Platmosphere

A MIA-PLATFORM INVITATION

CHAPTER 2024

Compose Your Future

mia Platform

MILAN - 14th MAY

Talent Garden Calabiana

Speakers

Fabio Grasso
Solutions Engineer
@Okta

Federico Maggi
Senior Technical Leader
@Mia-Platform

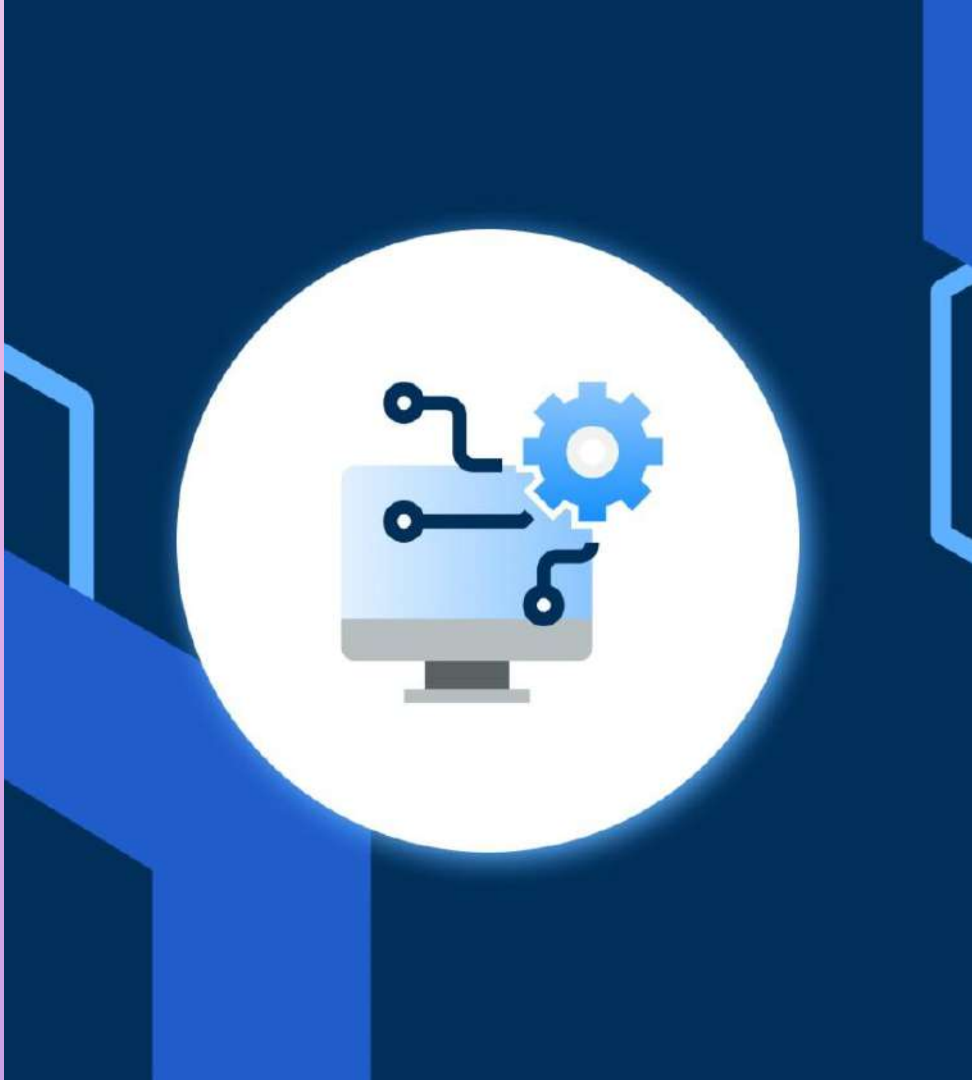
Secure Your Platform with IAM and Fine Grained Authorization

Platform Engineering and IAM

What is Platform Engineering and what are its challenges?

"Platform engineering emerged in response to the increasing complexity of modern software architectures. Today, non-expert end users are often asked to operate an assembly of complicated arcane services ..."

Gartner
https://www.gartner.com/en/articles/what-is-platform-engineering



Different People
Doing
Different Stuff

Platform Engineering Team

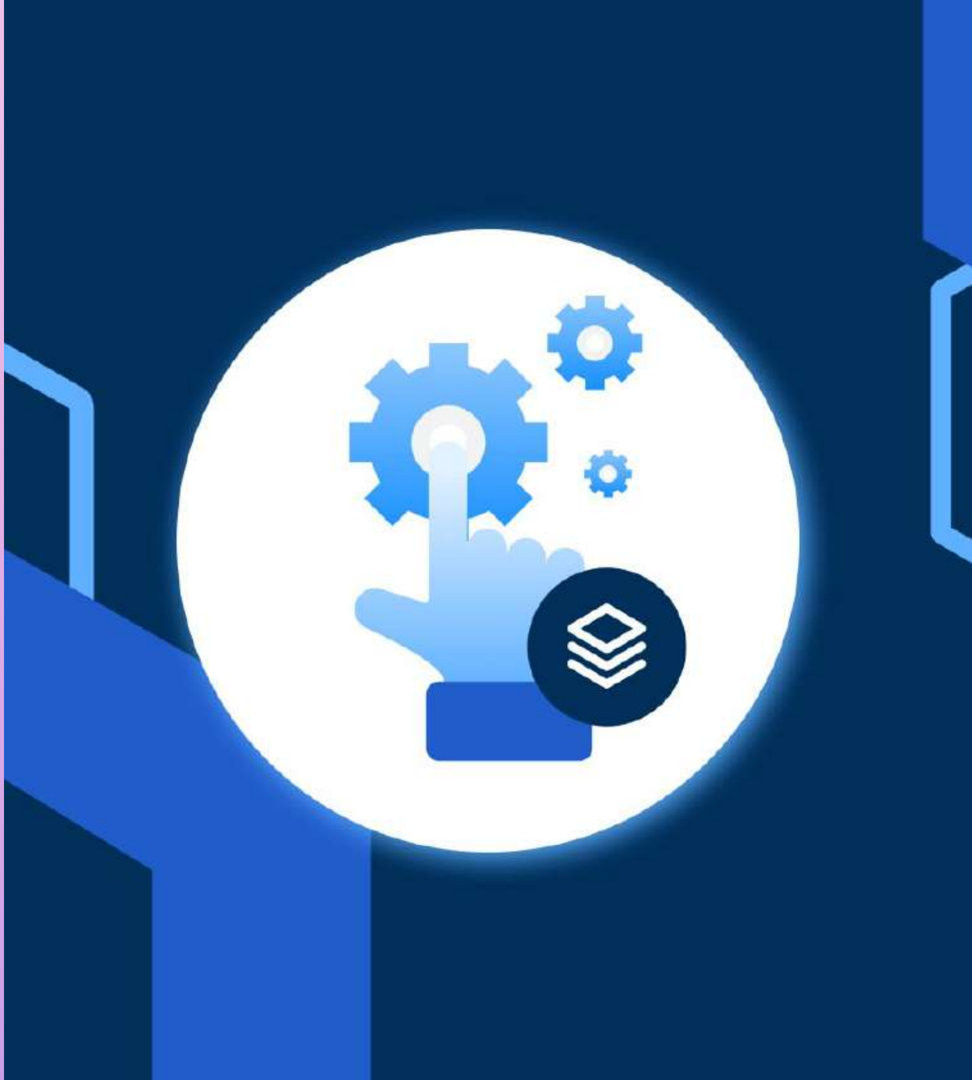
Provide tools and gold standards

Provide the means for managing all the resources that may be needed by developers.

Everything as-a-Service

—

Cluster and services provisioning



Application Developers

Focus on delivering code
Embracing DevOps practicing

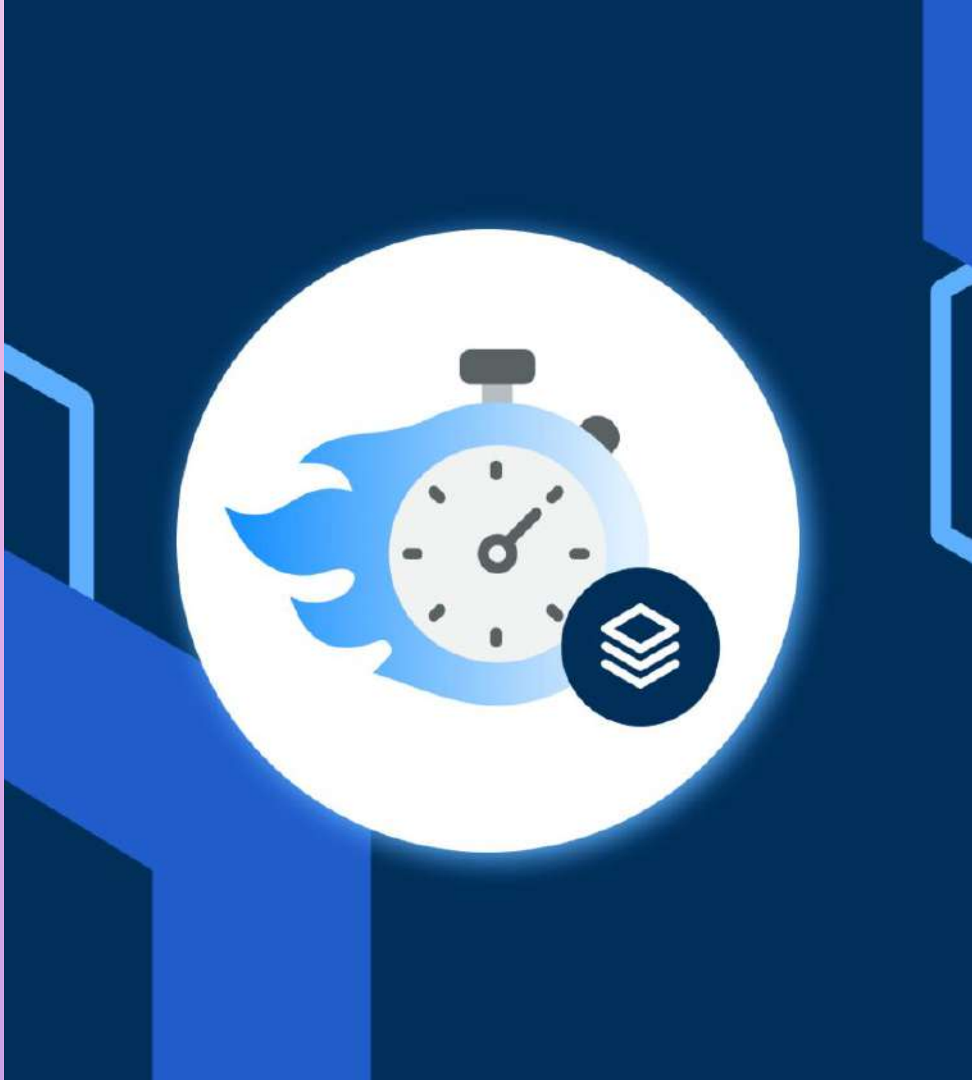
Using services provided by the Platform Engineering team they just ship their code:

From 0 to Production

—

Collaborate on Code Repositories

Runtime environment creation/deploy/monitoring



Product Managers

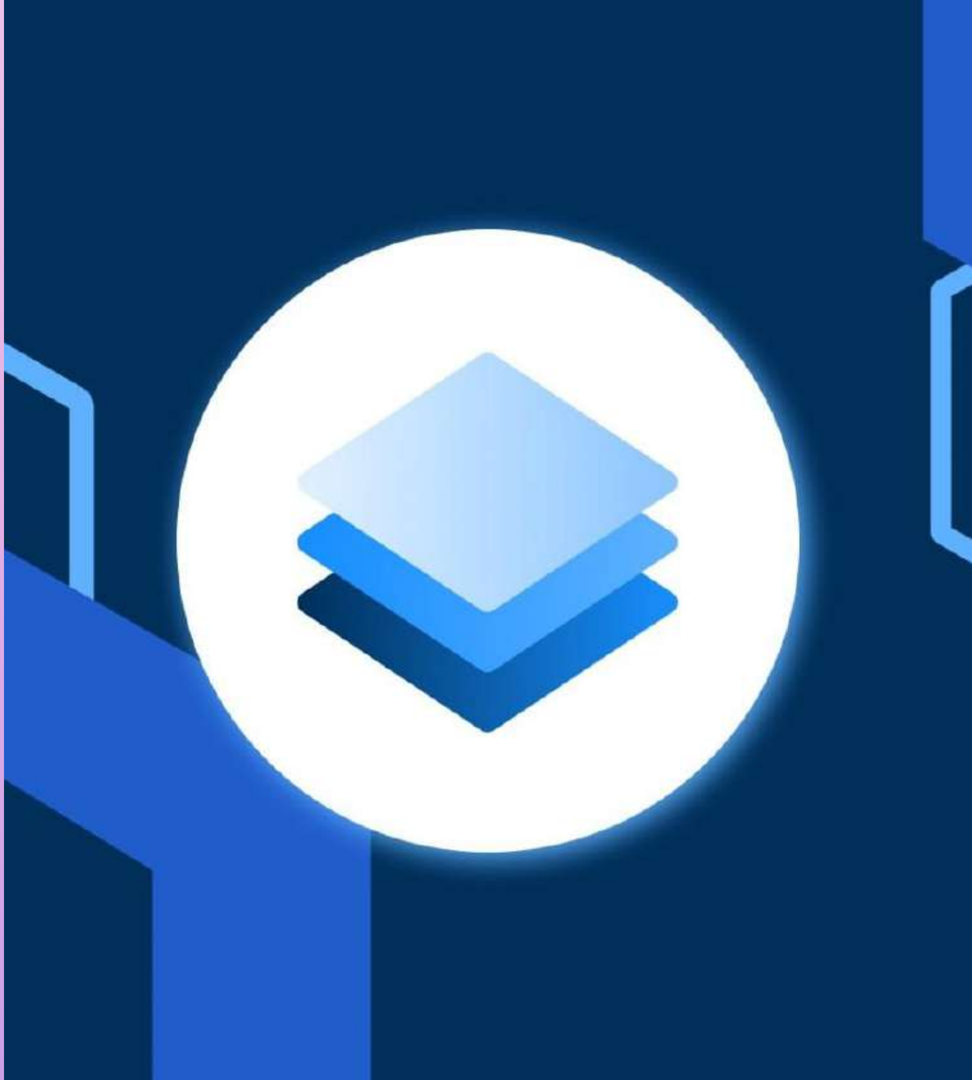
Kept involved in their Product lifecycle

Product/Project managers can keep the development pace and use the Platform insights to understand their Product's future

—

Review product catalog

inspect business insight from application metrics



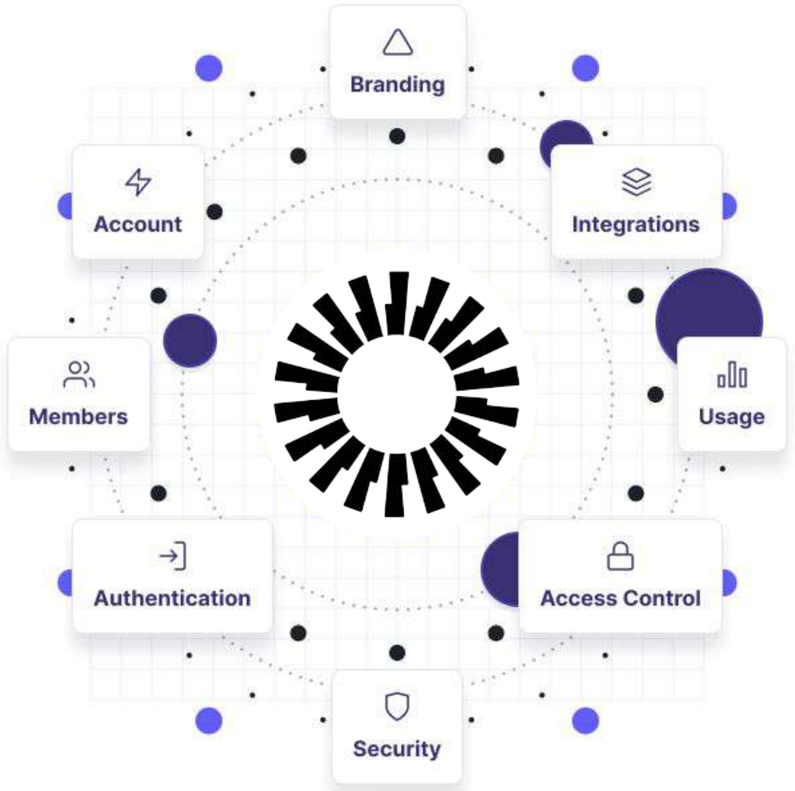
IAM and Fine Grained Access Control are a Fundamental player

IAM

IAM & CIAM

(Customer) Identity and Access Management

- **Authentication:** Verify the identity of users or services accessing the system.
- **Authorization:** Control access based on permissions: Grant or deny access to resources based on predefined permissions.
- **Monitor and audit:** Keep track of user access and system activities for security and compliance purposes.
- **Orchestration:** Automate user provisioning, deprovisioning, and access management processes throughout the user lifecycle.

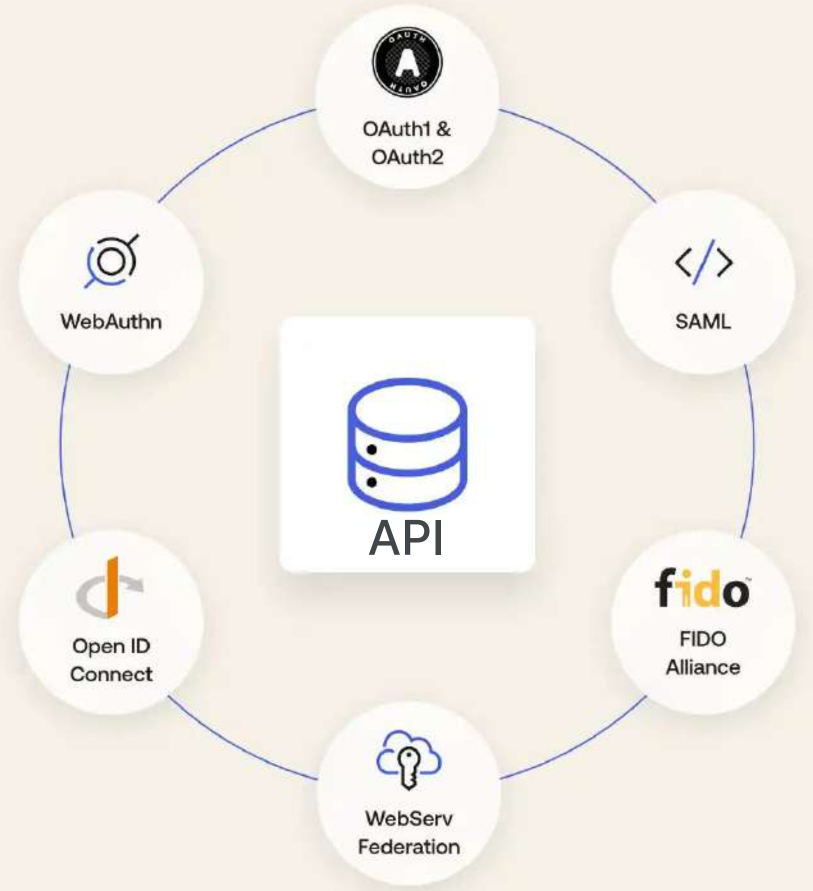


API Access Management

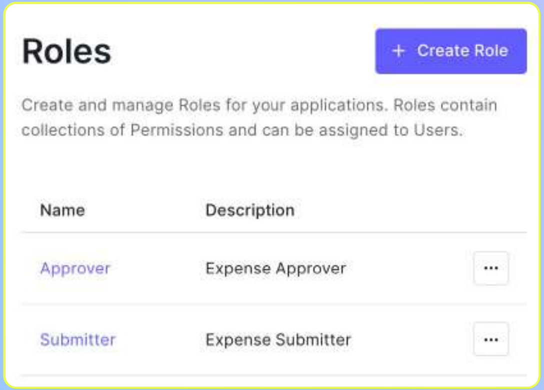
Leverage IAM for protect API (...and platforms)

Regulate access to Application Programming Interfaces (APIs) to ensure secure and controlled data exchange between software systems.

Implement authentication, authorization, and policies such as rate limiting and IP whitelisting to safeguard APIs from unauthorized access and potential threats.



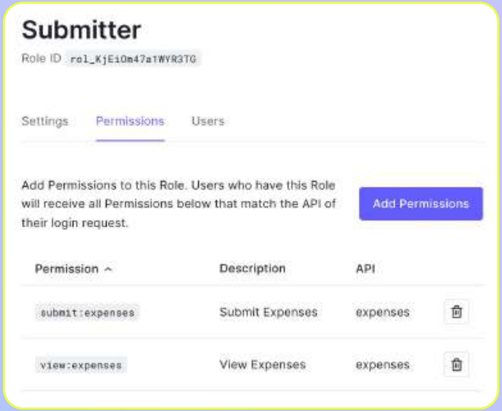
Role-Based Access Control



Roles

Create and manage Roles for your applications. Roles contain collections of Permissions and can be assigned to Users.

Name	Description
Approver	Expense Approver
Submitter	Expense Submitter

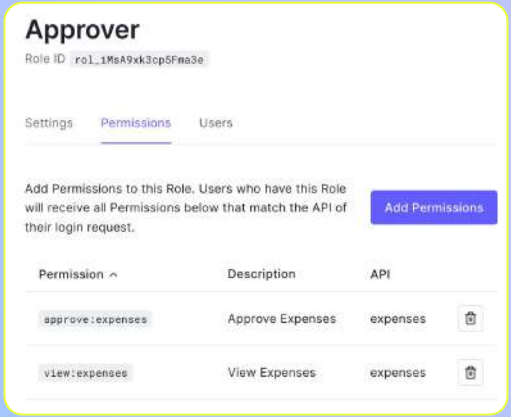


Submitter

Role ID `rol_KjEiOm47aTWYR3TG`

Add Permissions to this Role. Users who have this Role will receive all Permissions below that match the API of their login request.

Permission ^	Description	API
`submit:expenses`	Submit Expenses	expenses
`view:expenses`	View Expenses	expenses

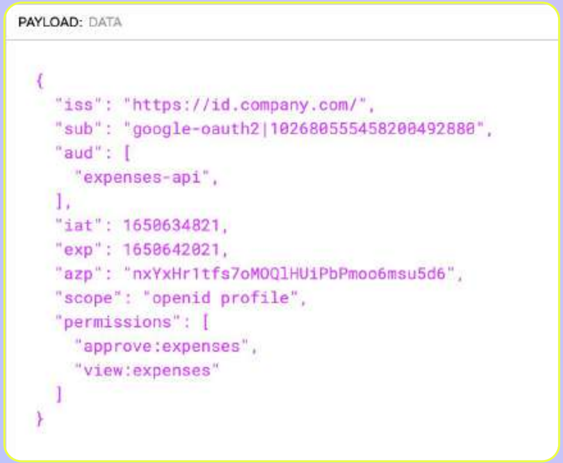


Approver

Role ID `rol_1MsA9xk3cp5Fma3e`

Add Permissions to this Role. Users who have this Role will receive all Permissions below that match the API of their login request.

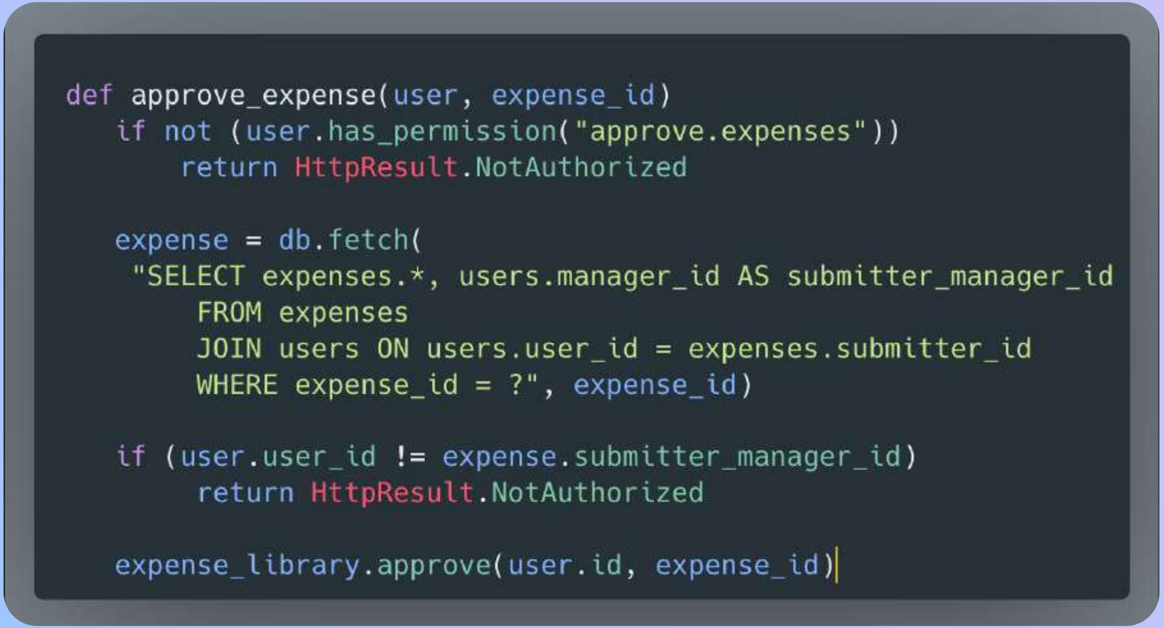
Permission ^	Description	API
`approve:expenses`	Approve Expenses	expenses
`view:expenses`	View Expenses	expenses



PAYLOAD: DATA

```
{
  "iss": "https://id.company.com/",
  "sub": "google-oauth2|102680555458200492880",
  "aud": [
    "expenses-api",
  ],
  "iat": 1650634821,
  "exp": 1650642021,
  "azp": "nxYxHr1tfs7oMOQlHUiPbPmoo6msu5d6",
  "scope": "openid profile",
  "permissions": [
    "approve:expenses",
    "view:expenses"
  ]
}
```

RBAC + Custom Code, ABAC?



```
def approve_expense(user, expense_id)
   if not (user.has_permission("approve.expenses"))
       return HttpResult.NotAuthorized

   expense = db.fetch(
    "SELECT expenses.*, users.manager_id AS submitter_manager_id
        FROM expenses
        JOIN users ON users.user_id = expenses.submitter_id
        WHERE expense_id = ?", expense_id)

   if (user.user_id != expense.submitter_manager_id)
        return HttpResult.NotAuthorized

   expense_library.approve(user.id, expense_id)
```

Relationship-Based Access Control
Fine Grained Authorization

1. Authorization Model

```
type employee
  relations
    define manager: [employee]
    define can_manage: manager or can_manage from manager

type report
  relations
    define submitter: [employee]
    define approver : can_manage from submitter
```

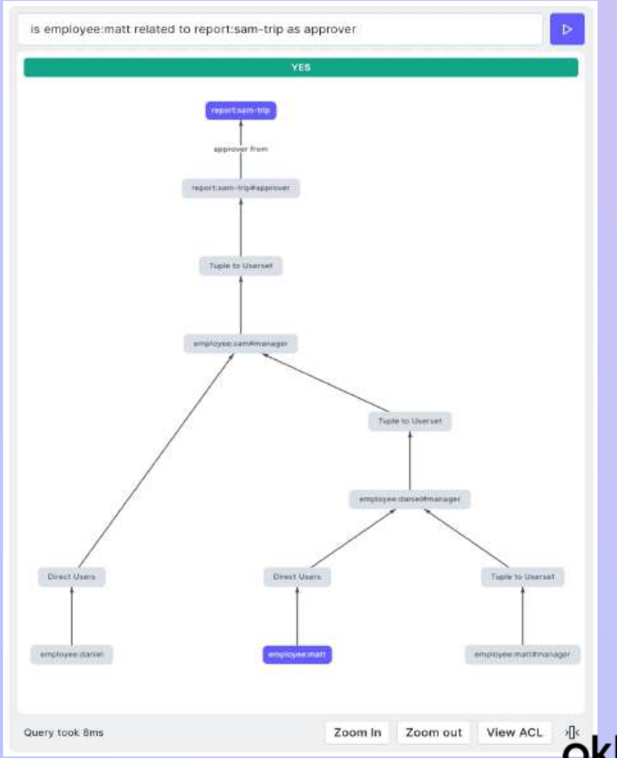
2. Tuples

USER	employee:sam
OBJECT	report:sam-trip
RELATION	submitter

USER	employee:daniel
OBJECT	employee:sam
RELATION	manager

USER	employee:matt
OBJECT	employee:daniel
RELATION	manager

3. Query



is employee:matt related to report:sam-trip as approver

YES

How to: Fine Grained Authz

A brief journey with
Rönd and OpenFGA

Rönd

A lightweight container to distribute security policy enforcement

Rönd is based on **Open Policy Agent** and provides a way to **decentralize security enforcement** throughout your application.

Rönd runs as a **sidecar**, intercepts the API traffic and **applies security policies** before forwarding the request to your application.

Allows for integration **with no changes to the codebase**.

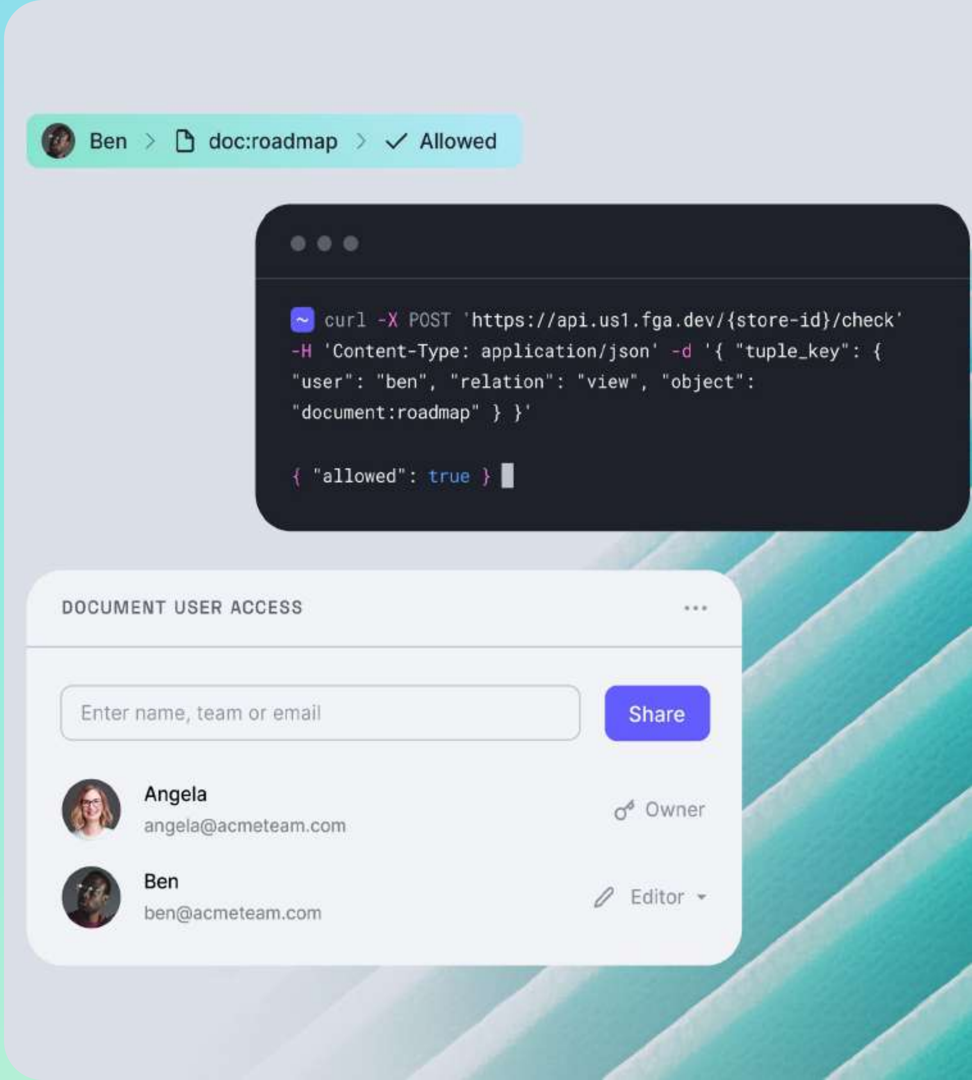
Okta FGA / OpenFGA

Fine Grained Authorization

Inspired by **Google Zanzibar** (Used in Drive, Youtube, etc.)

Design authorization models, from coarse grained to fine grained, in a way that's **centralized, flexible, fast, scalable and easy to use.**

Easily manage permissions to specific resources for groups, teams, organizations, or **any set of users**, using a **declarative language for access control models**.



```
~ curl -X POST 'https://api.us1.fga.dev/{store-id}/check'
-H 'Content-Type: application/json' -d '{ "tuple_key": {
"user": "ben", "relation": "view", "object":
"document:roadmap" } }'

{ "allowed": true }
```

DOCUMENT USER ACCESS

Enter name, team or email — Share

Angela	angela@acmeteam.com	Owner
Ben	ben@acmeteam.com	Editor

Rönd cornerstones

Security in depth

Sidecar service proxy to **decentralize policy enforcement**

Fine Grained Authorization

Run **policies** on any **user** or **contextual attributes**

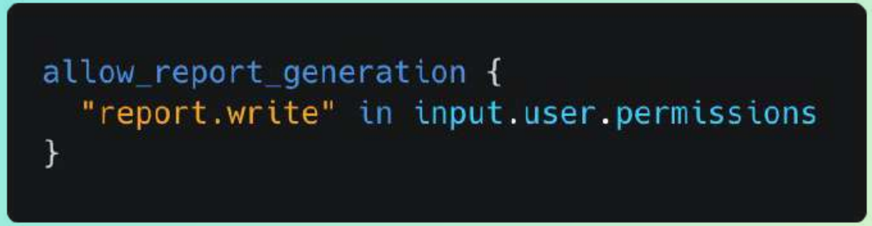
Developer experience

No code changes required to application code
Declarative policy language: **Rego** from **Open Policy Agent**

RBAC Permission

Simple RBAC permission-based policy

Only accept requests from users with the “report.write” permission

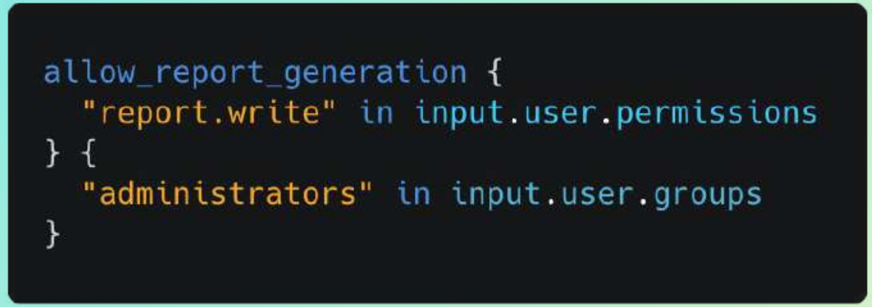


```
allow_report_generation {
  "report.write" in input.user.permissions
}
```

RBAC Permission

Simple RBAC permission-based policy

Only accept requests from users with the “report.write” **permission or have the** “administrators” **group**

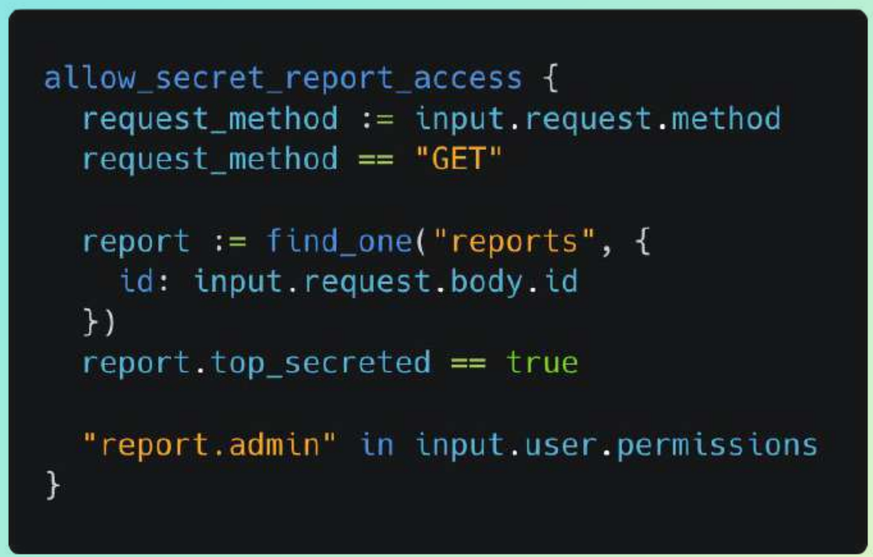


```
allow_report_generation {
  "report.write" in input.user.permissions
} {
  "administrators" in input.user.groups
}
```

Context based authorization

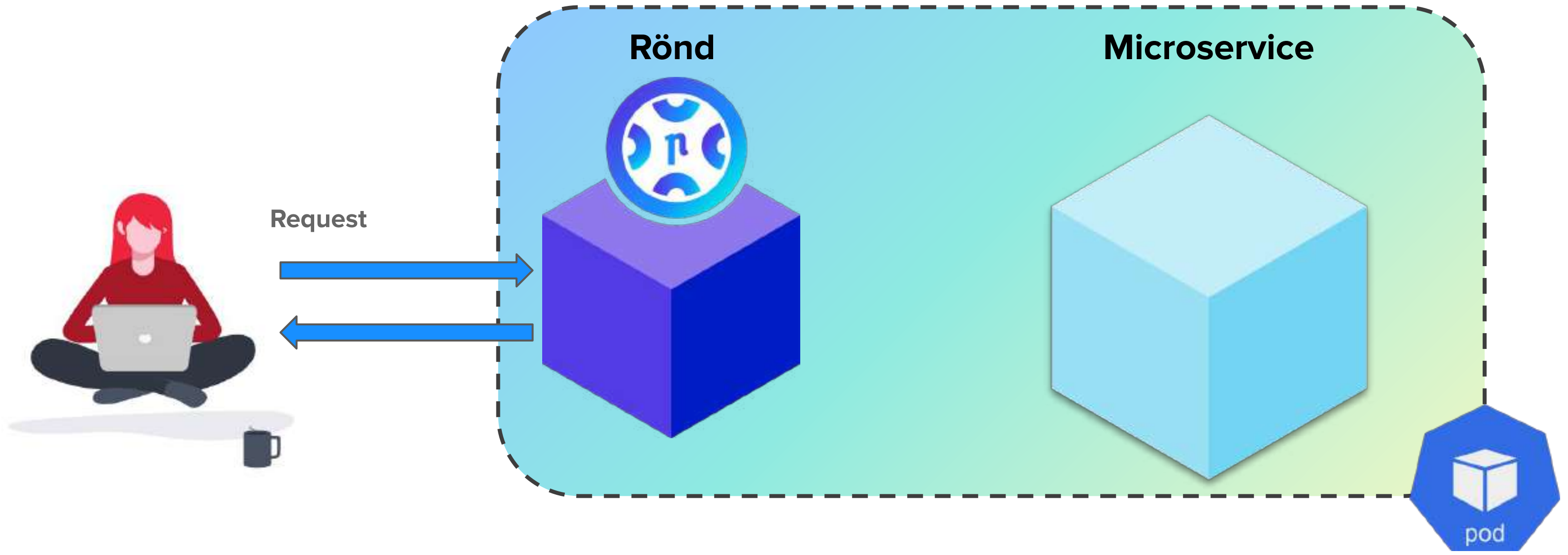
More complex authorization logics based on request data and information stored in the database

Allow read access to top secret reports only to users with the “report.admin” **permission**

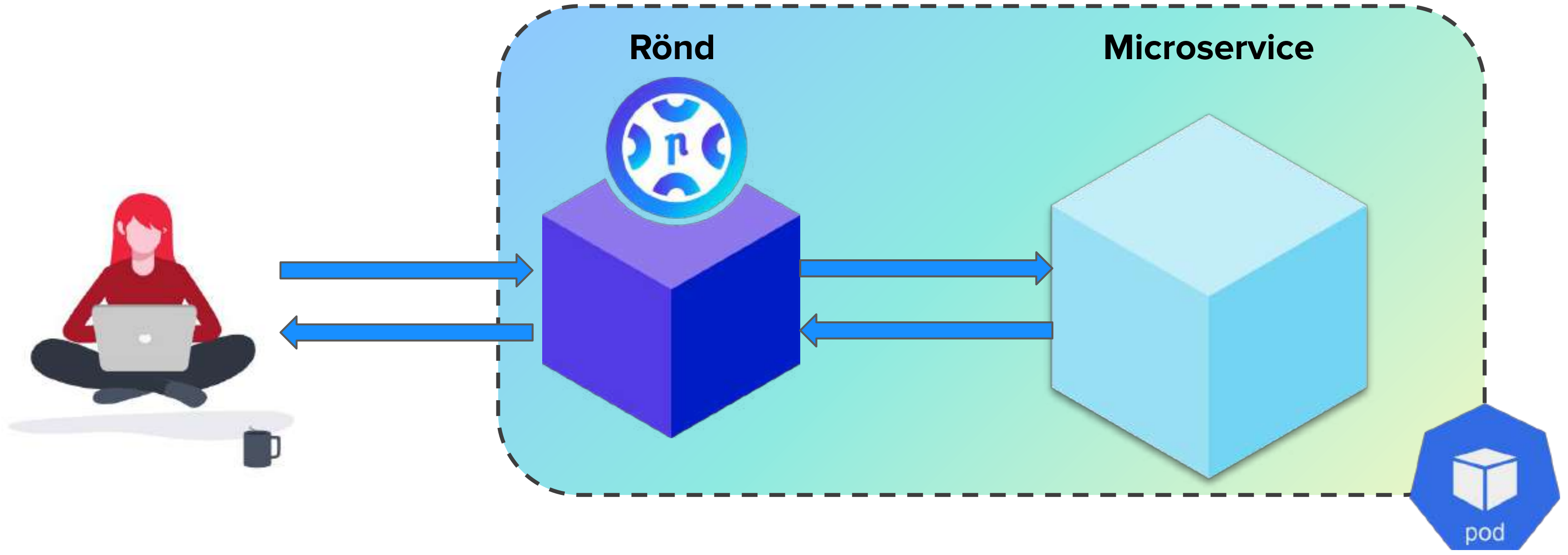


```
allow_secret_report_access {
  request_method := input.request.method
  request_method == "GET"

  report := find_one("reports", {
    id: input.request.body.id
  })
  report.top_secreted == true

  "report.admin" in input.user.permissions
}
```

Rönd flow rejecting the user request



Rönd flow authorizing the user request



Rönd Features

When running as a sidecar **Rönd** provides three main features

Prevent undesired API accesses

Modify responses to remove sensitive information

Protect your data with data filtering generation

Okta FGA

Fine-grained Authorization

Introducing Okta FGA / OpenFGA

Fine Grained Authorization Service for Developers

Relationship Based Access Control (ReBAC)

An evolution from **Role Based Access Control** and **Attribute Based Access Control**.

Inspired by Google Zanzibar

Used in Google Drive, Youtube, etc

Flexible enough to model any application domain at large scale.

Built to Scale

Can scale to **billions** of globally distributed users and resources.

Developer Friendly

Enable user collaboration and granular access control in your applications using **developer friendly APIs**.

FGA APIs

Use developer friendly APIs for everything, including adding relationships and performing authorization checks.

```
write(
 user = "employee:sam",
 relation = "submitter",
 object = "report:sam-trip"
)
```

```
check(
 user = "employee:matt",
 relation = "approver",
 object = "report:sam-trip"
)
```

Relationship Based Queries

Can Matt approve the *sam-trip* report?

```
check(
 user = "employee:matt",
 relation = "approver",
 object = "report:sam-trip"
)
```

Which users can approve the *sam-trip* report?

```
list-users(
 object = "report:sam-trip",
 relation = "approver"
)
```

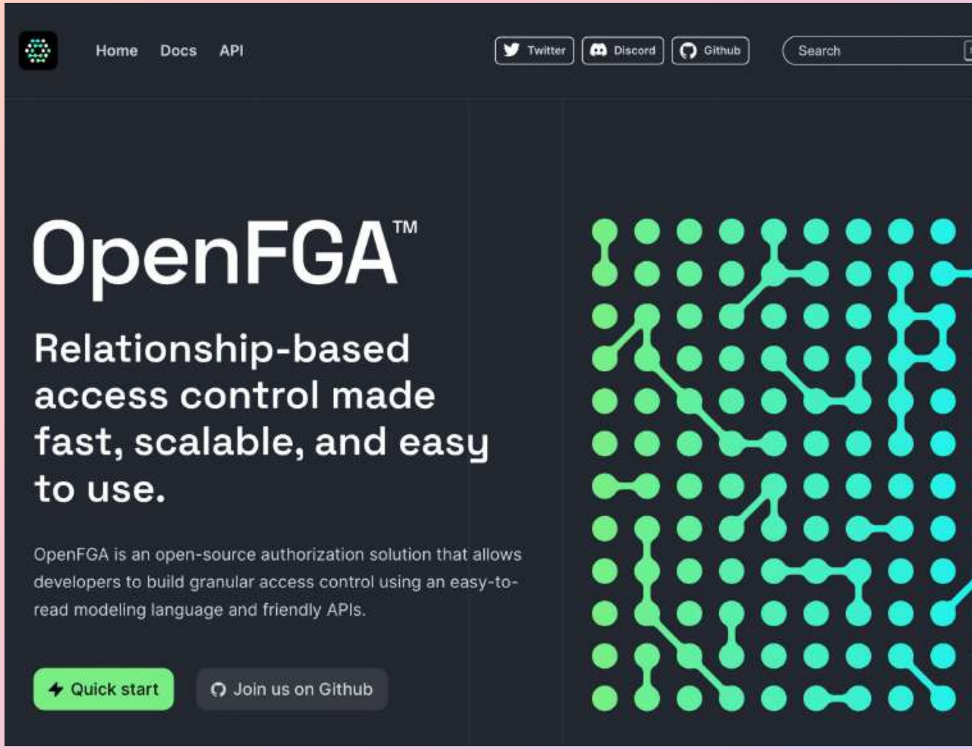
What actions can Matt perform on the *sam-trip* report?

```
list-relations(
 object = "report:sam-trip",
 user : "employee:matt"
)
```

OpenFGA

openfga.dev

- **Open Source solution**, owned by Cloud Native Computing Foundation, maintained by Okta.
- Used as **the core of Okta FGA**.
  Okta FGA has a DynamoDB backend, OpenFGA can be used with Postgres & MySQL.
- Okta does not have any commercial offering around OpenFGA.
- **OktaFGA** is offered as Cloud Service and provide **Active-Active replication**, **Enterprise-grade support** and managed security.



OpenFGA™

Relationship-based access control made fast, scalable, and easy to use.

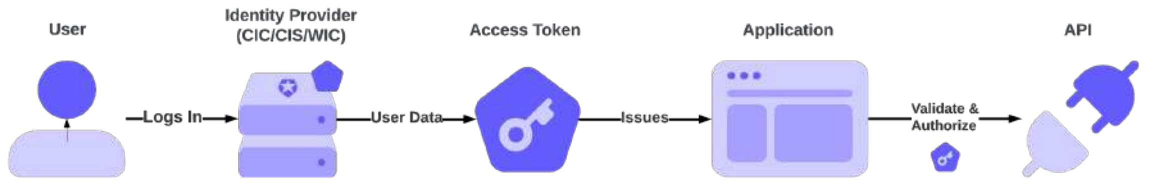
OpenFGA is an open-source authorization solution that allows developers to build granular access control using an easy-to-read modeling language and friendly APIs.

Implementing FGA: Authorization Patterns

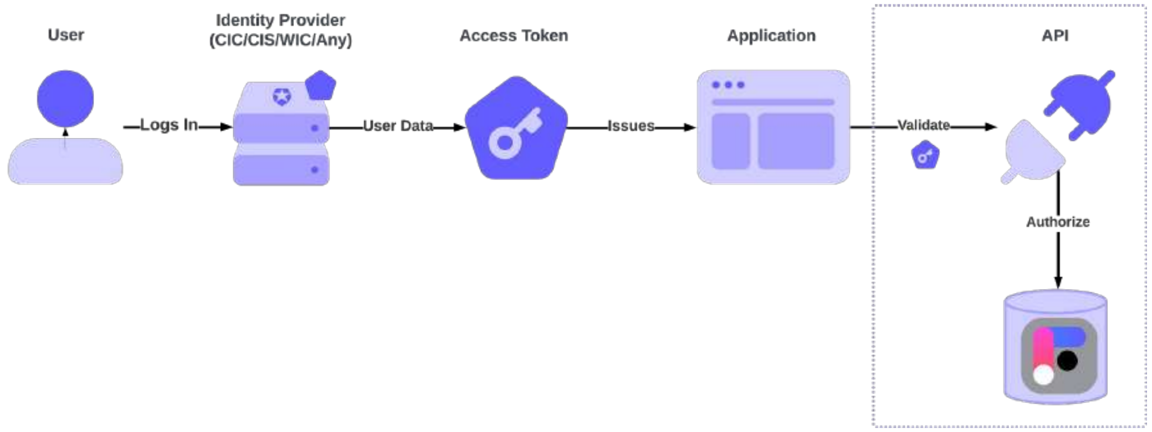
okta

Authorization Flows

Without FGA



With FGA



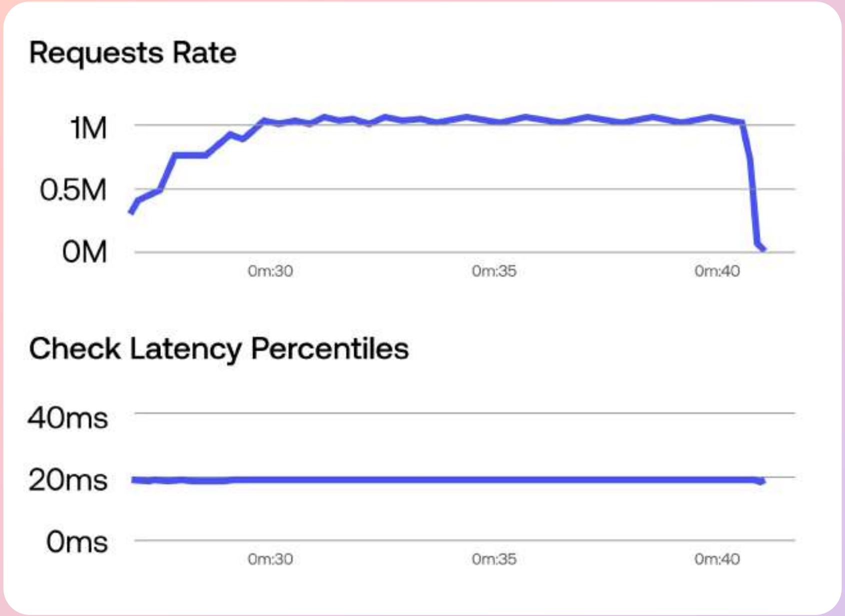
Unparalleled scalability

Okta FGA

1 million requests per second

100 billion relationships

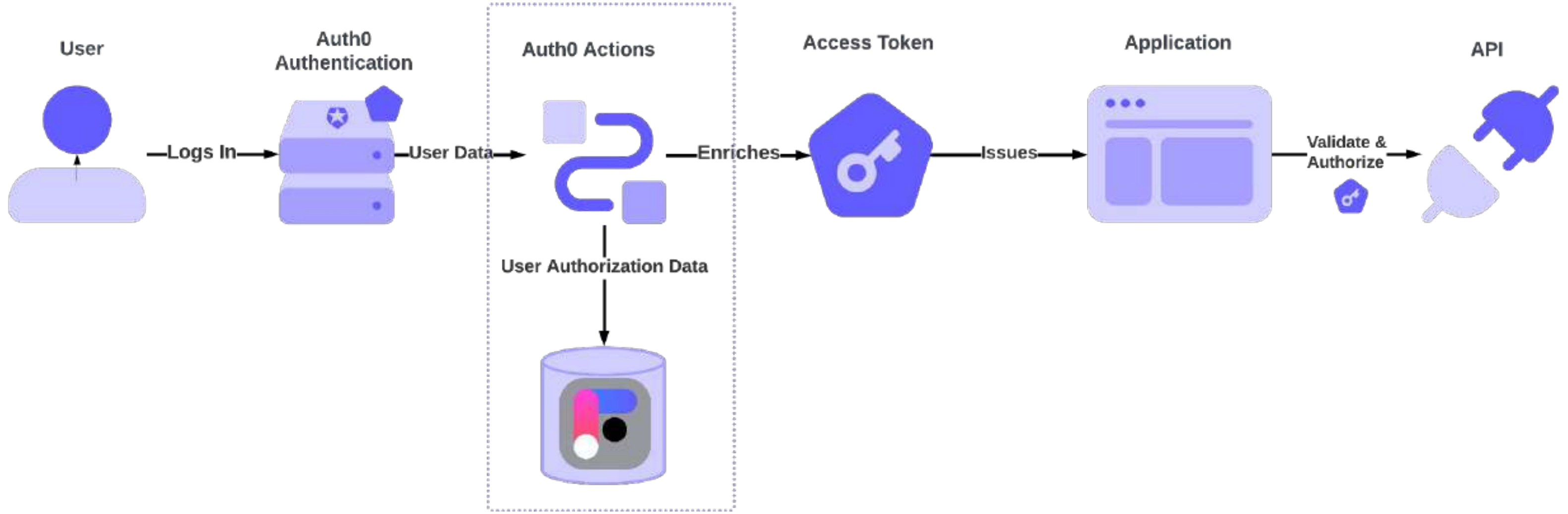
< 20ms P95 latency



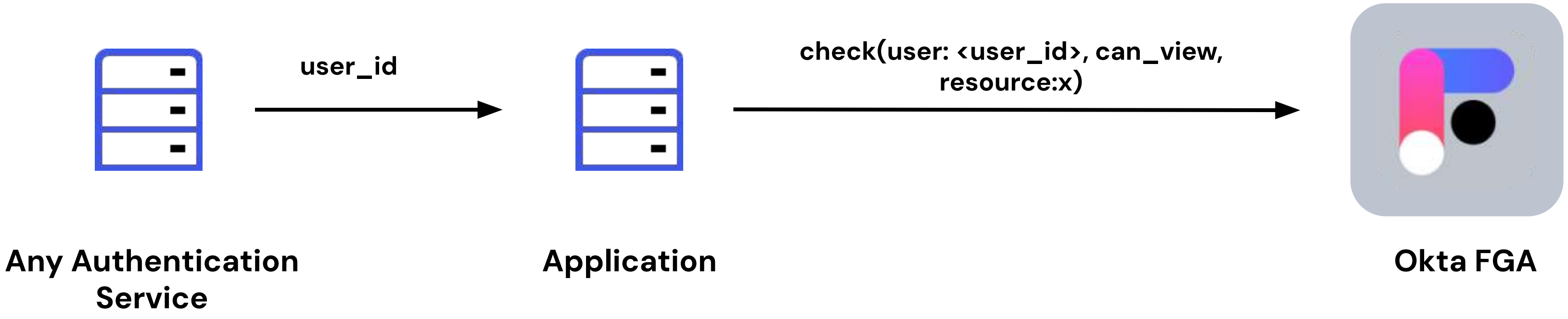
FGA unlimited scalability blog post

Authorization Flows

JWT-based authorization using FGA



FGA is independent of the Authentication Service


user_id
check(user: <user_id>, can_view, resource:x)
Any Authentication Service
Application
Okta FGA

Key takeaways

Platform Engineering gives a lot of power and flexibility to developers, without governance the risk of things going out of control is high.

—

IAM and Fine Grained Authorization are the cornerstone of Internal Developer Platforms, ensuring everyone can do what they are supposed to do and protecting organizations from being damaged.

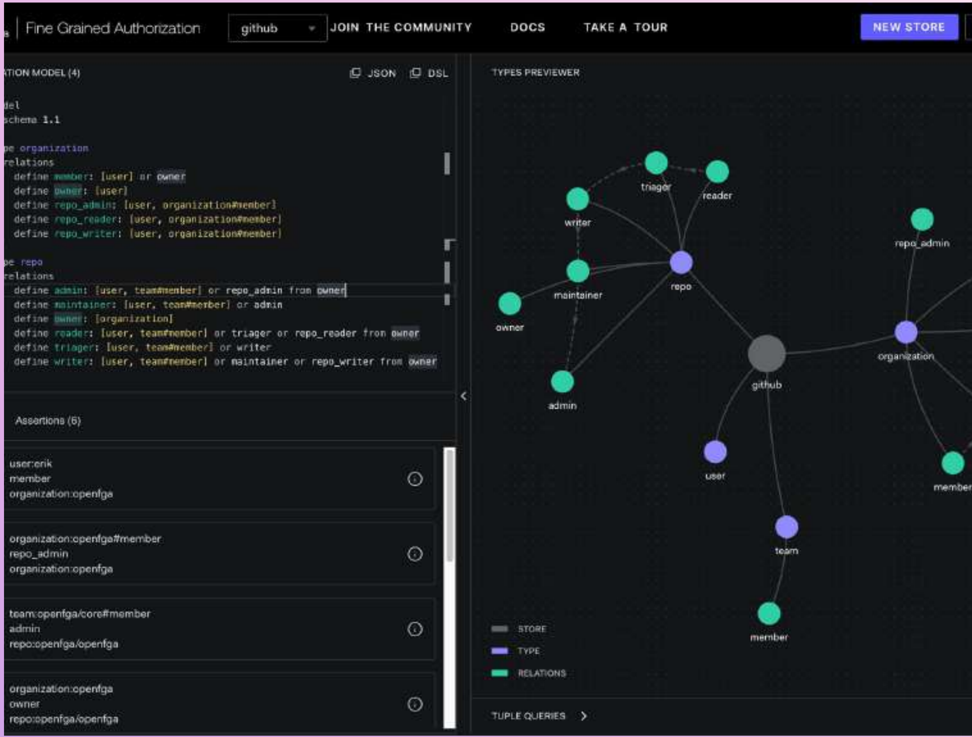
Rönd and OpenFGA provide different solutions to allow for IAM customization and fine grained control over user actions

- Rönd:
  - Based on Open Policy Agent
  - Decentralized security enforcement without application changes
- OpenFGA / Okta FGA:
  - Graph-based data model, enabling more complex relationships and fine-grained control
  - Accessible via API or using SDKs

Okta FGA Playground

play.fga.dev

- No signup required.
- Explore existing sample models (github, google drive, entitlements, IoT, expenses).
- Use it as a learning tool to iterate on your model and tuples.
- Documentation: docs.fga.dev
- Okta FGA Free tier for developers docs.fga.dev/subscription-plans



```
del
schema 1.1

pe organization
relations
  define member: [user] or owner
  define owner: [user]
  define repo_admin: [user, organization#member]
  define repo_reader: [user, organization#member]
  define repo_writer: [user, organization#member]

pe repo
relations
  define admin: [user, team#member] or repo_admin from owner
  define maintainer: [user, team#member] or admin
  define owner: [organization]
  define reader: [user, team#member] or triager or repo_reader from owner
  define triager: [user, team#member] or writer
  define writer: [user, team#member] or maintainer or repo_writer from owner
```

Assertions (6)

user:erik
member
organization:openfga

organization:openfga#member
repo_admin
organization:openfga

team:openfga/core#member
admin
repo:openfga/openfga

organization:openfga
owner
repo:openfga/openfga

Q&A & Feedback

mia Platform

Thanks

HEADQUARTERS

Italy

Via Imbonati 18, MAC7
20159 Milan