Voltaired

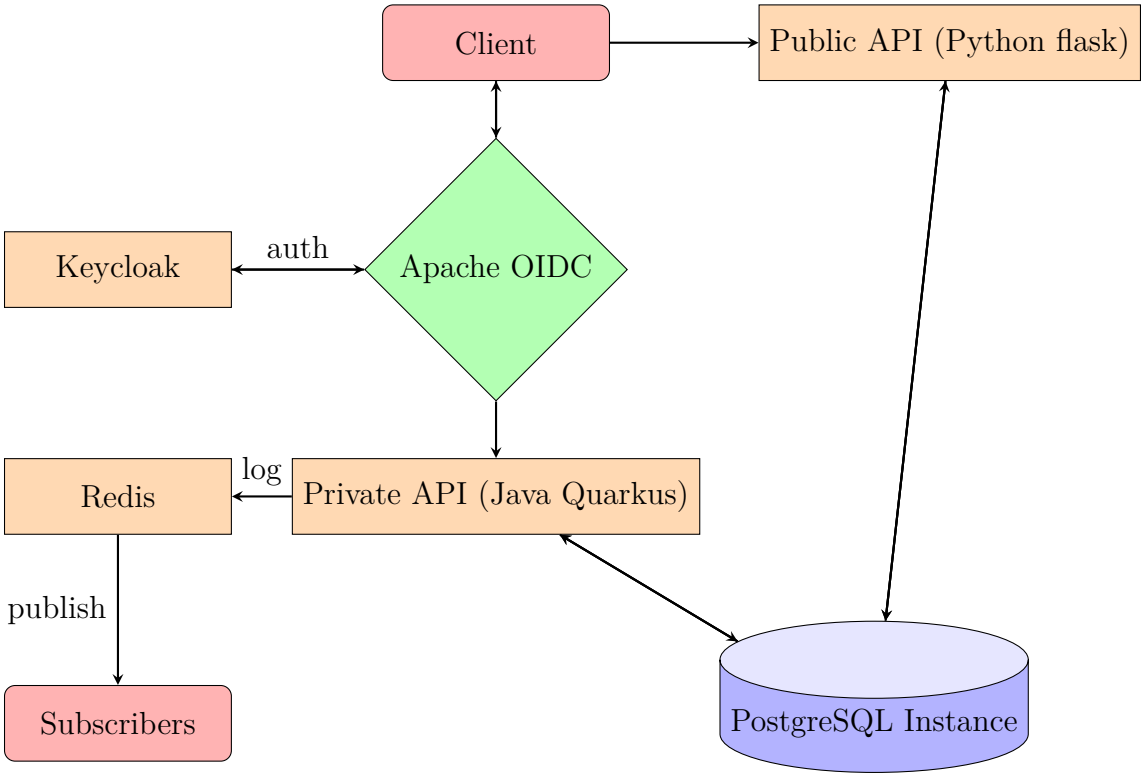
Rapport de projet SOA

Auteurs:
Chassignol Valentin
Desmarais Lowen
Giraud Lise
Guilbaud Mathis

Contents

1 Contexte

2 Architecture technique



3 Documentation

3.1 Installation

Tout d'abord, afin de pouvoir faire tourner notre application il est necessaire d'ajouter un host a votre DNS local, il faut que dps.epita.local redirige vers 127.0.0.1
Ensuite nous pouvons debuter l'installation.

```
git clone https://gitlab.cri.epita.fr/lowen.desmarais/voltaired.git
```

3.2 Deploiement

Pour deployer le projet:

```
cd voltaired
docker compose up -d
```

ensuite se connecter sur `https://dps.epita.local/`.
Maintenant vous etes assuré que le service tourne, des lors, pour acceder aux differentes application il suffit d'utiliser l'un des deux liens suivant:

- `https://dps.epita.local/java`
- `https://dps.epita.local/python`

4 Developpement

4.1 Structure de la base de donee:

la base de donnee a ete generee par quarkus a l'aide d'hibernate. Elle est structuree comme suit:

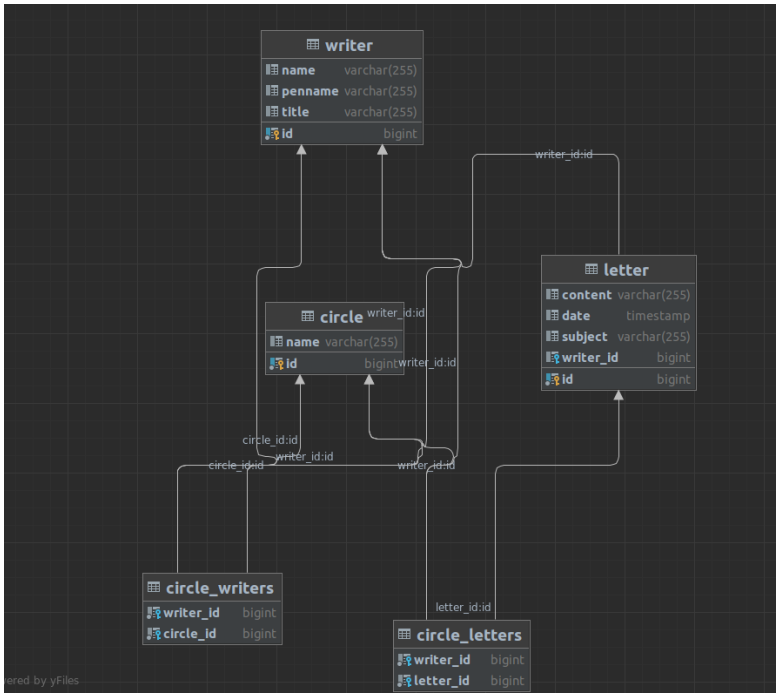


Figure 1: Structure de la DB

4.2 API Publique (Python flask)

Pour notre projet nous avons tout oriente autour d'un backend fait en java avec quarkus. Donc pour creer l'api publique en python il nous a fallu nous adapter au format de la db cree par hibernate pour pouvoir nous y connecter et faire ainsi toutes les requetes GET.
Pour ce faire, nous avons utilise Flask pour le coeur de l'API, ainsi que SQLAlchemy en tant qu'ORM. La representation de la DB pour SQLAlchemy est structuree comme suit:

```
from app import db

class Circle(db.Model):
    __tablename__ = 'circle'
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(255))

    def __repr__(self):
        return "<Circle␣{}>".format(self.id)

class Letter(db.Model):
    __tablename__ = 'letter'
    id = db.Column(db.Integer, primary_key=True)
    date = db.Column(db.DateTime)
    subject = db.Column(db.String(255))
    content = db.Column(db.String)
    writer_id = db.Column(db.Integer, db.ForeignKey('writer.id'))
    circle_id = db.Column(db.Integer, db.ForeignKey('circle.id'))
    circle = db.relationship(Circle, backref='letters')

class Writer(db.Model):
    __tablename__ = 'writer'
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(255))
    penname = db.Column(db.String(255))
    title = db.Column(db.String(255))
    letters = db.relationship(Letter, backref='writer')
    circles = db.relationship(Circle, secondary='circle_writers')

class CircleWriters(db.Model):
    __tablename__ = 'circle_writers'
    circle_id = db.Column(db.Integer, db.ForeignKey('circle.id'),
   primary_key=True)
    writer_id = db.Column(db.Integer, db.ForeignKey('writer.id'),
   primary_key=True)

class CircleLetters(db.Model):
    __tablename__ = 'circle_letters'
    letter_id = db.Column(db.Integer, db.ForeignKey('letter.id'),
   primary_key=True)
    writer_id = db.Column(db.Integer, db.ForeignKey('writer.id'),
   primary_key=True)
```