Présentation du projet : Imagine AI

L'objectif du projet est de développer une application de génération d'histoires et de comptines personnalisées pour enfants, basée sur l'Intelligence Artificielle, à destination des parents et de leurs enfants.

L'application permet au parent de fournir différents éléments créatifs tels que :

- un thème
- un ou plusieurs personnages
- une morale
- une émotion
- une situation du quotidien ou imaginaire

À partir de ces éléments, l'IA génère une histoire adaptée à l'âge de l'enfant, dans un univers bienveillant et enfantin.

L'histoire reste entièrement personnalisable : le parent peut ajuster le prompt, demander une nouvelle version, modifier la morale, le ton ou la longueur de l'histoire.

L'enfant est encouragé à participer à la création (choix des personnages, de l'émotion, de la fin…), favorisant :

- le développement de l'imagination
- l'expression émotionnelle
- un moment de complicité et d'échange avec le parent
- Le parent peut poster son histoire et se connecter

Potentiel d'évolution

- Génération automatique d'illustrations associées à l'histoire
- Mise en page du récit
- Impression physique via un partenariat avec des imprimeries
- Création de cadeaux personnalisés (livres, carnets, affiches)

- Voix off / narration audio pour les enfants en text to speech.
- Bibliothèque d’histoires collaboratives parent-enfant

Exigences fonctionnelles

- Génération d’histoires et de comptines personnalisées via IA
- Interface simple et intuitive (utilisateur non expérimenté)
- Sauvegarde des histoires générées
- Consultation de l’historique des histoires
- Possibilité de :
    - régénérer une histoire
    - modifier les prompts existants
    - améliorer ou enrichir une histoire déjà créée
- Gestion de différents paramètres :
    - âge de l’enfant
    - longueur de l’histoire
    - ton (joyeux, rassurant, drôle, éducatif)

Exigences non fonctionnelles

Sécurité et éthique

- Respect de l’AI Act et des règles de protection des mineurs
- Contenu strictement :
    - adapté aux enfants
    - sans violence, propos choquants ou inappropriés
- Mécanismes de contrôle pour limiter les hallucinations de l’IA

- Filtrage sémantique des réponses générées

Performance et technique

- Gestion du nombre de tokens consommés
- Limitation de la longueur des histoires (pages / mots)
- Optimisation du temps de génération
- Gestion de la latence des réponses IA
- Scalabilité pour supporter plusieurs utilisateurs simultanés

User Stories

US 1 – Génération simple

En tant que parent non expérimenté,
je souhaite pouvoir interagir facilement avec l'IA,
afin de générer rapidement une histoire personnalisée pour mon enfant,
sans avoir de connaissances techniques.

US 2 – Personnalisation

En tant que parent,
je souhaite modifier ou régénérer une histoire existante,
afin de l'adapter aux envies et à l'imagination de mon enfant.

US 3 – Participation de l'enfant

En tant qu'enfant,
je souhaite pouvoir choisir des éléments de l'histoire (personnage, émotion, fin),
afin de participer activement à la création du récit.

US 4 – Conservation

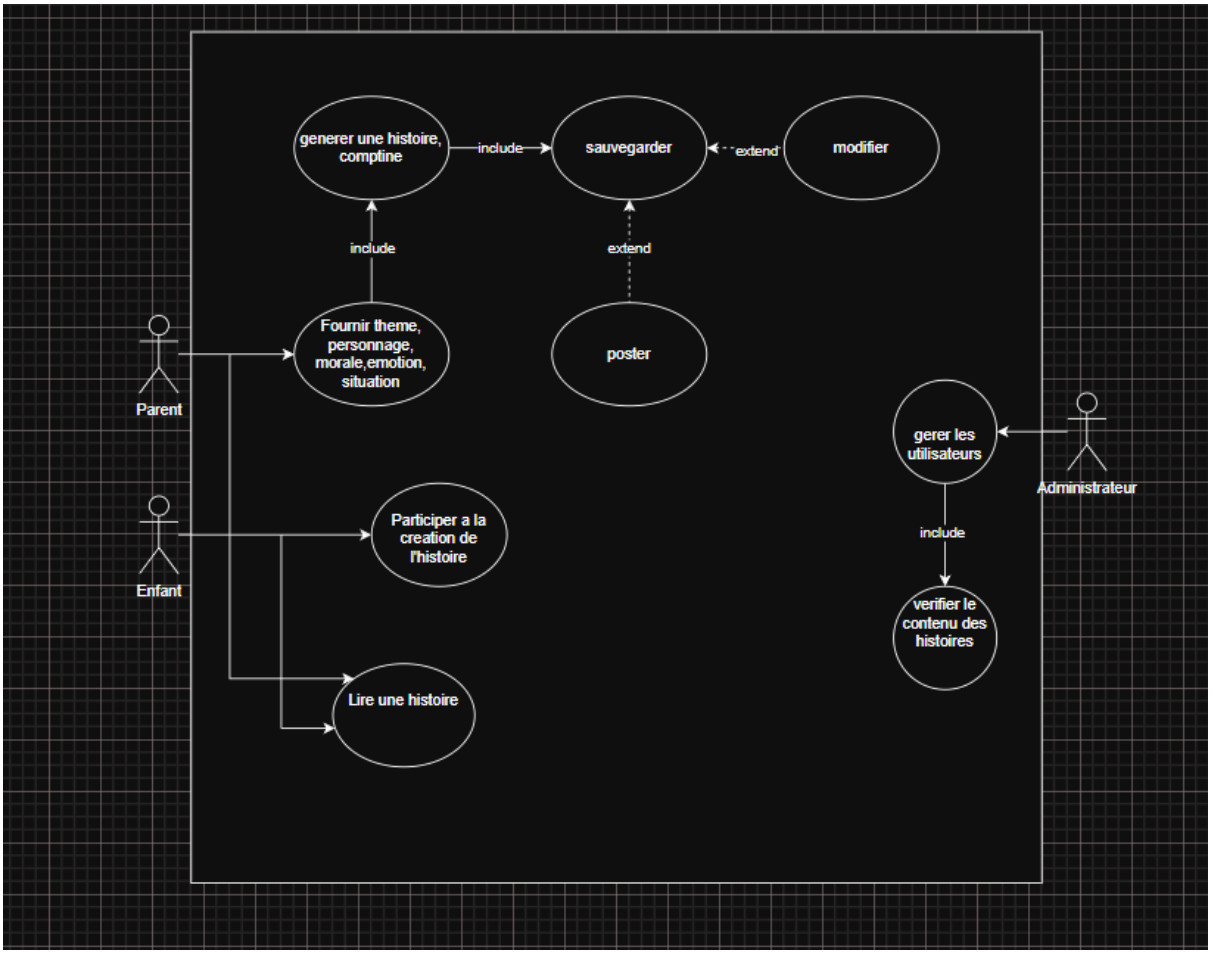
En tant que parent,
je souhaite conserver les histoires générées,
afin de pouvoir les relire, les modifier ou les offrir plus tard.

- Périmètre MVP — Application web responsive, authentification, profils parent/enfants, génération IA (texte), bibliothèque privée, publication et feed public, likes, signalement, export PDF, modération admin.

- Hors périmètre — Génération d’illustrations, narration audio, impression physique, application mobile native.

Diagramme UML:

Use Case :



Classe:

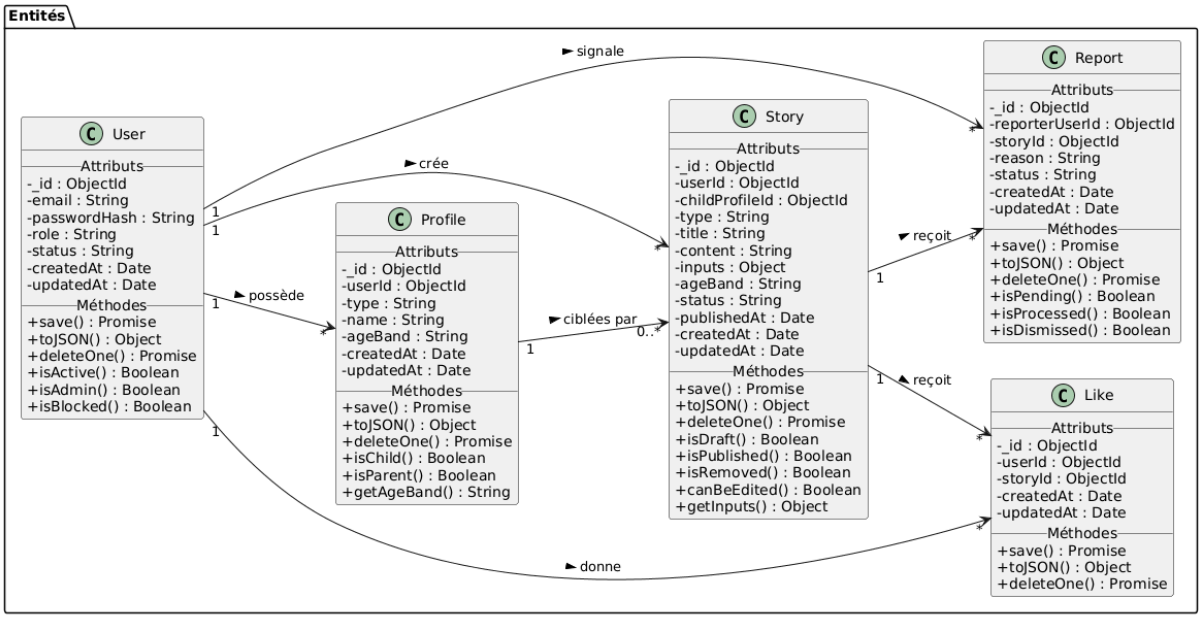
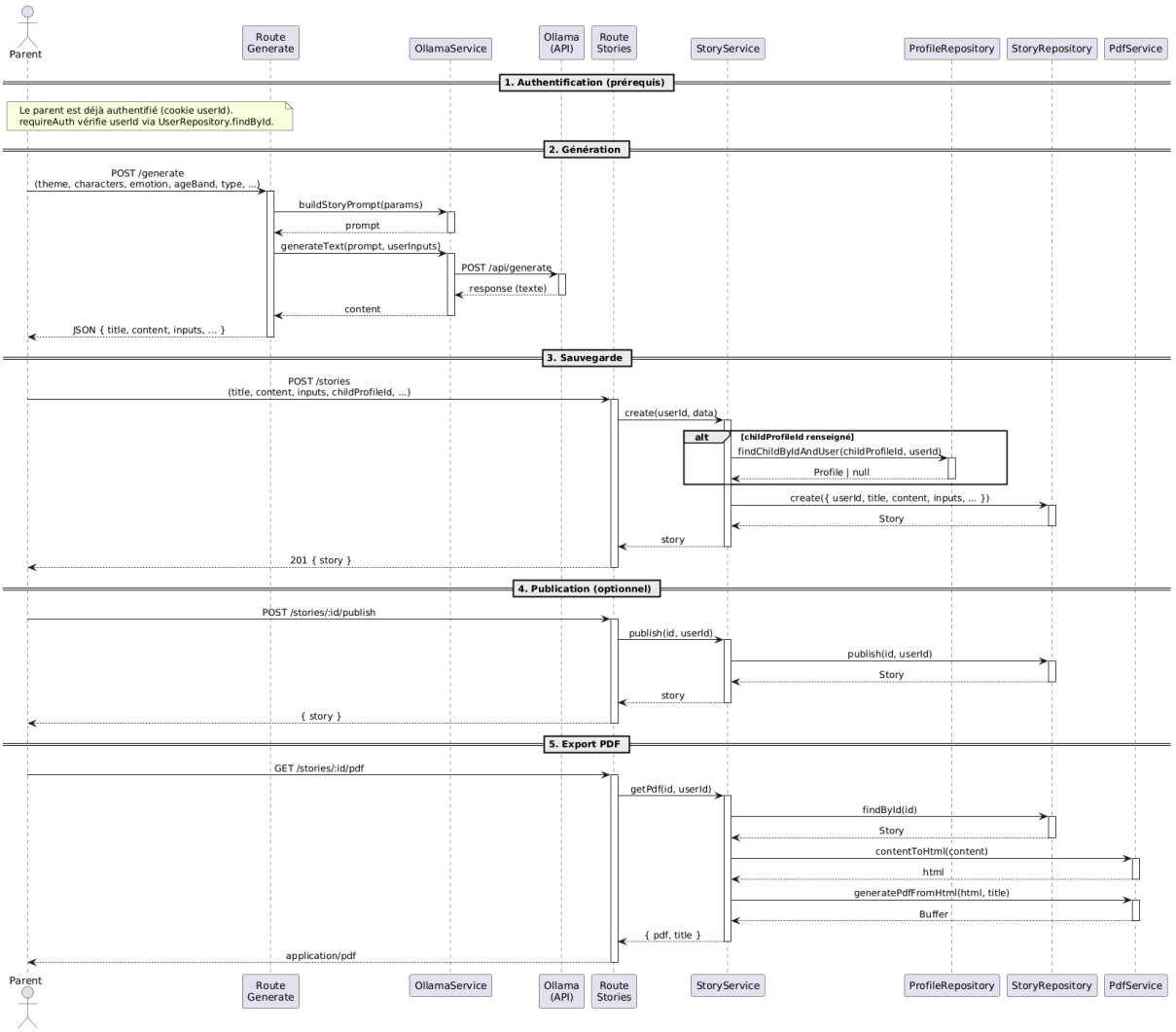
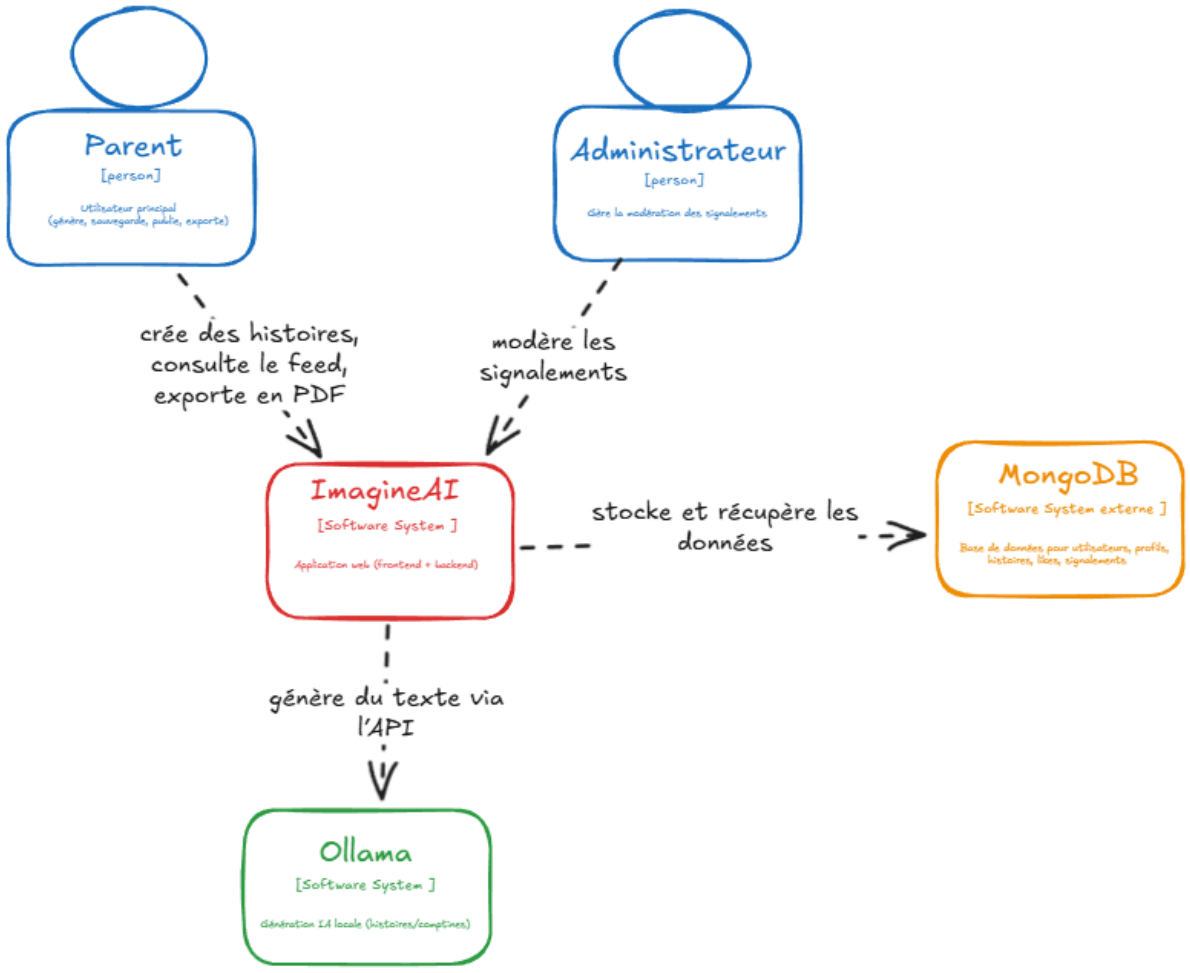


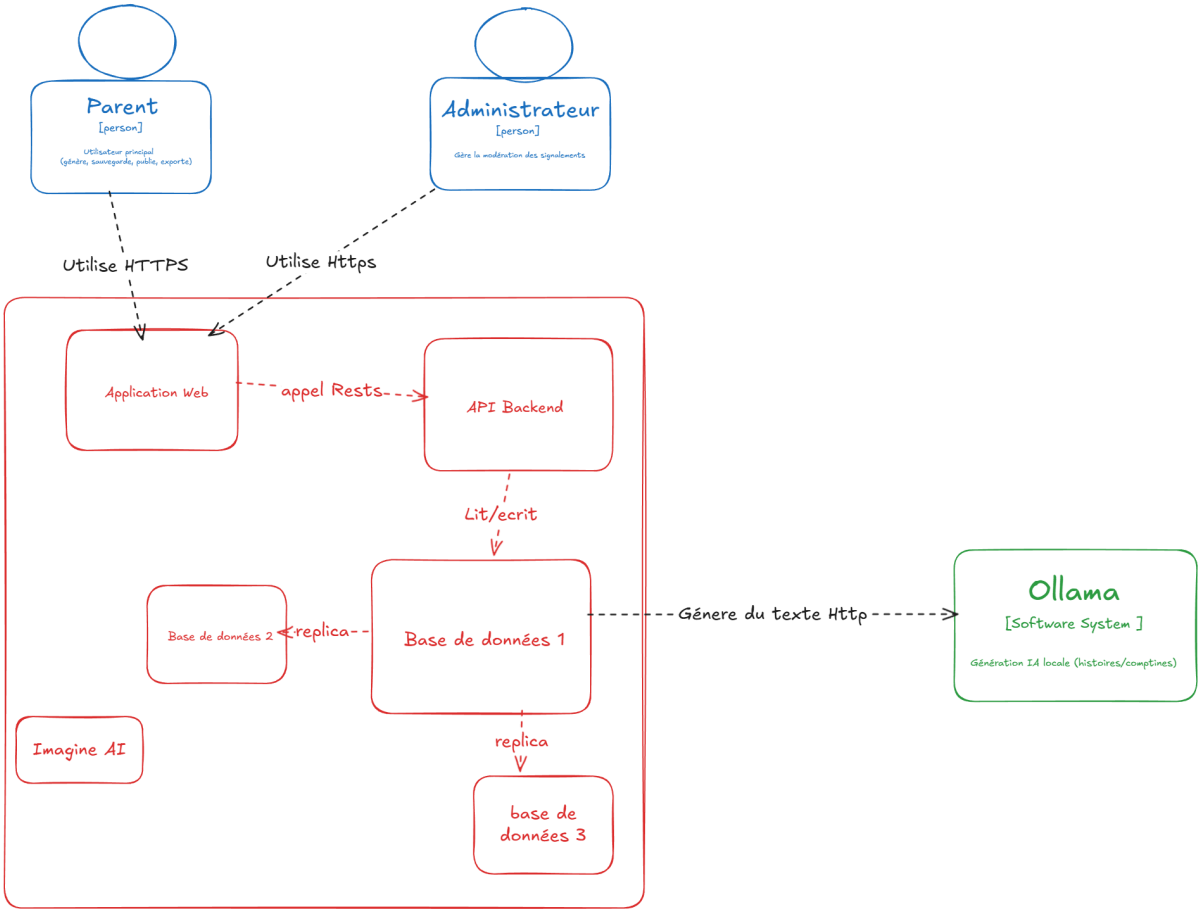
Diagramme de séquence:



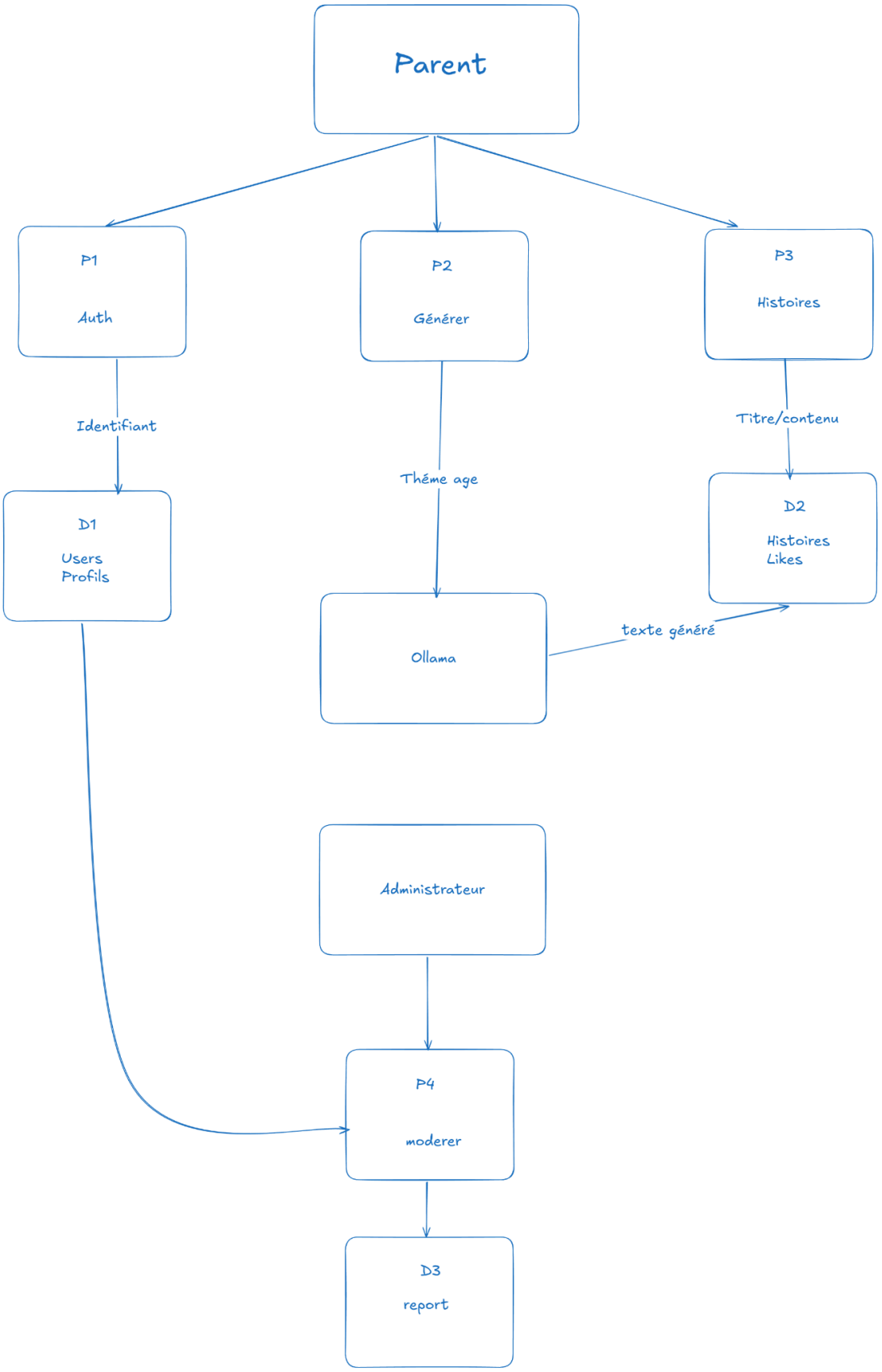
Schémas C4 contexte :



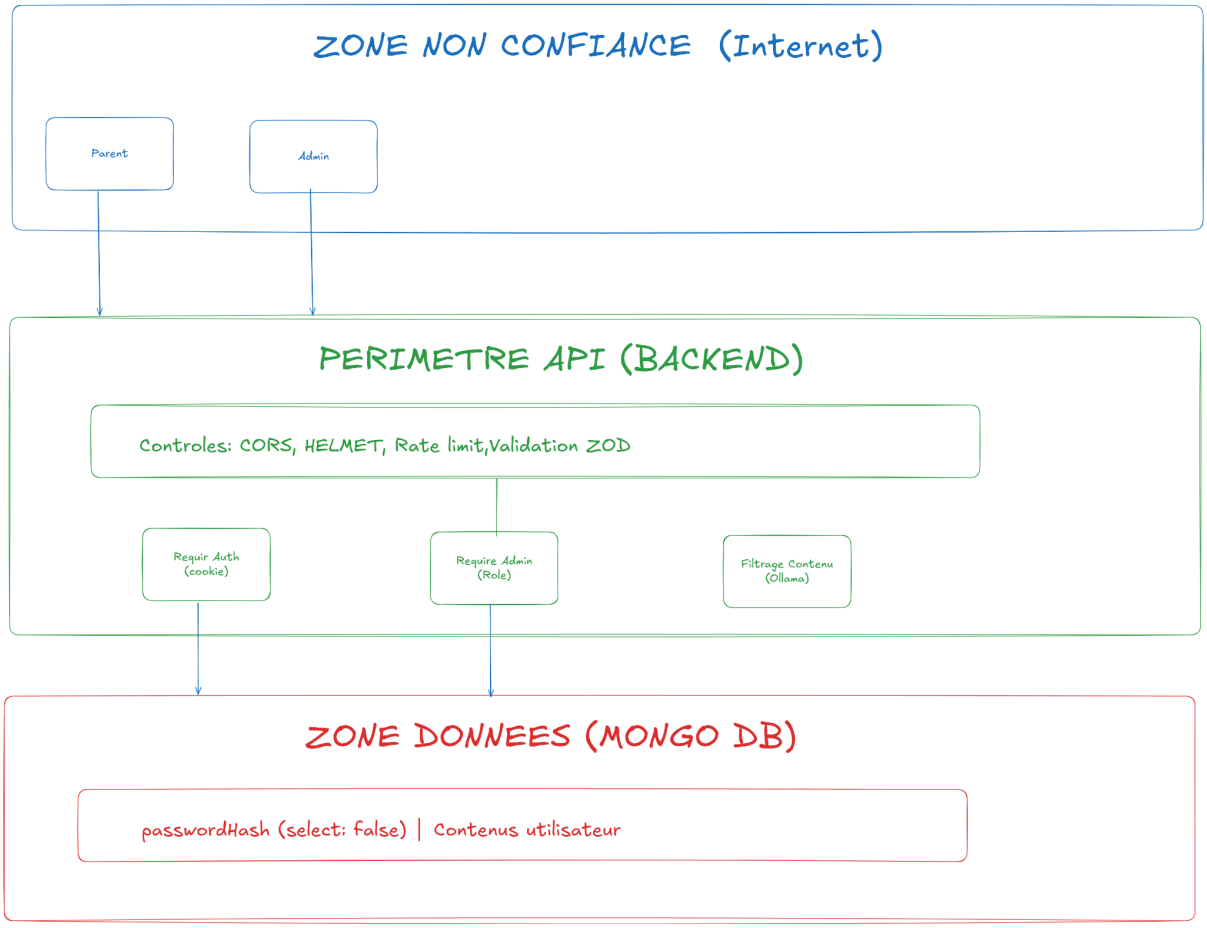
Schémas C4 Container:



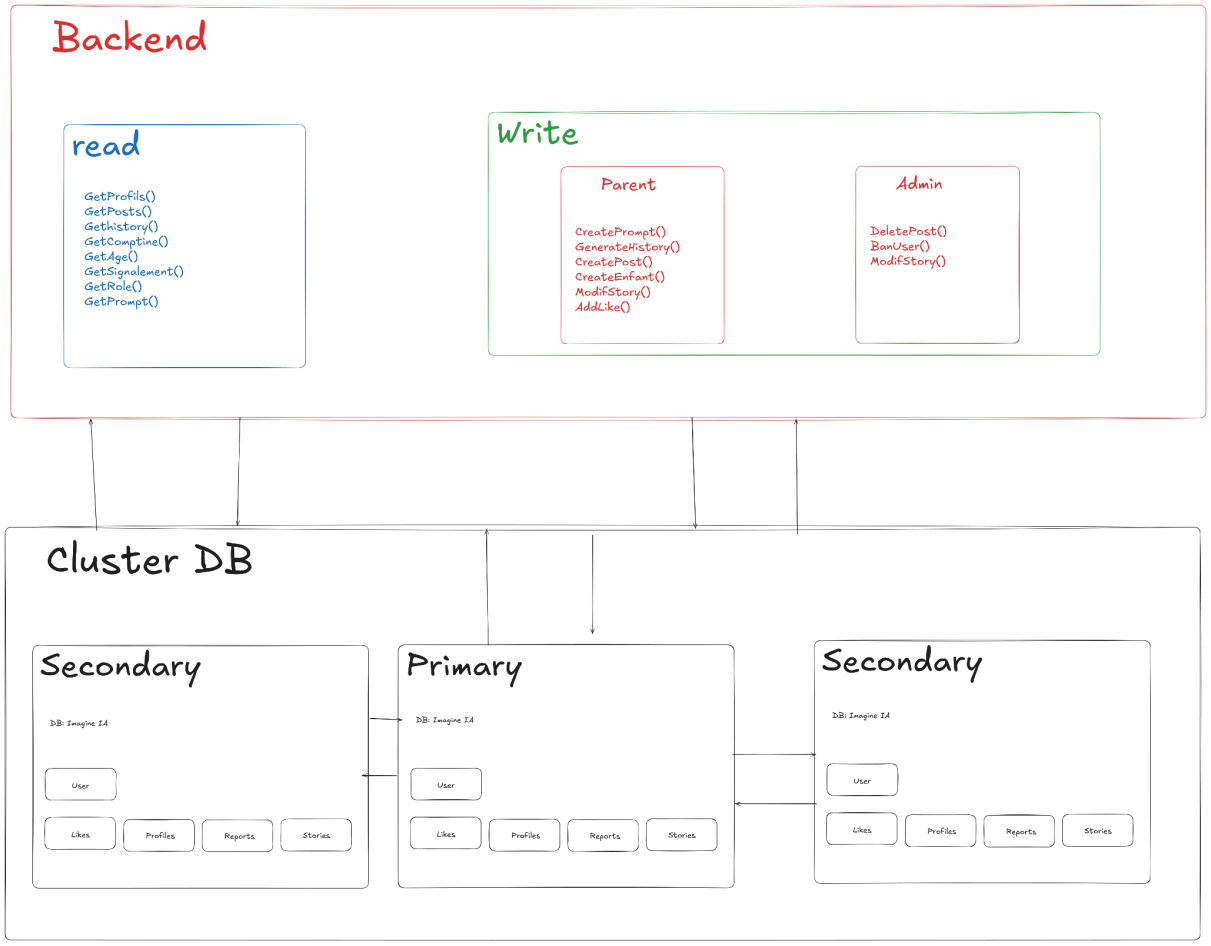
Schémas de données:


Parent
P1
Auth
P2
Générer
P3
Histoires
Identifiant
Théme age
Titre/contenu
D1
Users
Profils
D2
Histoires
Likes
Ollama
texte généré
Administrateur
P4
moderer
D3
report

Schémas de sécurité :



Schémas cluster Mongodb:



Verification replication:

```
PS C:\Users\erwan> docker exec -it mongodb-0 mongosh --eval "rs.status().members.map(m => ({name:m.name, state:m.stateStr, health:m.health
  { name: 'mongodb-0:27017', state: 'PRIMARY', health: 1 },
  { name: 'mongodb-1:27017', state: 'SECONDARY', health: 1 },
  { name: 'mongodb-2:27017', state: 'SECONDARY', health: 1 }
]
```

test d'écriture sur le primary:

```
PS C:\Users\erwan> docker exec -it mongodb-0 mongosh "mongodb://mongodb-0:27017,mongodb-1:27017,mongodb-2:27017/imagine-ai?replicaSet=rs0"
 --eval "
>> db.test_cluster.insertOne({msg:'initial write', ts:new Date()})
>> "
>>
{
  acknowledged: true,
  insertedId: ObjectId('698df4ed29c1bef6368ce5b0')
}
```

Vérifier la réplication en lisant sur un SECONDARY:

```
PS C:\Users\erwan> docker exec -it mongodb-0 mongosh "mongodb://mongodb-0:27017,mongodb-1:27017,mongodb-2:27017/imagine-ai?replicaSet=rs0"
 --eval "
>> db.test_cluster.insertOne({msg:'initial write', ts:new Date()})
>> "
>>
{
  acknowledged: true,
  insertedId: ObjectId('698df4ed29c1bef6368ce5b0')
}

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image →docker debug mongodb-0
    Learn more at https://docs.docker.com/go/debug-cli/
[
  {
    _id: ObjectId('698df4ed29c1bef6368ce5b0'),
    msg: 'initial write',
    ts: ISODate('2026-02-12T15:42:37.545Z')
  }
```

Simulation FAILOVER : arrêter le PRIMARY et verifier le nouveau

```
PS C:\Users\erwan> docker stop mongodb-0
>>
mongodb-0
PS C:\Users\erwan> docker exec -it mongodb-1 mongosh --eval "rs.status().members.map(m => ({name:m.name, state:m.stateStr, health:m.health
}))"
>>
[
  {
    name: 'mongodb-0:27017',
    state: '(not reachable/healthy)',
    health: 0
  },
  { name: 'mongodb-1:27017', state: 'PRIMARY', health: 1 },
  { name: 'mongodb-2:27017', state: 'SECONDARY', health: 1 }
]

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image →docker debug mongodb-1
    Learn more at https://docs.docker.com/go/debug-cli/
```

Passage des tests:

```
PS C:\Users\erwan\OneDrive\Documents\SLAM\imagine_AI\backend> npm run test

> imagine-ai-backend@1.0.0 test
> vitest run


 RUN  v4.0.18 C:/Users/erwan/OneDrive/Documents/SLAM/imagine_AI/backend

 ✓ src/utils/validate.test.ts (1 test) 15ms
 ✓ src/services/ollamaService.test.ts (2 tests) 15ms
stderr | tests/integration/auth.integration.test.ts
Using NodeJS below 20.19.0

 ✓ tests/integration/health.integration.test.ts (1 test) 103ms
 ✓ tests/integration/auth.integration.test.ts (1 test) 4202ms
     ✓ crée un utilisateur et retourne 201  468ms

 Test Files  4 passed (4)
      Tests  5 passed (5)
   Start at  17:02:33
   Duration  7.52s (transform 1.25s, setup 0ms, import 6.92s, tests 4.33s, environment 2ms)
```