Synthese d'Activite - Projet ThrottleX

Module IDV-AQL5 - Qualite du Code

Date : Fevrier 2026
Equipe : - Nassim Bouziane - Thomas Boulard - Abdelkoudousse Boustani

---

1. Presentation du Projet

1.1 Objectif

Developper un service de rate limiting et quotas multi-tenant pour proteger des APIs d'inference. Le service permet de limiter le nombre de requetes par client selon des politiques configurables.

1.2 Livrables Produits

Livrable	Description	Statut
API REST	Endpoints de gestion des politiques et d'evaluation	Termine
Tests unitaires	Suite pytest avec couverture > 80%	Termine
Tests de charge	Scripts k6 pour benchmarks	Termine
Pipeline CI/CD	GitLab CI avec quality gates	Termine
Documentation	OpenAPI, ADR, guides techniques	Termine

---

2. Organisation de l'Equipe

2.1 Repartition des Roles

Nous avons organise notre equipe selon trois roles principaux, chacun ayant des responsabilites clairement definies.

Membre	Role Principal	Responsabilites
Nassim Bouziane	Tech Lead / DevOps	Architecture, decisions techniques, CI/CD, revue de code
Thomas Boulard	Backend Developer	Implementation metier, tests unitaires, documentation
Abdelkoudousse Boustani	QA Engineer	API FastAPI, tests de charge, tests de proprietes

2.2 Description des Roles

Nassim Bouziane - Tech Lead / DevOps

En tant que Tech Lead, Nassim a pris en charge les decisions architecturales majeures du projet. Il a redige les ADR (Architecture Decision Records) documentant les choix techniques importants, notamment la selection de l'architecture stateless avec Redis. Il a egalement mis en place l'infrastructure CI/CD avec GitLab CI, configure les quality gates, et assure les revues de code pour maintenir la qualite du codebase.

Thomas Boulard - Backend Developer

Thomas s'est concentre sur l'implementation de la logique metier dans le module service.py. Il a developpe les algorithmes de rate limiting (Sliding Window et Token Bucket) et ecrit les tests unitaires correspondants. Il a egalement pris en charge la redaction de la documentation technique et des guides d'utilisation.

Abdelkoudousse Boustani - QA Engineer

Abdelkoudousse a implemente l'API REST avec FastAPI, en creant les endpoints conformes a la specification OpenAPI fournie. Il a egalement concu et execute les tests de charge avec k6, analyse les resultats de performance, et developpe les tests de proprietes avec la librairie Hypothesis pour garantir les invariants du systeme.

---

3. Planning et Deroulement

3.1 Phases du Projet

Le projet s'est deroule sur plusieurs semaines, divise en phases distinctes.

Phase	Duree	Activites
Phase 1 - Analyse	1 semaine	Etude du cahier des charges, analyse des options d'architecture
Phase 2 - Conception	1 semaine	Choix techniques, ADR, definition de l'architecture
Phase 3 - Implementation	2 semaines	Developpement de l'API, integration Redis, algorithmes
Phase 4 - Tests	1 semaine	Tests unitaires, tests d'integration, tests de charge
Phase 5 - Industrialisation	1 semaine	CI/CD, documentation, optimisations

3.2 Diagramme de Gantt Simplifie

```
Semaine 1  |===== Analyse =====|
Semaine 2  |===== Conception =====|
Semaine 3  |========== Implementation ==========|
Semaine 4  |========== Implementation ==========|
Semaine 5  |===== Tests =====|
Semaine 6  |===== Industrialisation =====|
```

3.3 Reunions et Communication

Nous avons mis en place une organisation agile legere : - Stand-up quotidien de 15 minutes (Discord) - Revue de sprint hebdomadaire - Retrospective en fin de phase - Canal Discord dedie pour les echanges asynchrones

---

4. Contributions Detaillees

4.1 Nassim Bouziane

Tache	Temps	Description
Architecture Redis	3h	Conception du schema de stockage, scripts Lua
Repository	4h	Implementation de repository.py avec operations atomiques
CI/CD GitLab	3h	Configuration du pipeline, stages, quality gates
ADR	2h	Redaction des documents de decision d'architecture
Revue de code	3h	Review des pull requests, feedback
Total	**15h**	

Contributions majeures : - Choix de l'architecture stateless + Redis documente dans ADR-001 - Scripts Lua pour garantir l'atomicite des operations de comptage - Configuration complete de la pipeline GitLab CI avec 4 stages - Mise en place des quality gates bloquants (couverture 80%, 0 vulnerabilite HIGH)

4.2 Thomas Boulard

Tache	Temps	Description
Service metier	4h	Implementation de service.py
Token Bucket	3h	Algorithme Token Bucket dans algorithms/
Tests unitaires	4h	Tests pytest pour service et models
Documentation	2h	README, guides techniques

Tache	Temps	Description
Modeles Pydantic	2h	Definition des schemas de donnees
Total	**15h**	

Contributions majeures : - Logique metier complete dans service.py - Implementation de l'algorithme Token Bucket avec gestion du burst - 25 tests unitaires couvrant les cas nominaux et les erreurs - Documentation utilisateur et technique

4.3 Abdelkoudousse Boustani

Tache	Temps	Description
API FastAPI	4h	Implementation de app.py, endpoints REST
Tests k6	3h	Scripts de benchmark, analyse des resultats
Tests proprietes	3h	Tests Hypothesis pour invariants
Integration	2h	Tests d'integration avec Redis reel
Metriques	2h	Exposition Prometheus dans metrics.py
Total	**14h**	

Contributions majeures : - Endpoints REST conformes a la specification OpenAPI - Script k6 simulant 100 utilisateurs virtuels sur 80 secondes - Tests de proprietes verifiant l'isolation multi-tenant - Metriques Prometheus pour le monitoring en production

---

5. Bilan Technique

5.1 Metriques du Projet

Metrique	Valeur
Lignes de code Python	~800
Nombre de fichiers source	12
Tests unitaires	30+
Couverture de code	82%
Temps d'execution tests	8 secondes

5.2 Resultats des Benchmarks

Les tests de charge realises avec k6 ont demontre les performances suivantes :

Metrique	Resultat	Objectif
Latence P95	64.83 ms	< 100 ms
Latence P99	87.28 ms	< 200 ms
Throughput	668 req/s	> 500 req/s

Metrique	Resultat	Objectif
Taux de blocage	98.31%	Attendu

5.3 Points Forts

- Architecture scalable et resiliente grace a Redis
- Atomicite des operations garantie par scripts Lua
- Pipeline CI/CD complete avec quality gates
- Documentation technique exhaustive
- Tests de proprietes pour garantir les invariants

5.4 Difficultes Rencontrees

Difficulte	Impact	Resolution
Race conditions Redis	Moyen	Scripts Lua atomiques
Tests async pytest	Faible	Plugin pytest-asyncio
Couverture 80%	Faible	Focus sur chemins critiques
Port 8000 occupe (erreur windows pas clair)	Faible	Documentation de debug

---

6. Retrospective

6.1 Ce qui a Bien Fonctionne

- **Communication** : Les stand-ups quotidiens ont permis de detecter rapidement les blocages
- **Repartition claire** : Chaque membre avait des responsabilites bien definies, evitant les conflits
- **Revues de code** : Les reviews systematiques ont ameliore la qualite globale
- **Tests automatises** : La CI/CD a detecte plusieurs regressions avant merge

6.2 Ce qui Pourrait Etre Ameliore

- **Estimation des temps** : Certaines taches ont pris plus de temps que prevu (notamment les scripts Lua)
- **Documentation** : Nous aurions pu documenter plus tot pour eviter la surcharge en fin de projet
- **Pair programming** : Nous aurions pu faire plus de sessions de programmation en binome sur les parties complexes

6.3 Enseignements

Ce projet nous a permis de mettre en pratique plusieurs concepts importants : - L'importance des tests automatises dans un workflow professionnel - La valeur des quality

gates pour maintenir la qualite du code - L'interet des ADR pour tracer les decisions architecturales - Les techniques de rate limiting utilisees en production

---

7. Conclusion

Le projet ThrottleX a ete mene a bien par notre equipe de trois personnes. Tous les livrables ont ete produits conformement au cahier des charges :

- API REST fonctionnelle avec rate limiting multi-tenant
- Couverture de tests superieure a 80%
- Pipeline CI/CD avec quality gates bloquants
- Documentation technique complete

La repartition des roles entre Tech Lead, Backend Developer et QA Engineer a permis une organisation efficace et une montee en competence de chaque membre sur son domaine d'expertise.

Le temps total investi par l'equipe est d'environ 44 heures, reparti equitablement entre les trois membres.

Annexe : Tableau Recapitulatif des Contributions

Membre	Role	Fichiers Principaux	Commits	Heures
Nassim Bouziane	Tech Lead / DevOps	repository.py, .gitlab-ci.yml	25	15h
Thomas Boulard	Backend Developer	service.py, token_bucket.py, tests/	22	15h
Abdelkoudousse Boustani	QA Engineer	app.py, k6/, test_properties.py	20	14h
Total			**67**	**44h**