WSO2CON 2024

# WSO2's Digital Transformation Journey with Choreo: A Platformless Approach



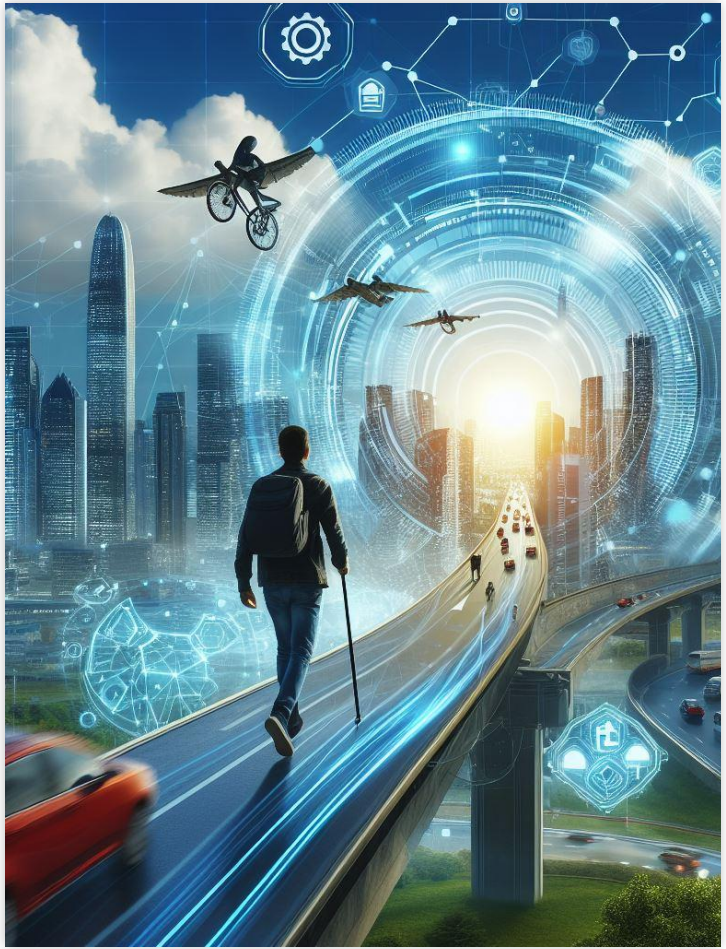
Chanuka Dissanayake
Senior Technical Lead
WSO2

# Navigating the Future Landscape

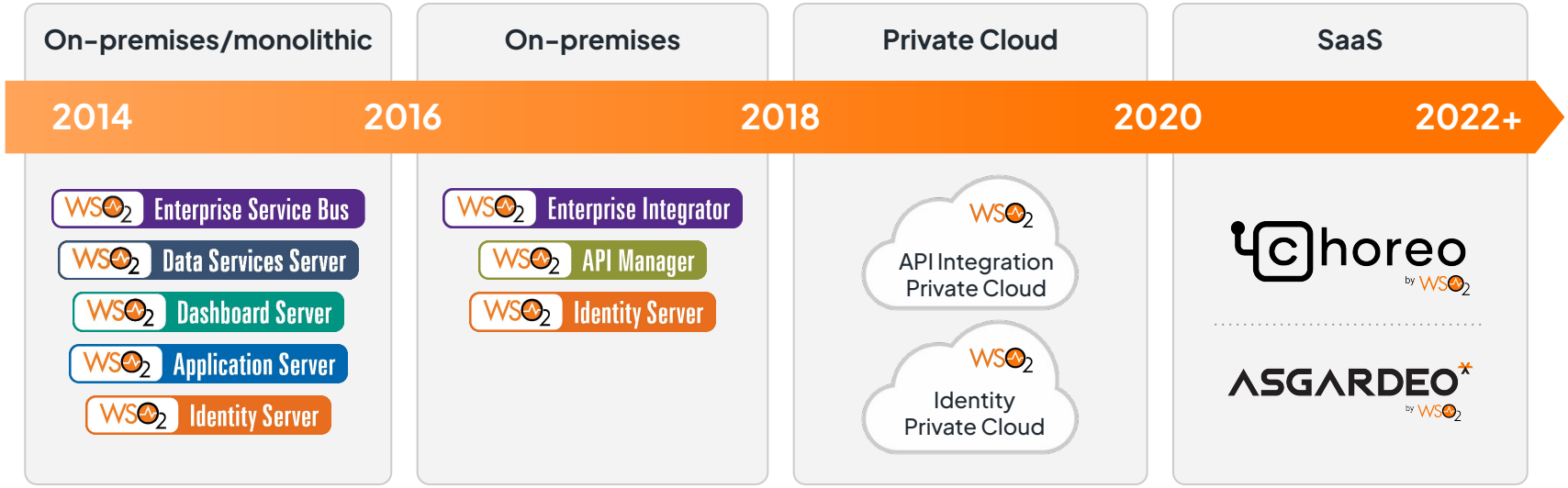
# Navigating the Future Landscape

Embark on a journey of innovation, where both **internal and external transformation** are essential.

Join us as we navigate through the key drivers, challenges, and opportunities shaping the future.

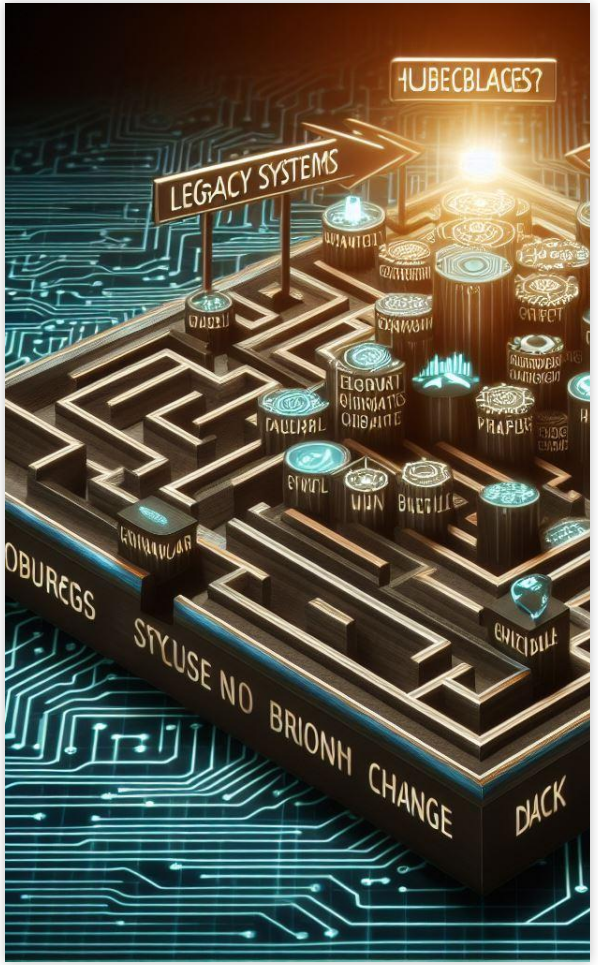


# Our Journey


On-premises/monolithic
On-premises
Private Cloud
SaaS
2014
2016
2018
2020
2022+
Enterprise Service Bus
Data Services Server
Dashboard Server
Application Server
Identity Server
Enterprise Integrator
API Manager
Identity Server
API Integration
Private Cloud
Identity
Private Cloud
choreo
by WSO2
ASGARDEO
by WSO2

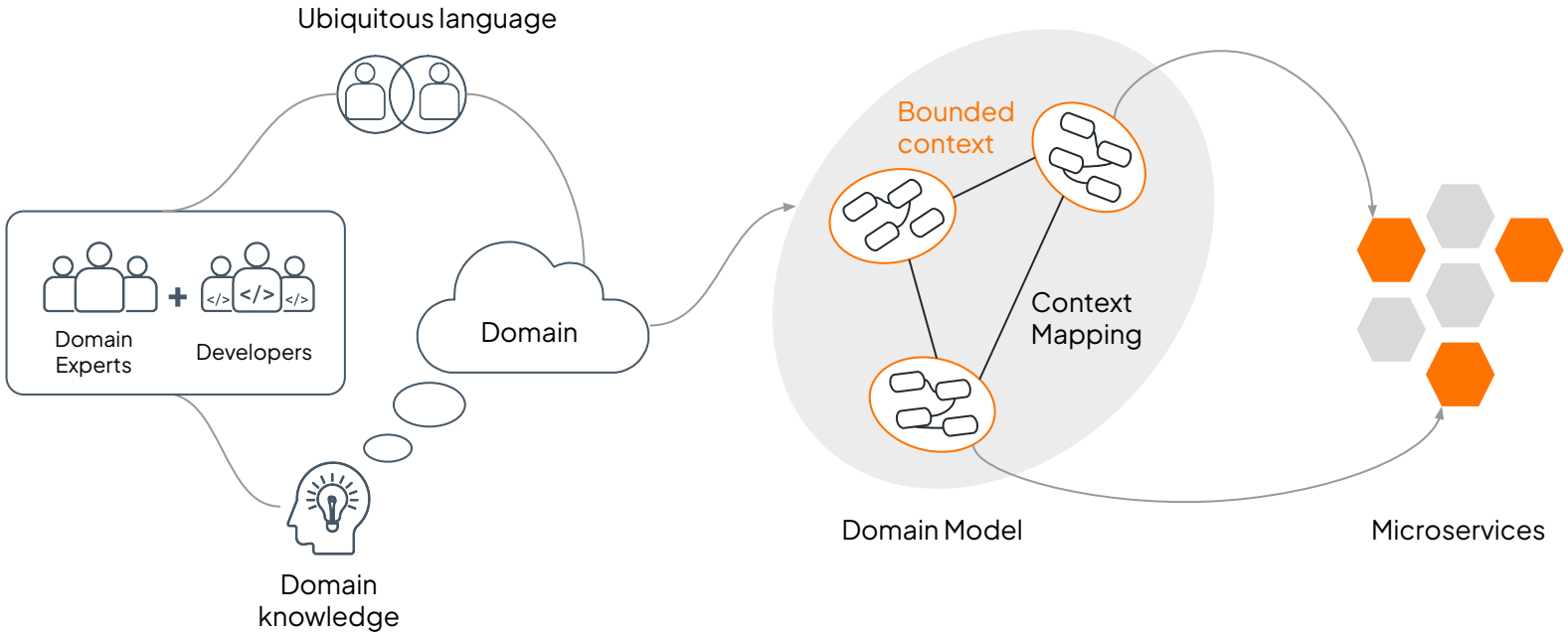
# What Were the Challenges?

- The difficulties of upgrading products, the costs of allocating resources, deploying, operating, and upgrading our own IT infrastructure
- The limitations of maintaining consistency which impacts on user experience and productivity
- The risks of running legacy applications, lack of support and security
- The barriers to innovation and collaboration



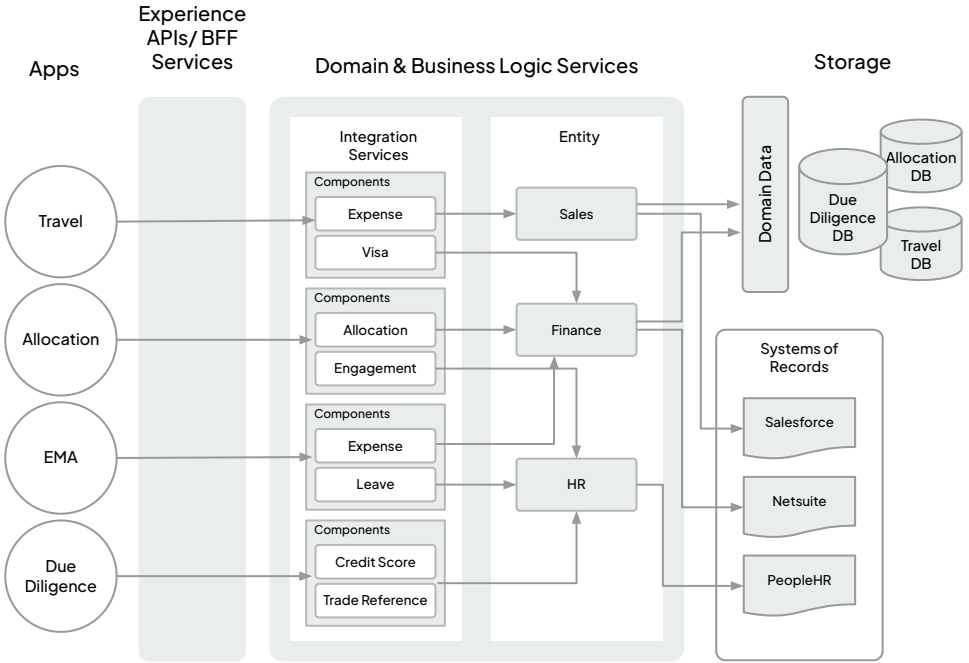
# Domain Driven Design

# Domain Driven Design



# Domain Driven Design, Components, Polyglot languages

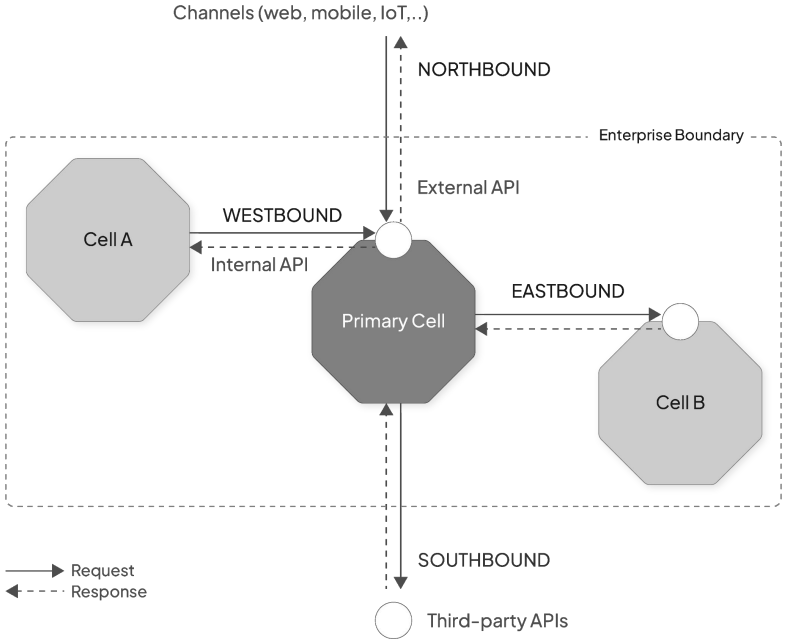
- DDD was used to identify entity services, integration services, apps, etc.
- Each service, app becomes a component in Choreo
  - Choreo supports hosting services, webapps, API proxies, period tasks, manual tasks, webhooks, etc.
- These services and apps are written in various languages
  - Choreo supports polyglot languages - Java, Go, Python, Ballerina, etc.
  - Ability to bring docker files or docker images



# Cell-Based Architecture

# Cells, Access Control

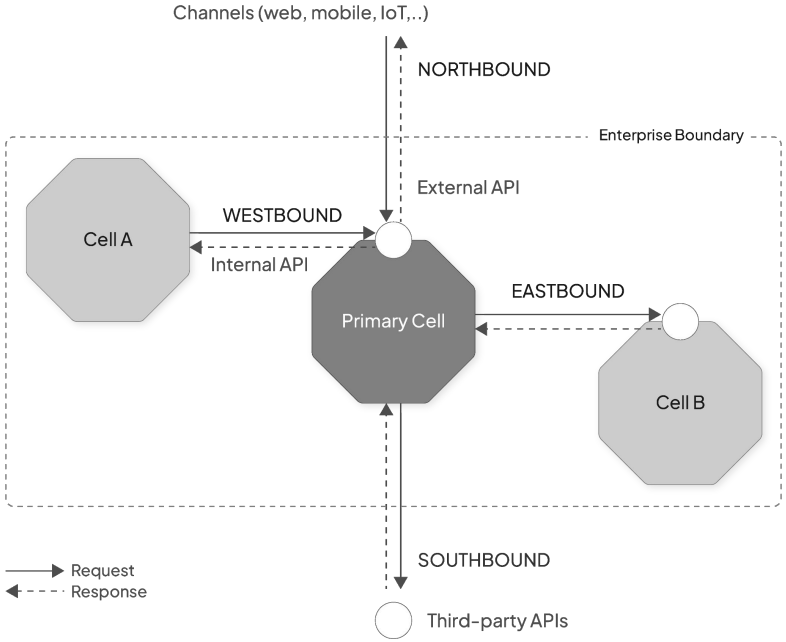
- Network boundary of microservices controlled by a cell
- Cell has two ingress gateways
  - Northbound to allow traffic from the Internet =>Public APIs
  - Westbound to allow traffic from other cells within the organisation ⇒ Organization Scoped APIs
- Component-to-component access within the same project ⇒ Project Scoped APIs

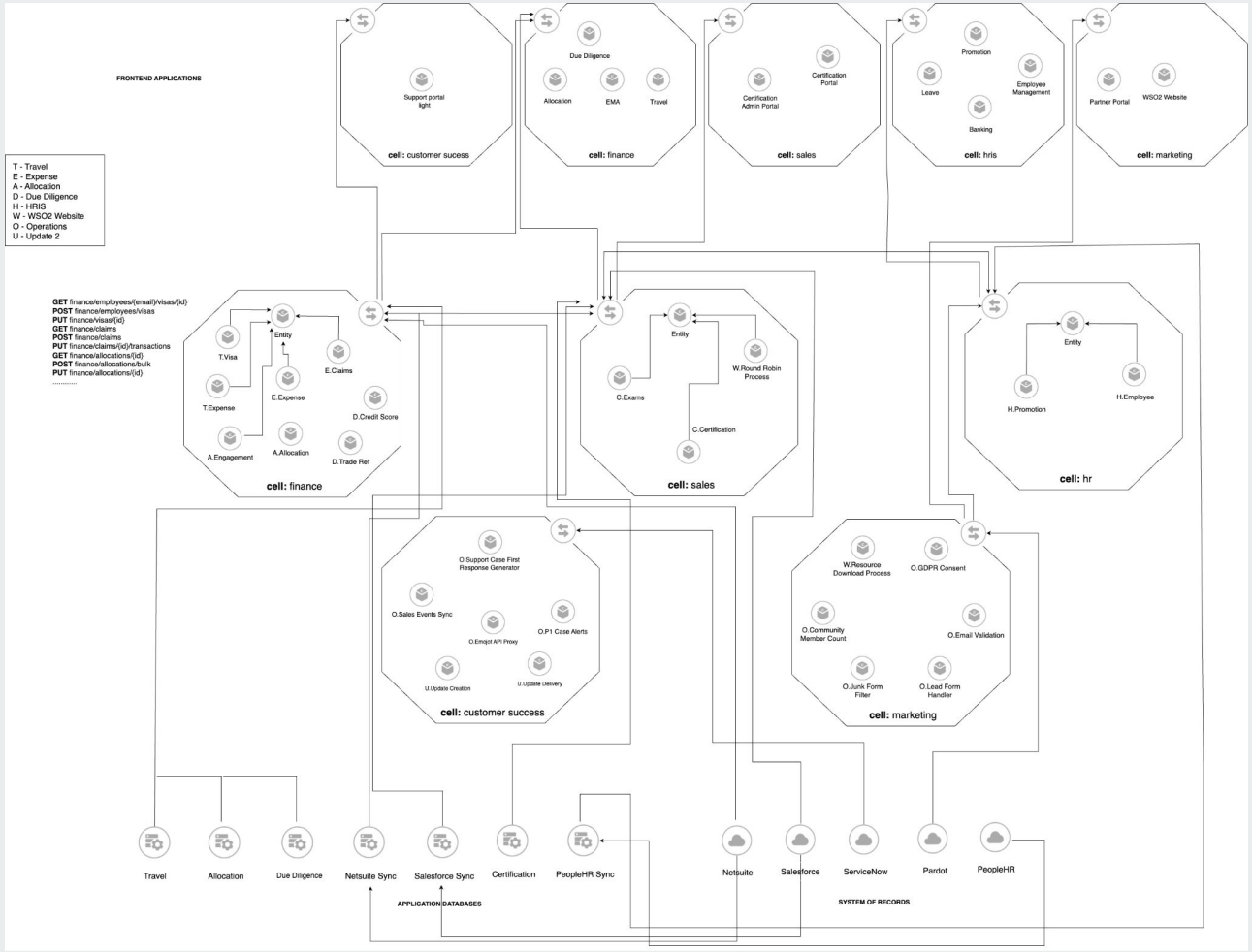


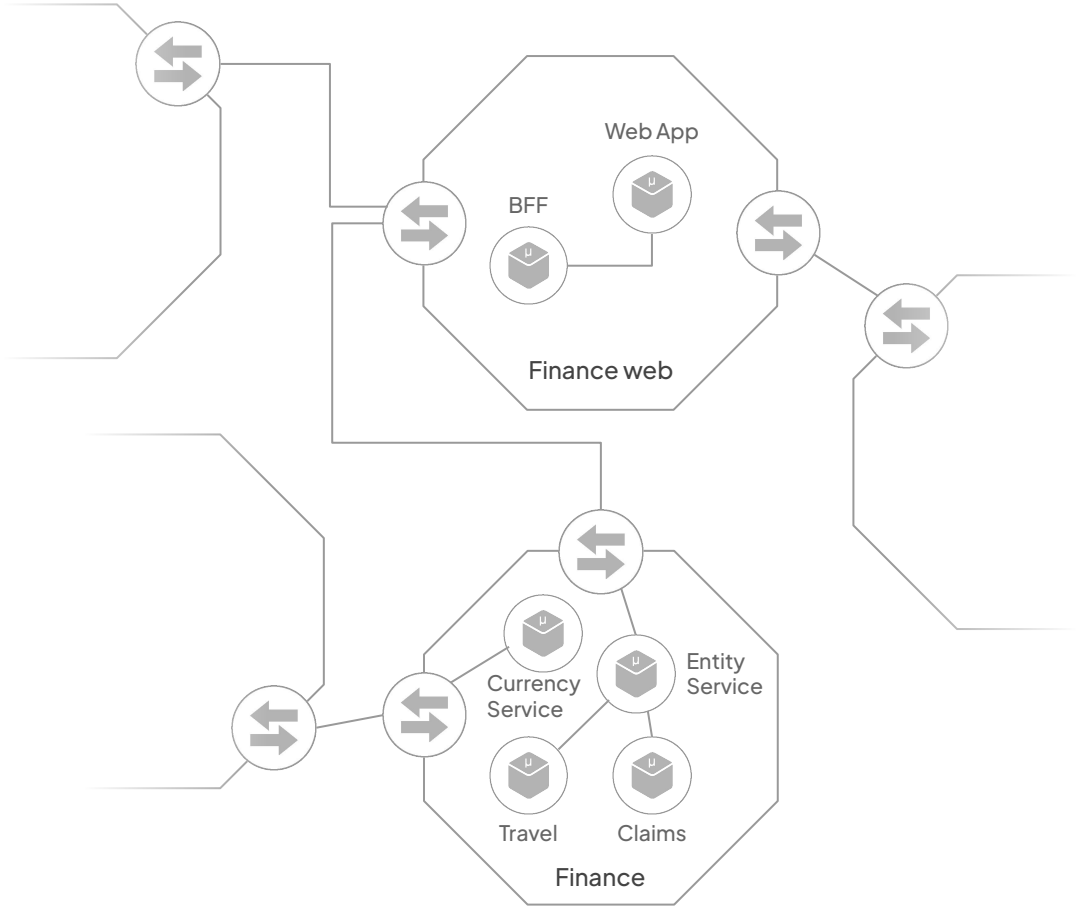

https://github.com/wso2/reference-architecture/blob/master/reference-architecture-cell-based.md

# Cells, Access Control (Cont…)

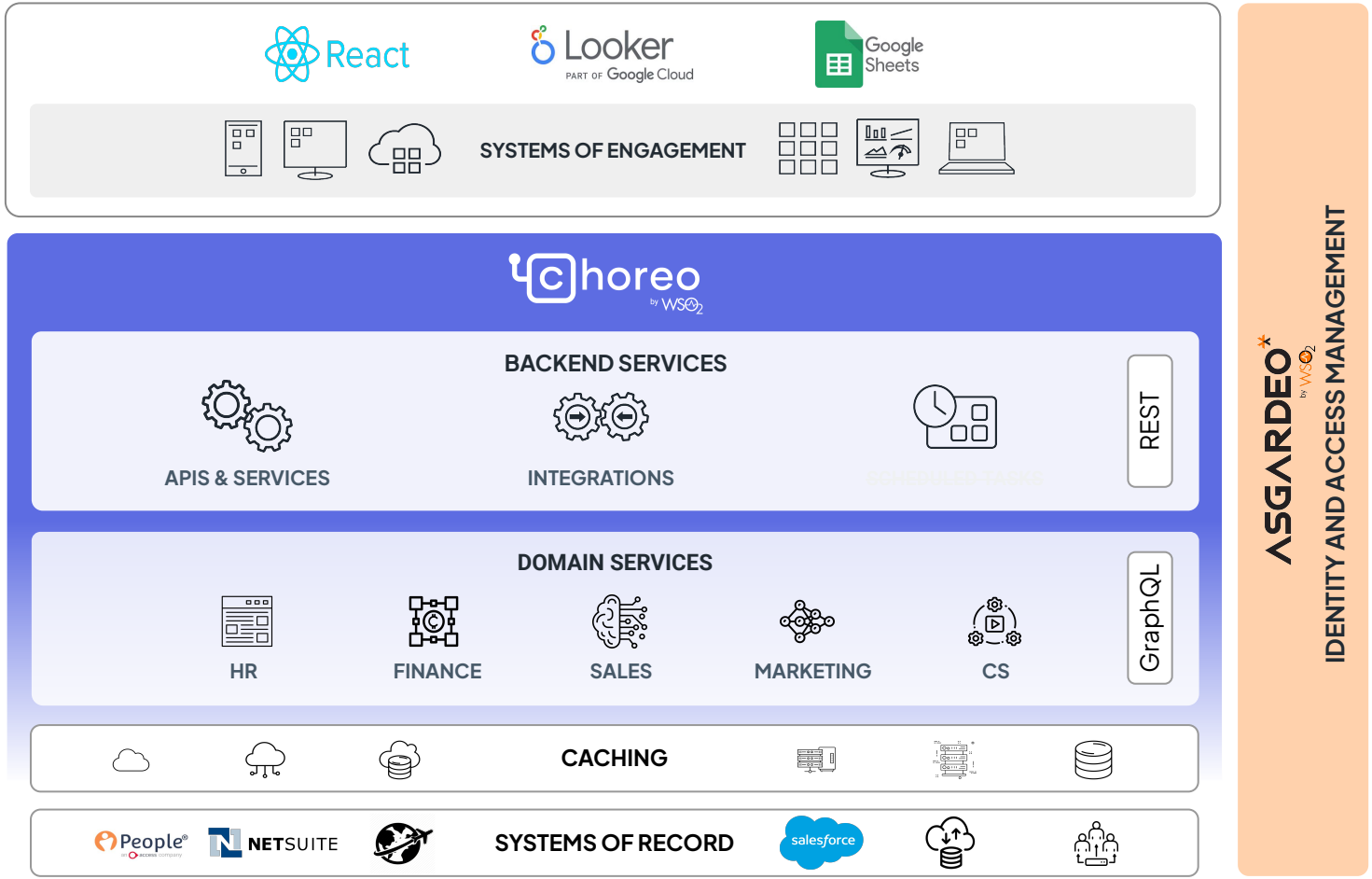
- Cell has two egress control
  - East bound: Traffic to other projects within the organization
  - South bound: Traffic to the Internet
- This will help to control the usage of 3rd party APIs by components
  - Better governance, monitoring and monetization




FRONTEND APPLICATIONS
T - Travel
E - Expense
A - Allocation
D - Due Diligence
H - HRIS
W - WSO2 Website
O - Operations
U - Update 2
Support portal light
cell: customer sucess
Due Diligence
Allocation
EMA
Travel
cell: finance
Certification Admin Portal
Certification Portal
cell: sales
Promotion
Leave
Employee Management
Banking
cell: hris
Partner Portal
WSO2 Website
cell: marketing
GET finance/employees/{email}/visas/{id}
POST finance/employees/visas
PUT finance/visas/{id}
GET finance/claims
POST finance/claims
PUT finance/claims/{id}/transactions
GET finance/allocations/{id}
POST finance/allocations/bulk
PUT finance/allocations/{id}
T.Visa
Entity
E.Claims
T.Expense
E.Expense
D.Credit Score
A.Engagement
A.Allocation
D.Trade Ref
cell: finance
Entity
W.Round Robin Process
C.Exams
C.Certification
cell: sales
Entity
H.Promotion
H.Employee
cell: hr
O.Support Case First Response Generator
O.Sales Events Sync
O.Emqtt API Proxy
O.P1 Case Alerts
U.Update Creation
U.Update Delivery
cell: customer success
W.Resource Download Process
O.GDPR Consent
O.Community Member Count
O.Email Validation
O.Junk Form Filter
O.Lead Form Handler
cell: marketing
Travel
Allocation
Due Diligence
Netsuite Sync
Salesforce Sync
Certification
PeopleHR Sync
APPLICATION DATABASES
Netsuite
Salesforce
ServiceNow
Pardot
PeopleHR
SYSTEM OF RECORDS


Web App
BFF
Finance web
Currency Service
Entity Service
Travel
Claims
Finance

# The Platformless Approach


React
Looker
PART OF Google Cloud
Google Sheets
SYSTEMS OF ENGAGEMENT
choreo
by WSO2
BACKEND SERVICES
APIS & SERVICES
INTEGRATIONS
SCHEDULED TASKS
REST
DOMAIN SERVICES
HR
FINANCE
SALES
MARKETING
CS
GraphQL
CACHING
People
NETSUITE
SYSTEMS OF RECORD
salesforce
ASGARDEO
by WSO2
IDENTITY AND ACCESS MANAGEMENT

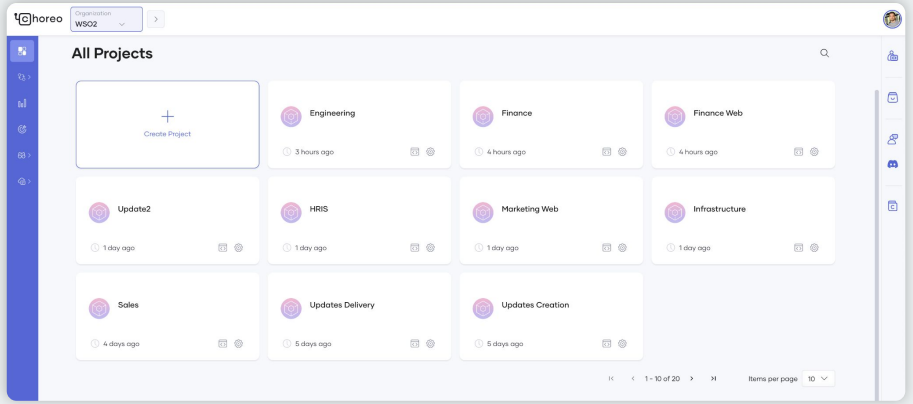
# The Deployment

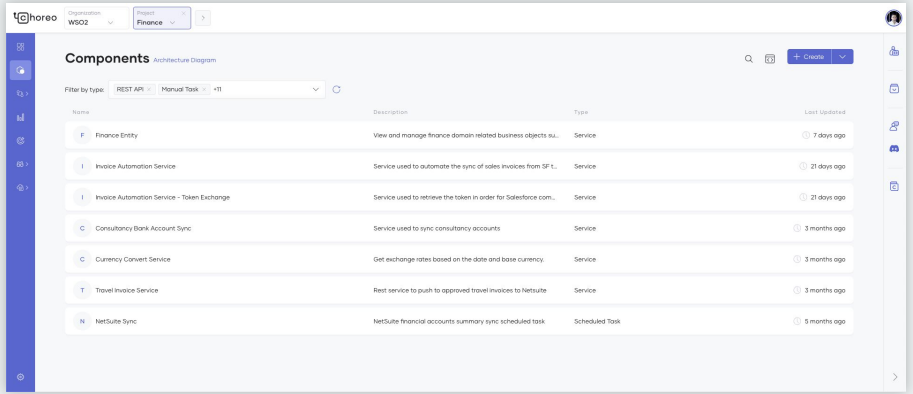
# Flexibility of monorepo and polyrepo, multiple versions, deployment tracks

- Support for Github
- Source repository can be attached at the project level or at the component level
- A component can have multiple versions
  - Multiple versions of an API are supported at the same time
  - Consumers can move from the version they are using to latest version in their own time
  - Intelligent routing of the APIs (e.g can use version as v1, v1.1 or latest)
- Deployment track defines how different versions are created
  - Project level for mono repo cases ⇒ All components of the project will inherit the same version
  - Component level for poly repo cases ⇒ Different components can have different versions

# Projects

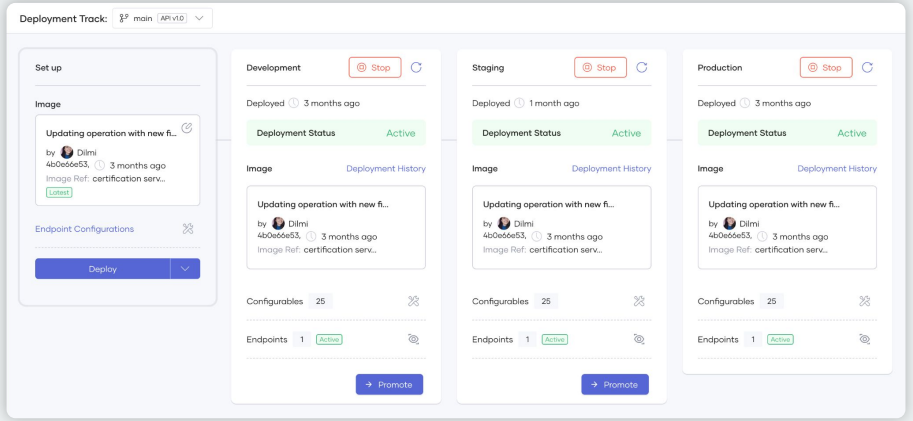
- Related components were grouped together into a single project
  - Entity services, integration services of each domain are in a single project
  - Each webapp and its backend are in a separate project
- One domain has multiple projects. One project belongs to a single domain.
- One project contains multiple components.
- One component belongs to a single project.



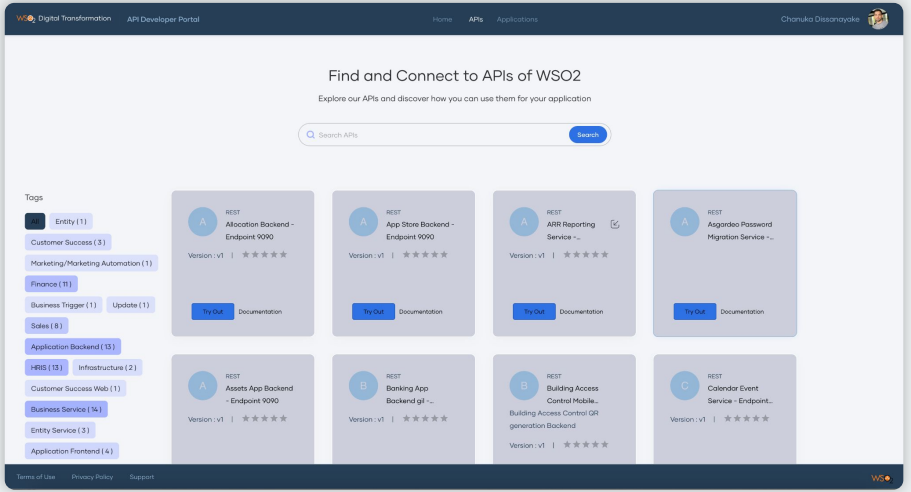
# Environments

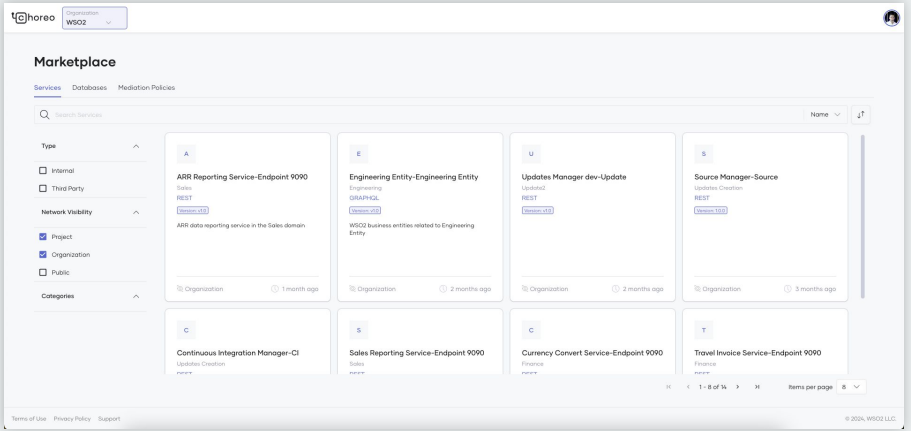
- Support for multiple environments
  - E.g Dev, Staging, Prod
- Ability to define new environments based on need
- Supports for shared environments for development, long running testings, before promoting to production.



# Developer Portal

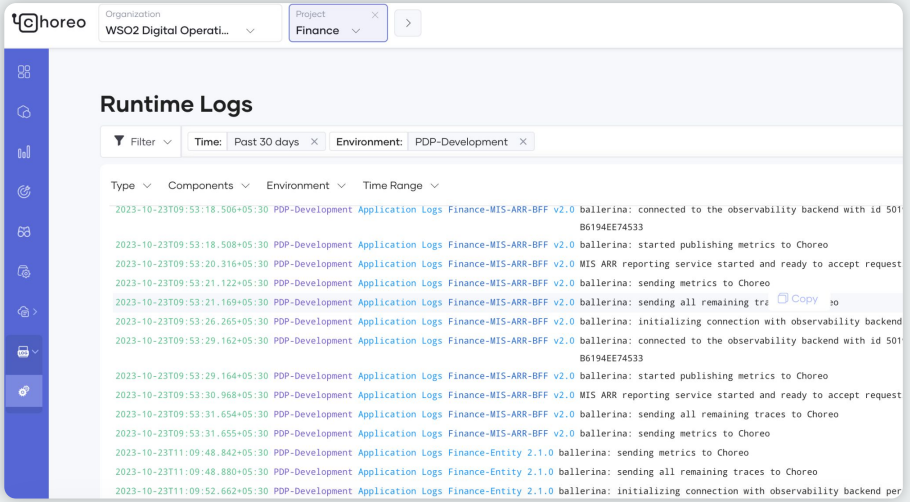
- Discover available APIs ⇒ Increase reusability
  - Documentation
  - Samples
  - Access keys
  - Tryit
- Multiple versions of an API
- Marketplace
  - Control which APIs, 3rd party APIs are available for a particular project for reuse
  - Also, can see organization scope and project scope APIs

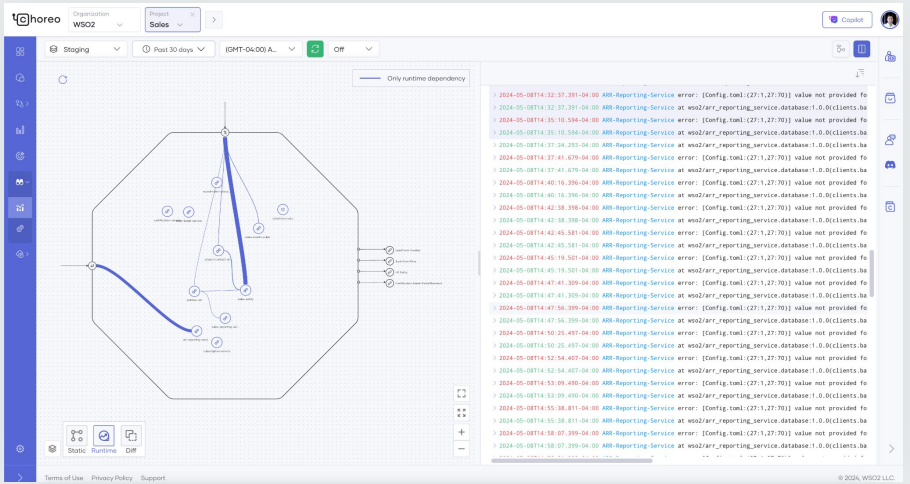


# Troubleshooting, observability, logs

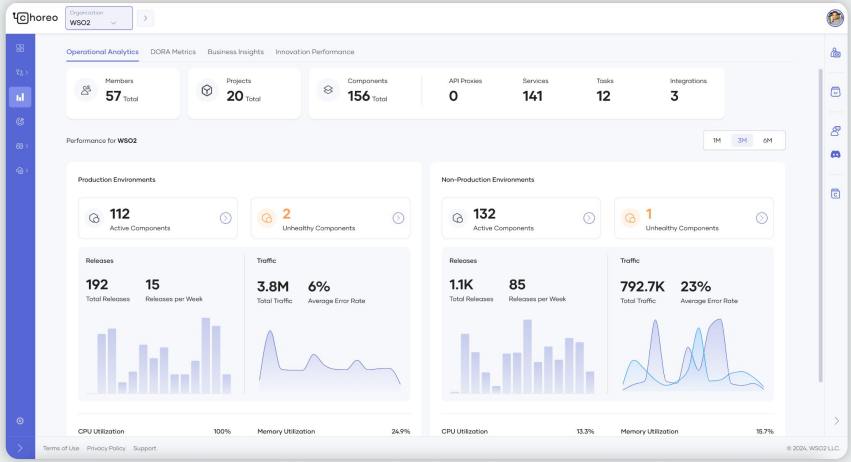
- Application Logs, observability support at various levels
  - Component level
  - Project level
  - Organization level
- Advanced Devops view for container level information
- Audit Logs

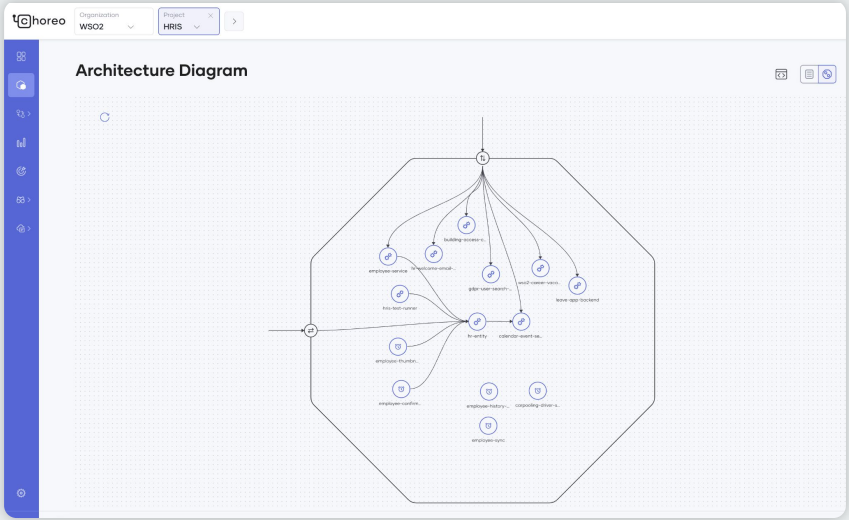


# Insights and architecture diagrams

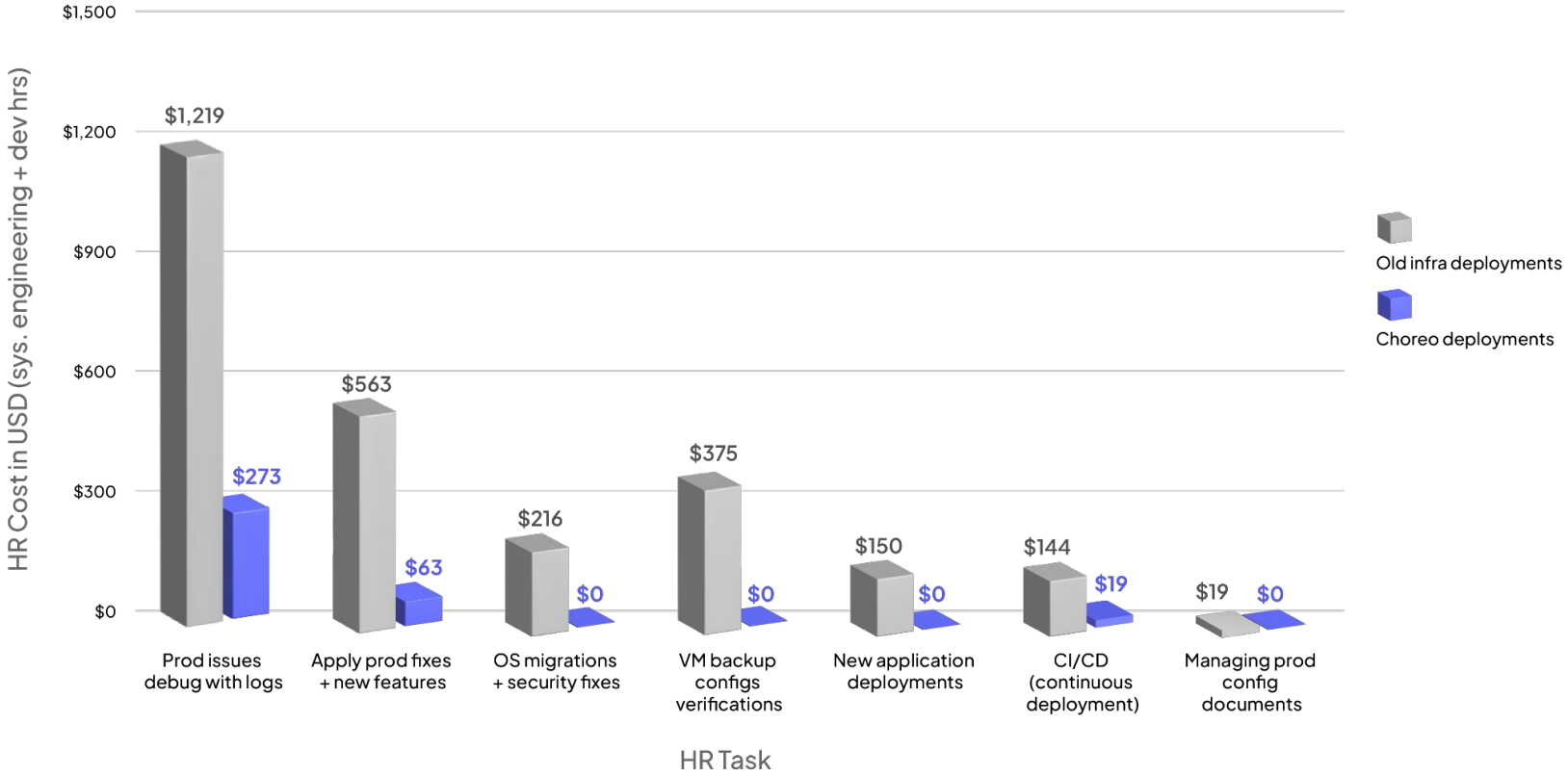
- API and usage insights
  - Which APIs are heavily used
- Development and deployment insights
  - Statistics on how many developers contributing, number of releases, DORA metrics
- Architecture diagram showing how the components interact with each other



# What We Gained

# Cost Reduction



# What we have achieved:



### Reduced Complexity and Cost:

Simplified API development and management in the cloud, reducing costs.



### Enhanced Security and Performance:

Improved API and service security, performance, and reliability with API gateway proxy and Asgardeo identity management.



### Faster Software Delivery:

Enabled quicker delivery of high-quality software.



### Improved Visibility and Governance:

Enhanced API and service visibility, observability, and governance with API insights and observability features.



### Unified Platform Across Environments:

Provided a consistent API development and management platform across multiple environments and clouds.



### Facilitated Collaboration and Integration:

Streamlined collaboration, reuse, and integration of APIs and services through domain model, discovery, governance features, and marketplace.

Choreo has completely changed the game for us, setting us up for big wins in the digital age. Let's embrace it and keep pushing forward!

# Thank You!