



OAR Rust Redux

Février 2026

🔍 Annexes

Salmane Amine
Er-Rami Moataz
Amessegher Aymane
Memil Dila

Contexte et objectif du projet

- **Epurer l'interface graphique actuelle**
 - Développer une interface similaire aux diagramme de gant sur Grid5000
 - Visualiser des statistiques
- **Améliorer l'intégration avec OAR grâce à la mise en place d'une API développée en Rust.**
 - Architecture modulaire et découplage fonctionnel
 - Alignement métier
 - Fiabilité du contrat d'interface
 - Sécurisation transverse et gestion des accès

Carte réseaux Grid5000

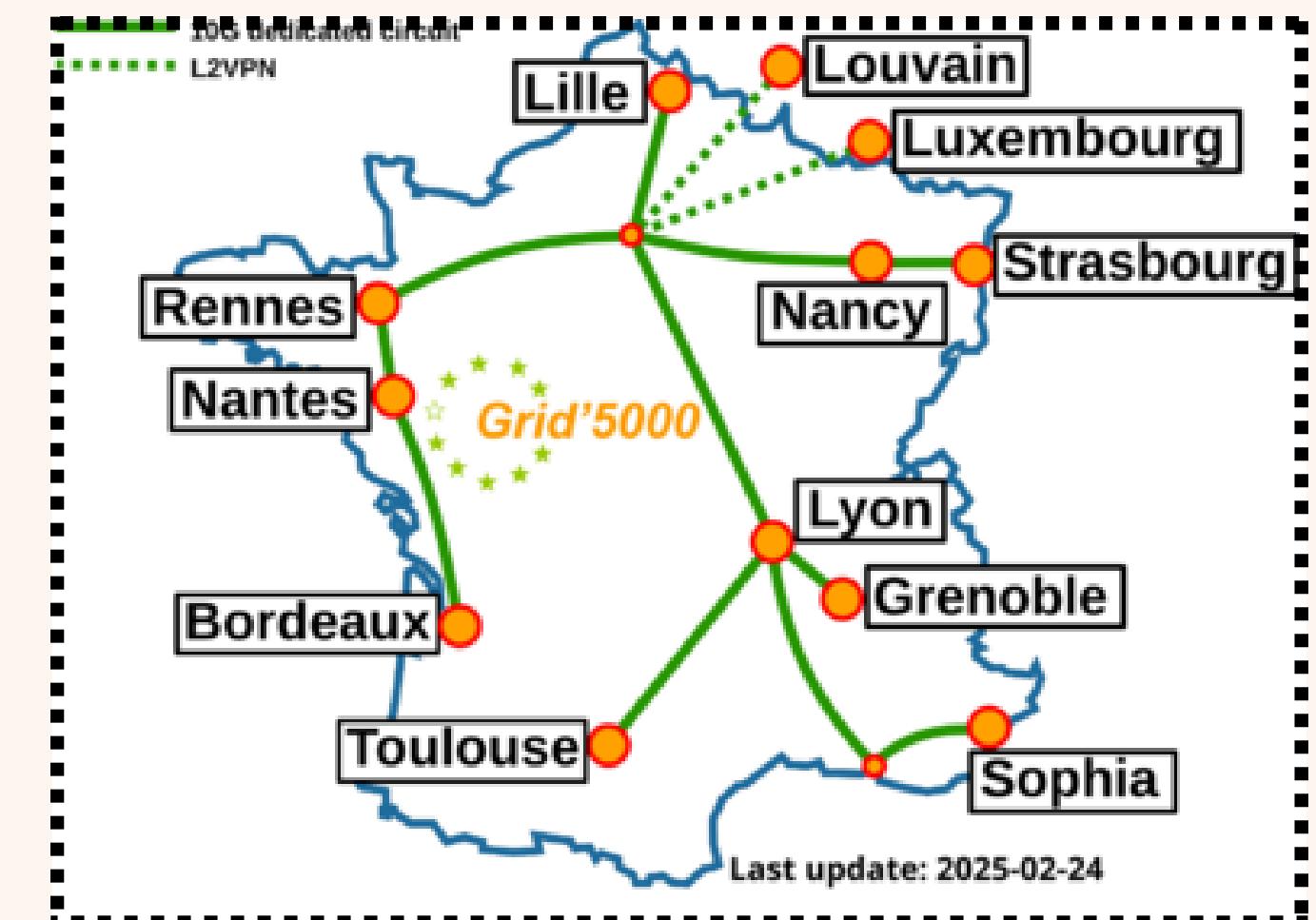
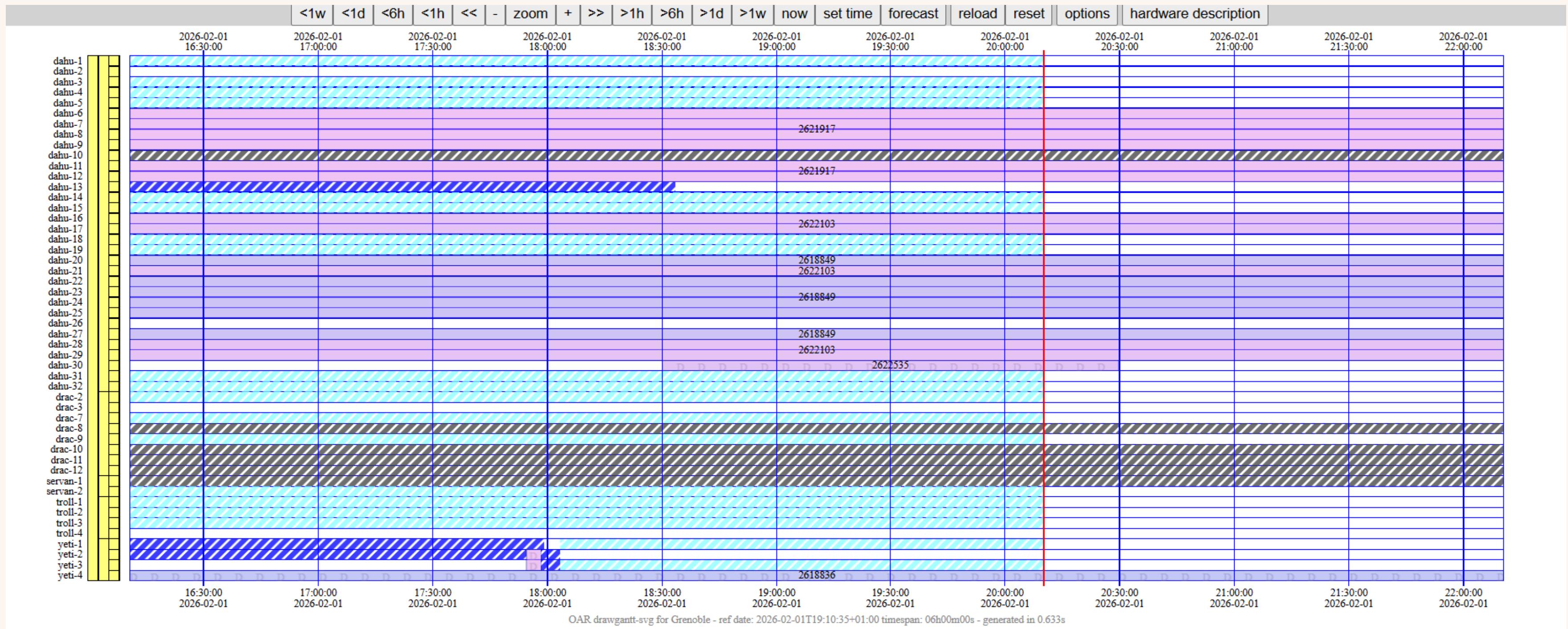
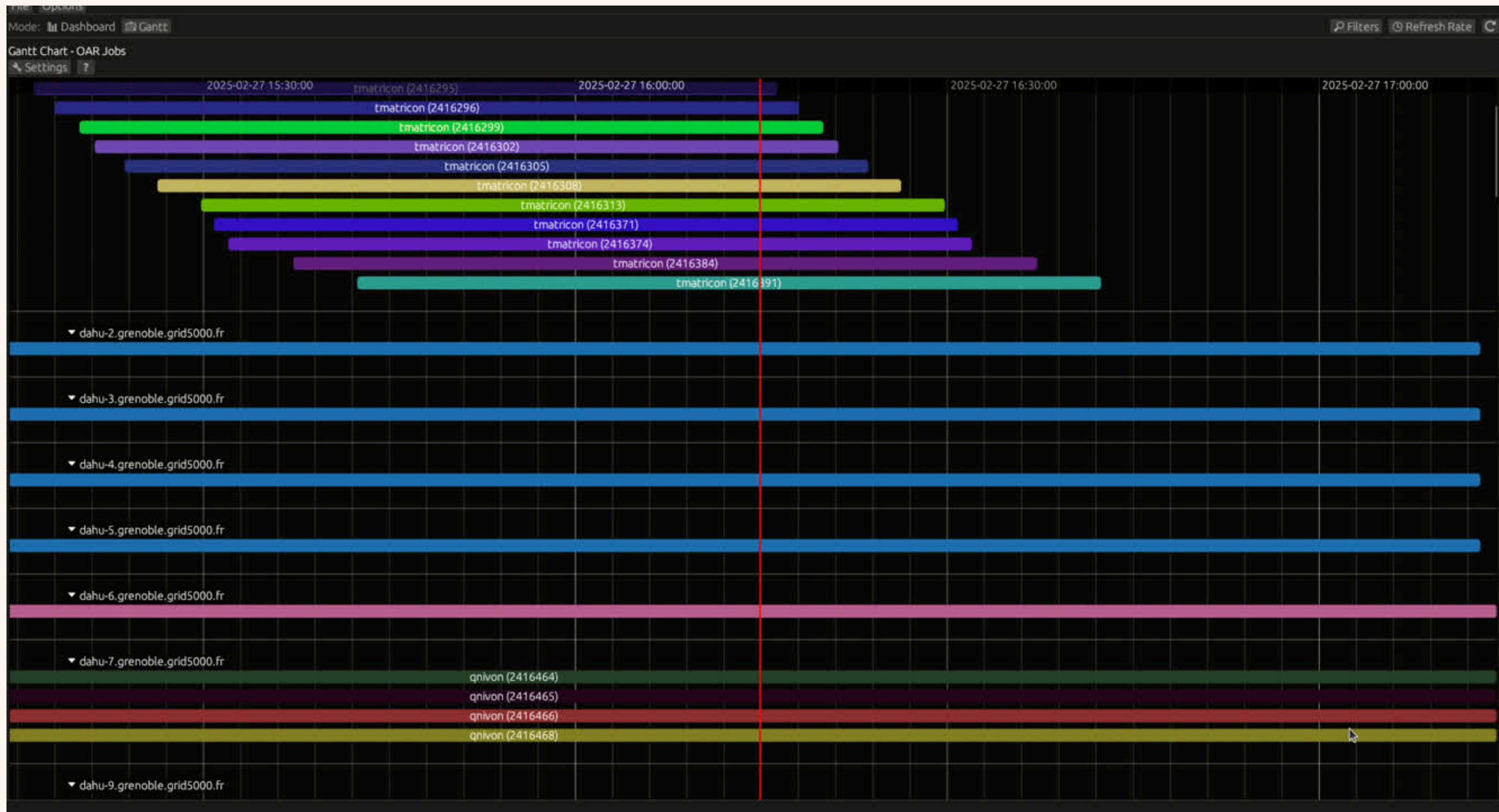


Diagramme de Gantt (Grid5000)



Goard Rust 2025



Source : <https://github.com/Goard-Rust/.github/blob/main/profile/goard.gif>

Composition de notre équipe

Amine
Salmane

API
Chef de Projet

Moataz
Er-rami

API

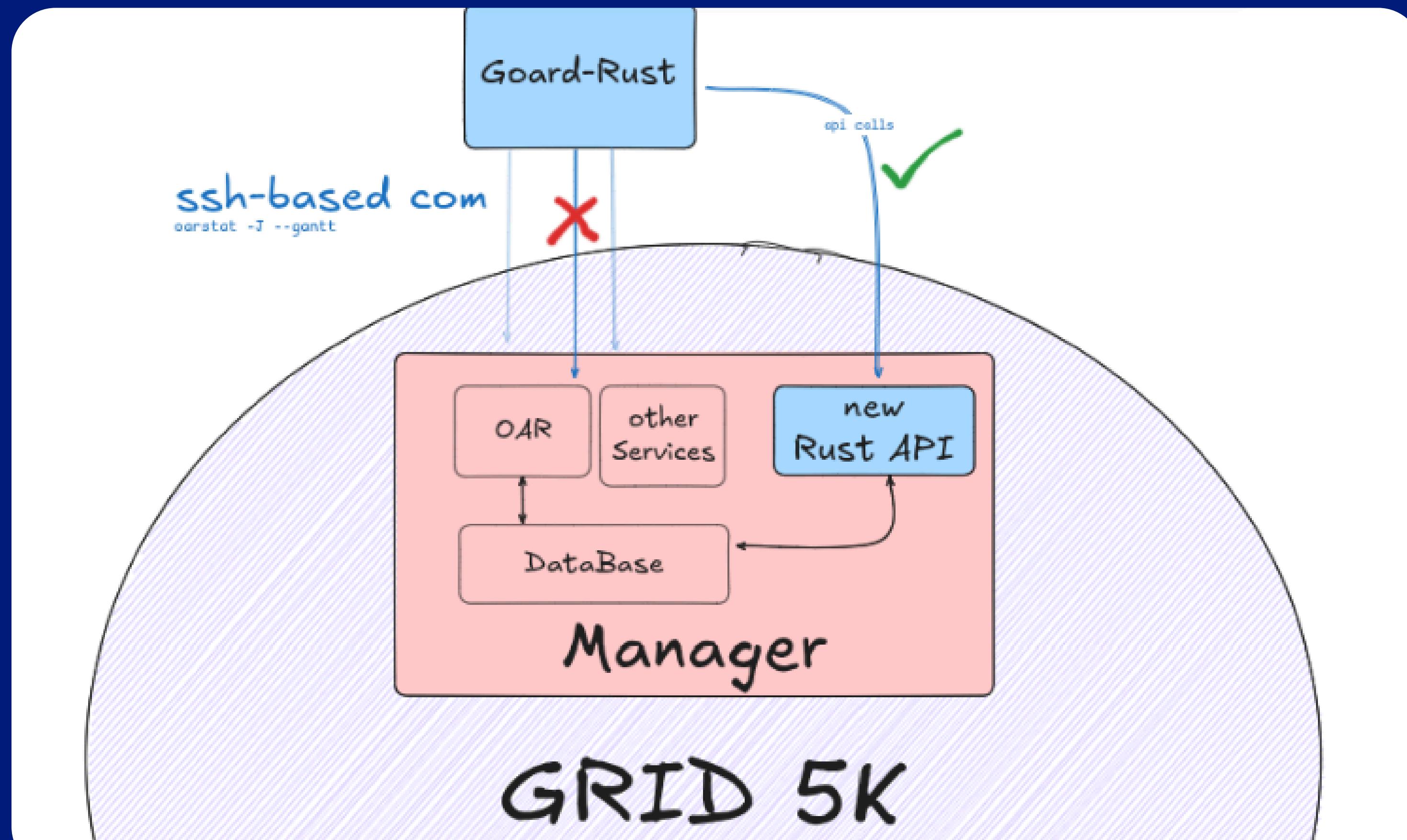
Aymane
Amessegher

UI/UX

Dila
Memil

UI/UX

L'architecture du système



Les technologies à utiliser

API

Tokio/AXUM 

Aide : Open Api Swagger/Scalar

GUI (UI)

Librairie : Egui 

Common

Cargo

Docker

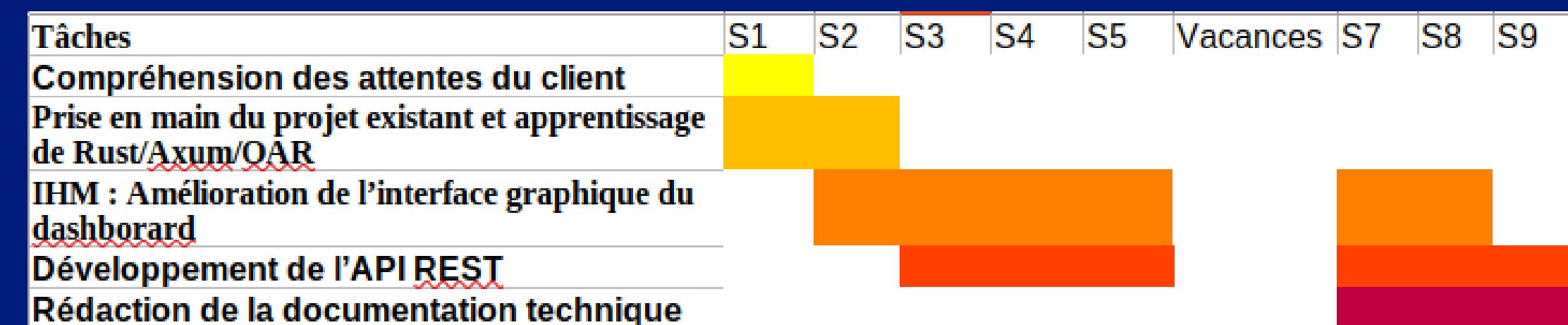
Github actions

Discord

Plan de travail

Ticket	Solution	Etat	Solution	Ticket	Etat
Architecture	DDD / Multi-crates	✓	-Réorganiser le code pour garantir une maintenabilité.	Architecture	X
Metier	Modeliser la bdd a partir de la doc/code OAR3		-Ajout de la barre hiérarchique (site,cluster,host)	Metier	✓
Confiance	Doc-as-Code	🔍 ✓			
Sécurité	Middleware OAuth / OpenID Connect	X	-Améliorer la lisibilité pour un résultat compact	Design	

Diagramme de Gantt



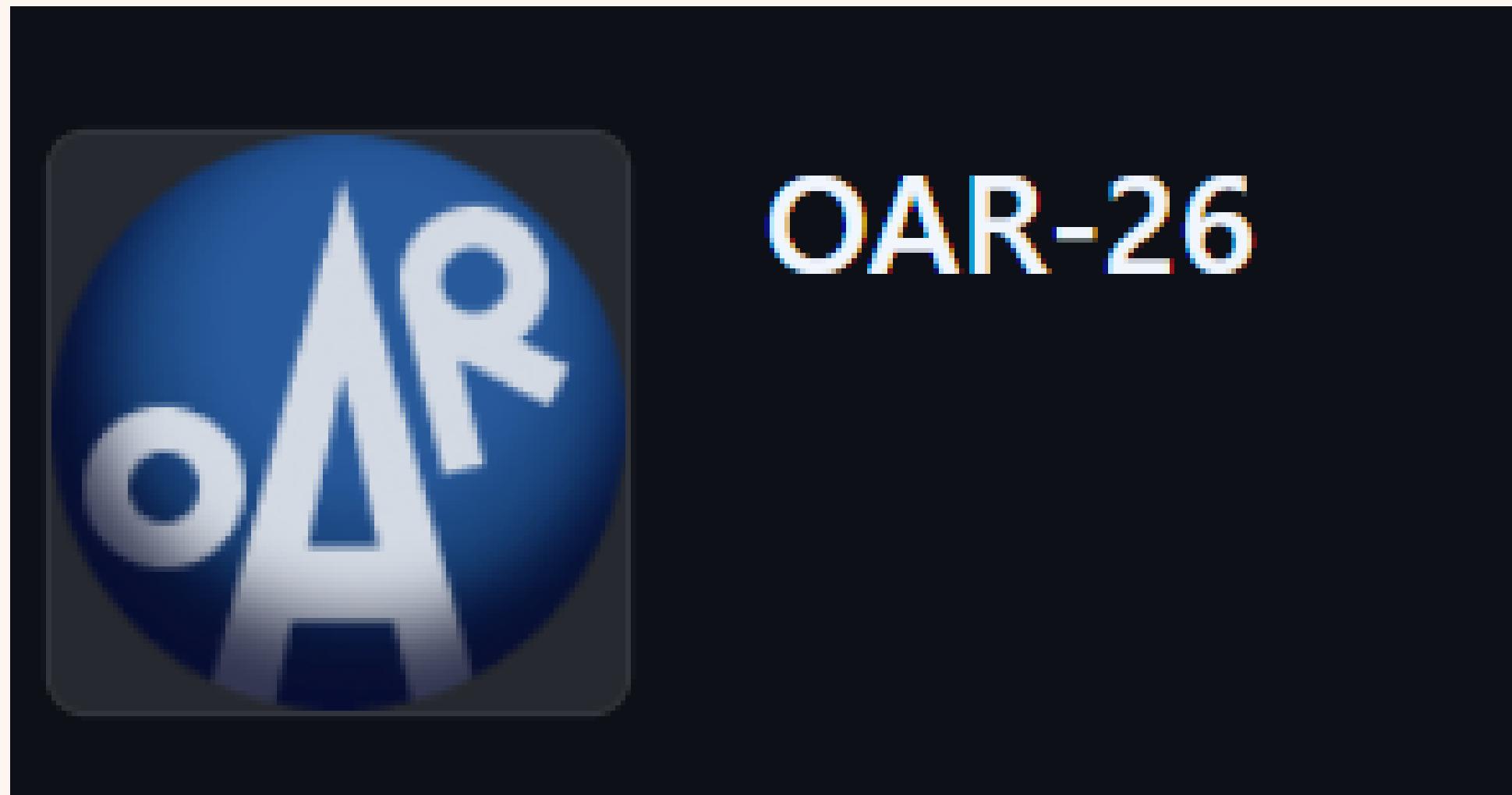
Difficultés

- Apprentissage Rust
- Manque de ressource (langage émergent)



**Merci pour votre
Attention !**

ANNEXE



Code source

Les enjeux :

OAR gère l'allocation de ressources HPC pour des milliers de jobs quotidiens dans les clusters de calcul. Migration progressive vers Rust pour gains en performance, sécurité mémoire et concurrence. Les centres de calcul exigent fiabilité 24/7 et réactivité (<100ms pour requêtes API).

Solutions actuelles & insuffisances :

Version Python actuelle : lente pour traiter gros volumes, consommation mémoire élevée, difficultés de maintenance. Dashboard 2024-2025 créé mais API REST manquante.

Besoins client :

- API REST performante (remplacement Python)
- Dashboard opérationnel avec Gantt temps réel
- Maintien compatibilité ascendante et compacité

Défis techniques :

- Interfaçage Rust <-> base existante OAR
- Architecture asynchrone (Axum/Tokio)
- Parsing/validation requêtes complexes

Approche solution :

Technique : Axum pour API REST, architecture modulaire (handlers → services → data), POC existant comme base
Organisationnelle : tests unitaires/intégration continus .

Cadrage initial :

- Durée : 9 semaines
- Livrables : architecture documentée, API fonctionnelle, dashboard enrichi
- Budget temps : ~200h
- Risques : complexité intégration existant, difficulté d'apprentissage Rust

Tableau comparatif Framework

Framework	Points forts	Limites
axum	<ul style="list-style-type: none"> - Ergonomie moderne & modulaire - Très bon écosystème async - Facile à étendre 	<ul style="list-style-type: none"> - Peut être verbeux avec traits avancés
actix-web	<ul style="list-style-type: none"> - Très hautes performances - Bon pour API à fort trafic - Large support middleware 	<ul style="list-style-type: none"> - API plus complexe - Mémoire plus importante
rocket	<ul style="list-style-type: none"> - API très conviviale - Routing macro intuitif - Bon pour prototypes 	<ul style="list-style-type: none"> - Moins performant - Moins adapté à gros services API
warp	<ul style="list-style-type: none"> - Style fonctionnel & composable - Bon pour APIs légers 	<ul style="list-style-type: none"> - Moins populaire - Architecture un peu différente

The screenshot shows the OAR 3 API documentation interface. At the top, there's a navigation bar with a search bar, a 'K' icon, and a sidebar menu. The sidebar includes 'Introduction', 'Users' (selected), 'Get user by ID' (with a 'GET' icon), and 'Models'. The main content area has a title 'OAR 3 API' with an 'OAS 3.1.0' badge, a 'Download OpenAPI Document' link, and a 'description' section. On the right, there's a 'Server' section with 'http://0.0.0.0:3000' and a cartoon crab icon, and a 'Client Libraries' section with links for 'Shell' (selected), Ruby, Node.js, PHP, Python, and More. Below this, there's a 'Curl Shell' section.

In the center, under the 'Users' section, there's a 'OPERATIONS' section with a 'GET /users/{id}' button. A large callout box highlights the 'Get user by ID' operation. This box contains the operation title, a description ('Retrieve a detailed user profile by their unique UUID from the database.'), 'Path Parameters' (with 'id' as a required parameter), 'Responses' (listing '200' and '404'), and a 'Test Request' button. To the right of the callout, there's a code snippet for 'curl' and a JSON schema for the response.

OAR 3 API OAS 3.1

[/api.json](#)

description

Users

GET /users/{id} Get user by ID

Retrieve a detailed user profile by their unique UUID from the database.

Parameters

Name Description

id <small>* required</small> string(\$uuid) (path)	The unique UUID of the user id
---	-----------------------------------

Responses

Code	Description	Links
200	Media type <small>application/json</small> Controls Accept header. Example Value Schema <pre>{ "email": "string", "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6", "username": "string" }</pre>	No links
404	User not found in the system	No links
500	Internal server error - something went wrong on our end	No links

Plan de Travail

Tâches	S1	S2	S3	S4	S5	Vacances	S7	S8	S9
Compréhension des attentes du client									
Prise en main du projet existant et apprentissage de Rust/Axum/QAR									
IHM : Amélioration de l'interface graphique du dashborard									
Développement de l'API REST									
Rédaction de la documentation technique									
S3	Ajout barre hiérarchie cluster					Choix Architecture + Generation doc			
S4	réorganiser le code (maintenabilité) prototype (Stats) + implémentation					Finaliser schema bdd			
S5	Ajout de filtre pour la visualisation					Implementation JWT			

État actuel de l'UI

