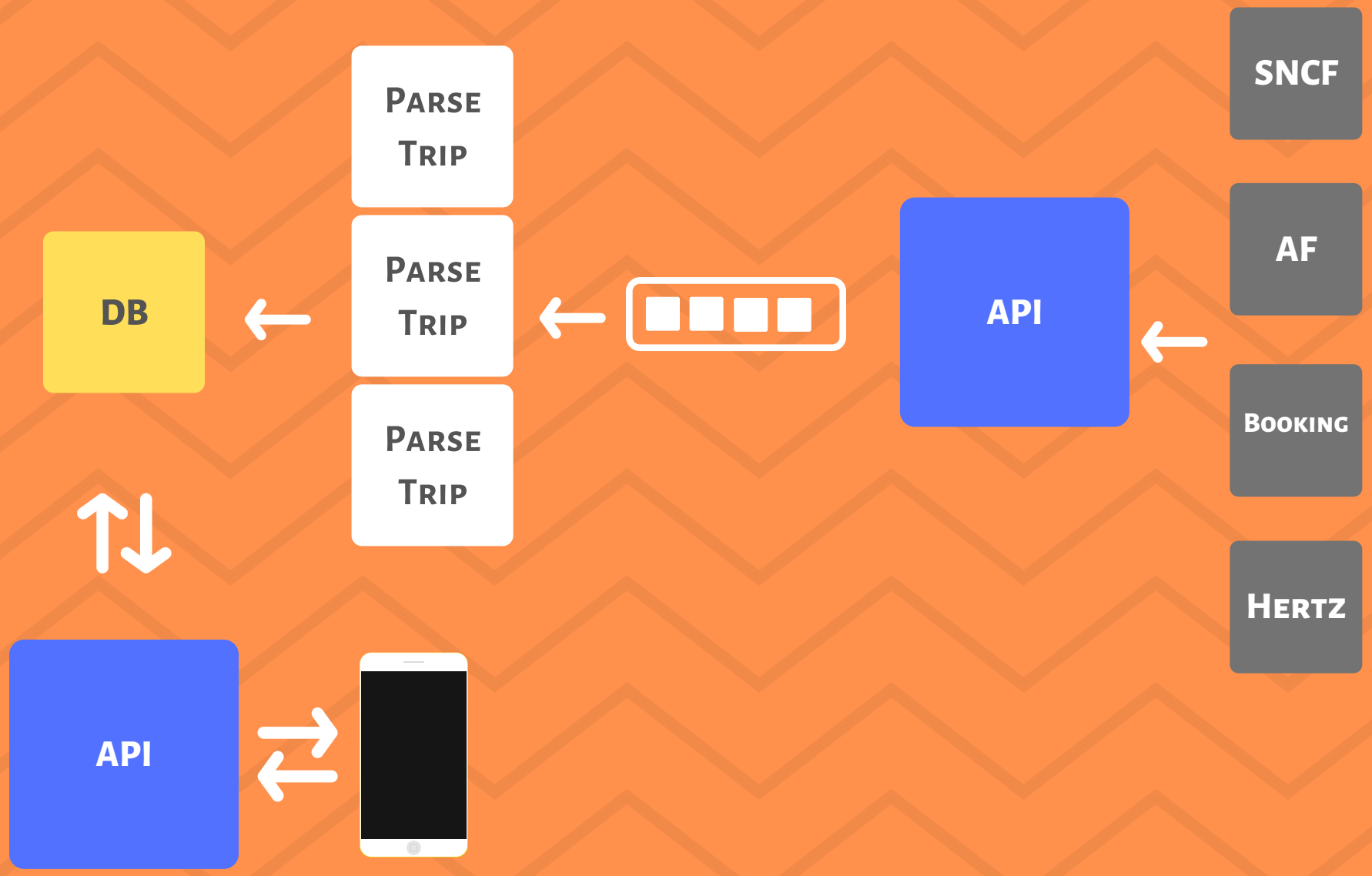
Pourquoi ?


You're good to go!
Your purchase is successful.
Your ticket
3 Train tickets
Carrier: SKM Trójmiasto
From
Gdynia Główna
To
Gdańsk Śródmieście
Passengers
Price Total
2 x Adults
1 x Senior
€ 2.80
Valid until:
Oct 23rd
PM 9:37
42087122800611NGISGR8
SAVE FOR CONTROL
NEW!
SEND TO FRIEND

Pourquoi ?



Pourquoi ?


SNCF
AF
Booking
Hertz
API
Parse Trip
Parse Trip
Parse Trip
DB
API

... Mais

Création d'une tâche

```
import time
from celery import shared_task


@shared_task(ignore_result=True)
def run_long_task():
    print('>>>>>> STARTING TASK')
    time.sleep(3)
    print('>>>>>> ENDING TASK')
```

Note sur les tests

Celery 4.2.0 documentation » User Guide »

previous | next | modules | index

This document describes the current stable version of Celery (4.2). For development docs, go here.

Testing with Celery

Tasks and unit tests

To test task behavior in unit tests the preferred method is mocking.

Eager mode:

The eager mode enabled by the `task_always_eager` setting is by definition not suitable for unit tests.

When testing with eager mode you are only testing an emulation of what happens in a worker, and there are many discrepancies between the emulation and what happens in reality.

A Celery task is much like a web view, in that it should only define how to perform the

Star 11,487

Please help support this community project with a donation:

Donate

Previous topic

Signals

Next topic

Obtenir le status d'une tâche

```
task.status
'PENDING'

task.status
'SUCCESS'
```

Retries

```
@shared_task(bind=True)
def run_long_task(self):
    try:
        raise Exception
    except Exception as _:
        self.retry()
```

Limite de temps

```
@shared_task(bind=True, time_limit=3)
def run_long_task(self):
    print('>>>>>> STARTING TASK')
    time.sleep(5)
    print('>>>>>> ENDING TASK')
```

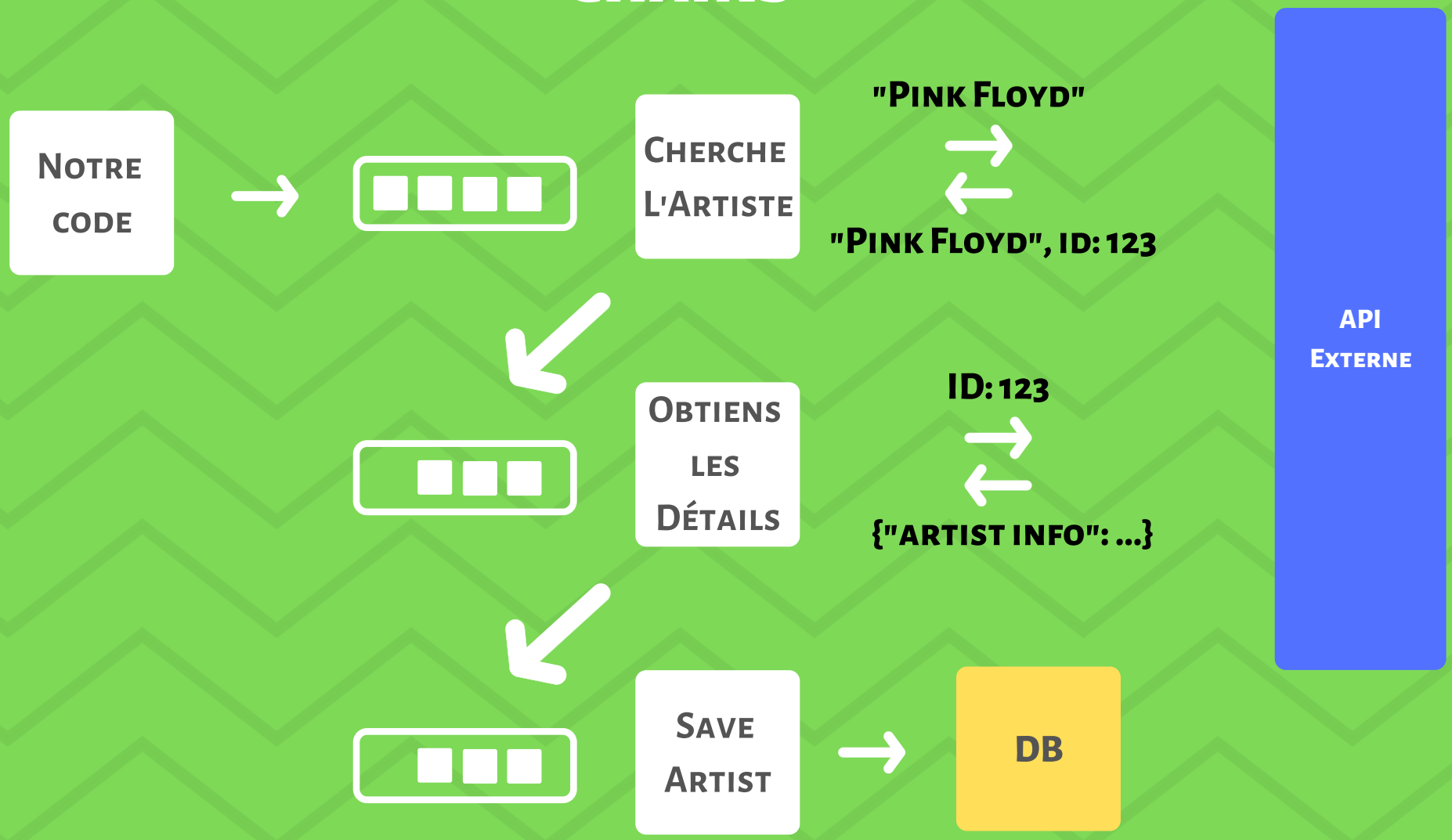
Héritage

```
class MeetupTask(Task):

    def __init__(self):
        print('++++++ INITIALIZING TASK')
        self.meetup_name = 'Python AFPY Lyon'
```

Chains

Chains


Notre code
Cherche L'Artiste
"Pink Floyd"
"Pink Floyd", id: 123
Obtiens les Détails
ID: 123
{"artist info": ...}
Save Artist
DB
API Externe

Chains

```
chain = add.s(2, 2) | mul.s(8) | mul.s(10)
chain.delay()
```

Tâches recurrentes

Autoscaling

Rate limits

Logging

Gestion des données sensibles

Tâches Synchrones

Monitoring (Flower)

Message Signing

Handlers

...

Merci

@alemangui