

## OSTIA – Documentazione Tecnico-Funzionale (Work in Progress)

### 1. Obiettivo del progetto

Costruire un'architettura cloud multi-tenant capace di:

- gestire autenticazione centralizzata (user → tenant → client),
- emettere JWT con claims strutturati,
- autorizzare l'accesso a servizi tramite entitlements dinamici configurabili lato DB,
- distribuire un client Windows installabile che si configura solo con credenziali.

### 2. Stack Tecnologico Google Cloud

- Cloud Run: servizio ostia-auth
- Identity Platform: autenticazione email/password
- Cloud SQL PostgreSQL: database ostia\_core
- Cloud Build + Artifact Registry: build e deploy container

### 3. Struttura JWT

Claim: sub, email, tenant\_id, client\_id, iat, exp (HS256)

### 4. Struttura DB ostia\_core

Tabelle: tenants, users, clients, services, entitlements

Tenant demo: TENANT\_001

Client demo: CLIENT\_001

User demo: user@client1.it

Servizi: s1, s2, s3

### 5. ostia-auth (Cloud Run)

Endpoint:

- POST /login
- GET /me

Funzioni: valida Identity Platform → controlla DB → genera JWT → restituisce claims.

## 6. Client

Versioni:

- CLI: login/me
- GUI: configurazione via interfaccia, salvataggio in %APPDATA%/OstiaClient

EXE standalone via PyInstaller.

## 7. Comando build EXE

```
pyinstaller --onefile --noconsole --name OstiaClient ostia_gui_client.py
```

## 8. Stato attuale

- Autenticazione funzionante
- DB popolato
- Client GUI funzionante
- EXE generabile
- Installer da creare (fase finale)