

Deception component generator

Rest API server

Luca **Dominici**
Enrico **Zacchioli**
Lorenzo **Ziosi**



form an **OpenAPI** specification
implement a dumb service with
fake/random data

accept configurations for **OAuth**
authentication (at generation time) to
secure the API endpoint and provide
an out of the box functionality

Requirements

Risorse API

Potenzialmente **vulnerabili** ad una ampia gamma di rischi alla sicurezza

Utenti non autorizzati possono guadagnare l'accesso a dati sensibili.

Attacchi di tipo injection e DDoS per sovraccaricare le API con grandi quantità di traffico.

Necessità di proteggere le API:

Protocollo di autorizzazione

OAuth

Protocollo standard per quanto riguarda l'**autorizzazione**

Permette ad un client di terze parti di ottenere accesso limitato a delle risorse messe a disposizione dal resource owner

Emissione di un **token di accesso** da parte di un server autorizzativo ad un client di terze parti



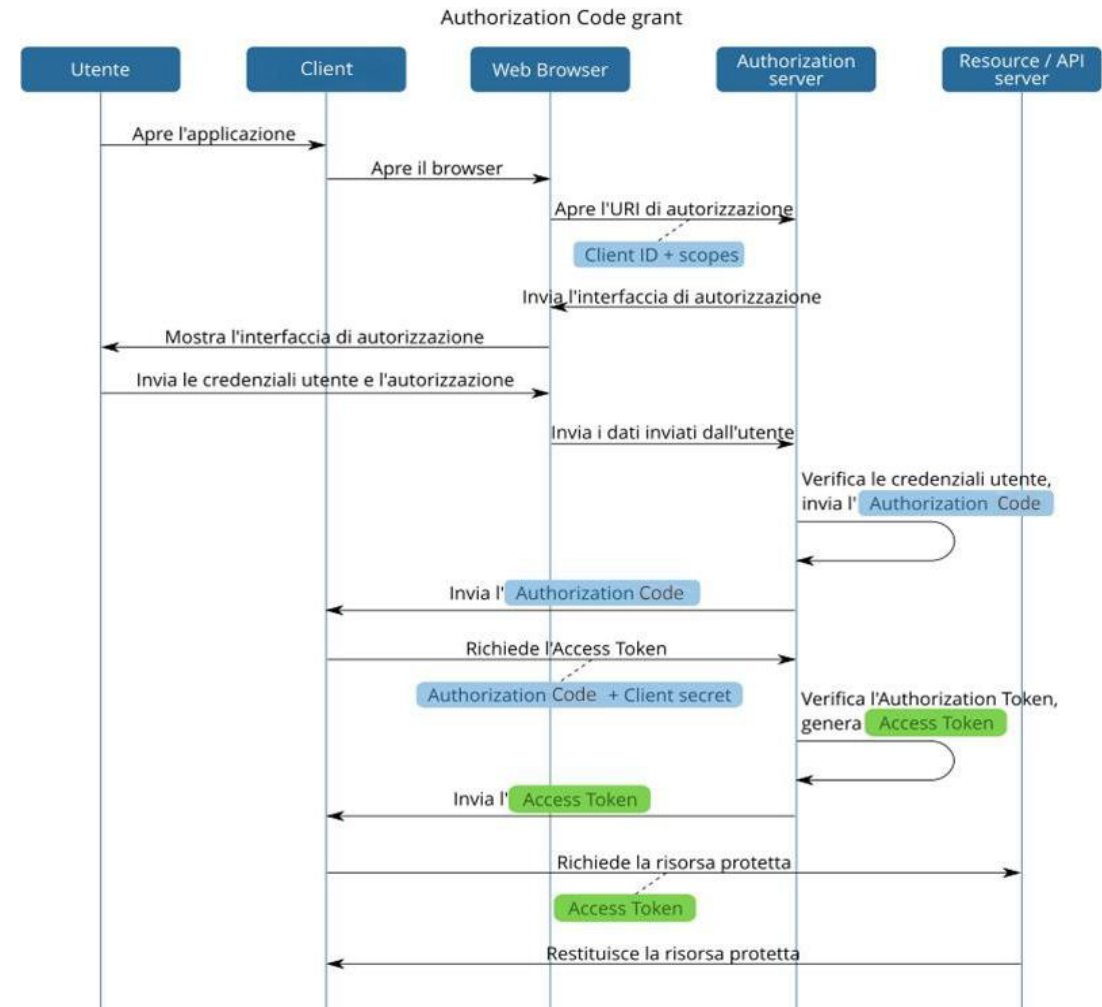
Authorization Code Flow

Authorization code

Il client ottiene l'autorizzazione code dall'auth server, che serve come autorizzazione temporanea.

Browser redirect

Token exchange



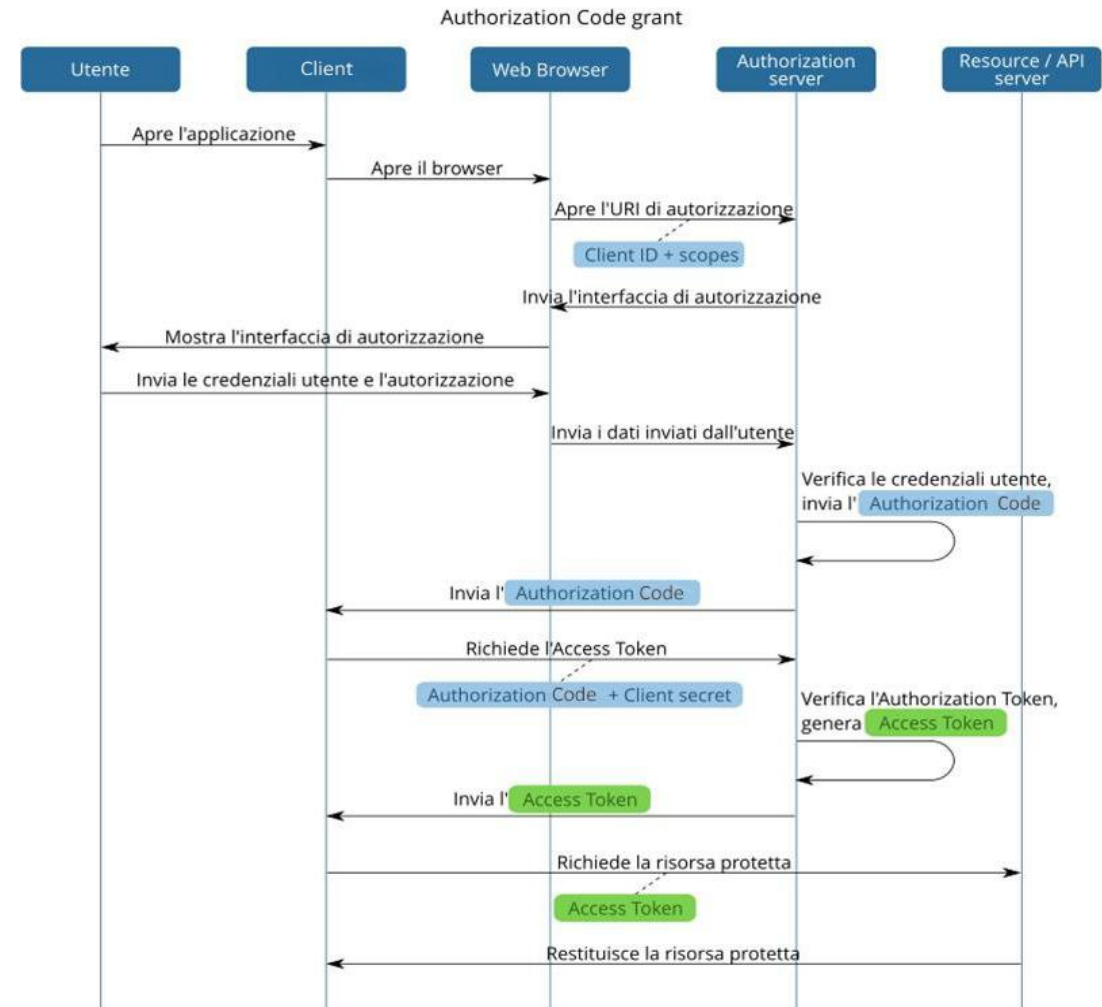
Authorization Code Flow

Authorization code

Browser redirect

Dopo l'autorizzazione dell'utente, l'auth server reindirizza il browser dell'utente al client con il codice di autorizzazione nell'URL di redirectione.

Token exchange



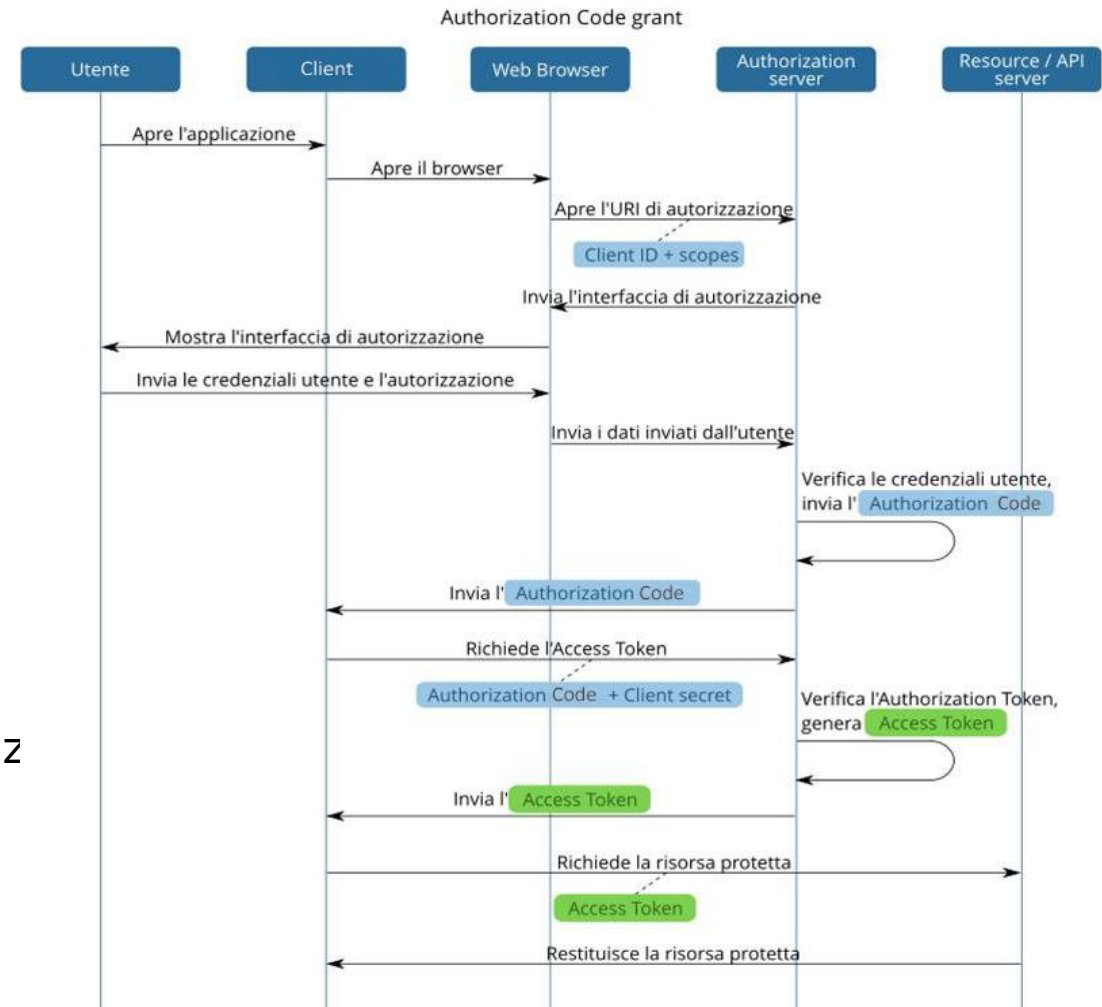
Authorization Code Flow

Authorization code

Browser redirect

Token exchange

Il client scambia l'autorization code con l'auth server per ottenere il token di accesso, garantendo una maggiore sicurezza durante il processo di autenticazione.



PKCE

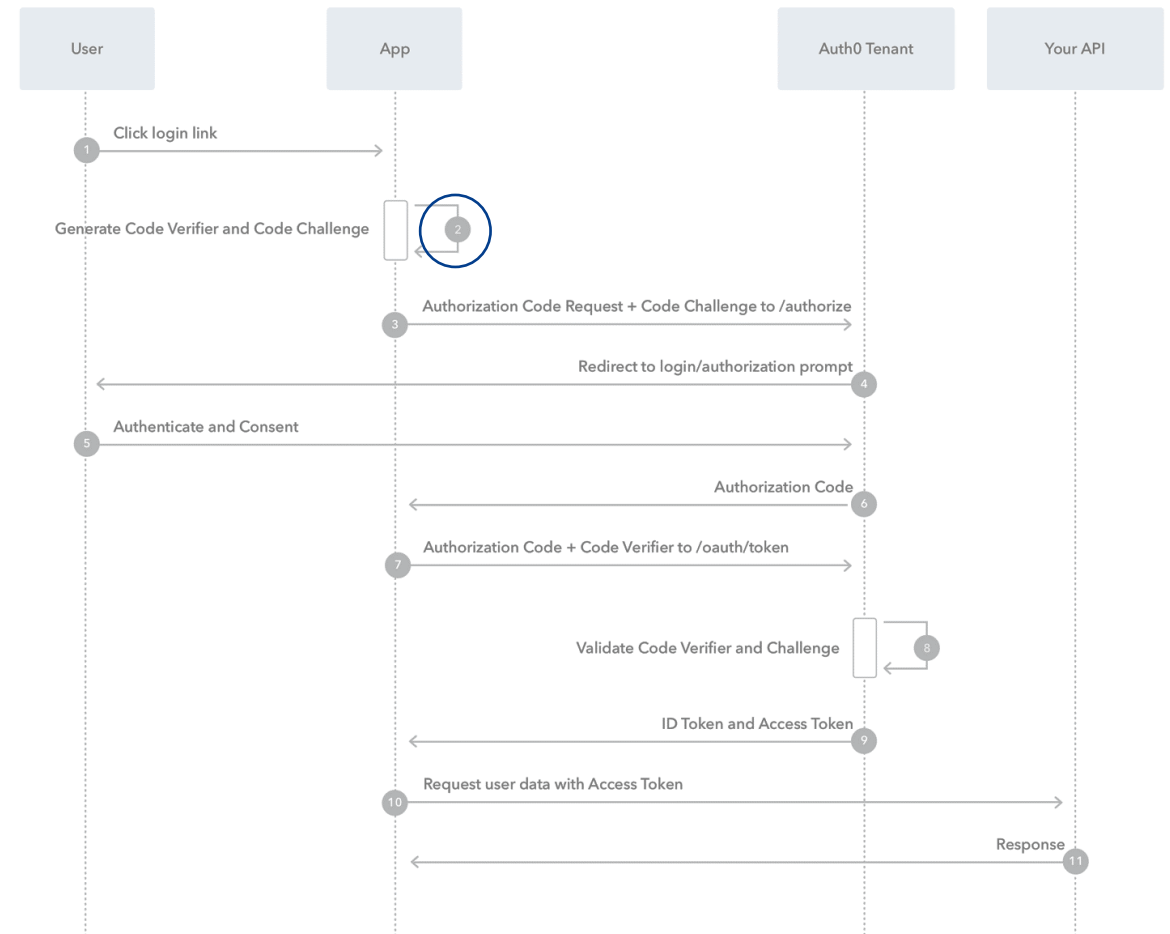
Proof Key for Code Exchange

2. Generazione

Il client genera un code verifier in modo casuale e ne deriva un code challenge

3. Invio code challenge

7. Invio code verifier



PKCE

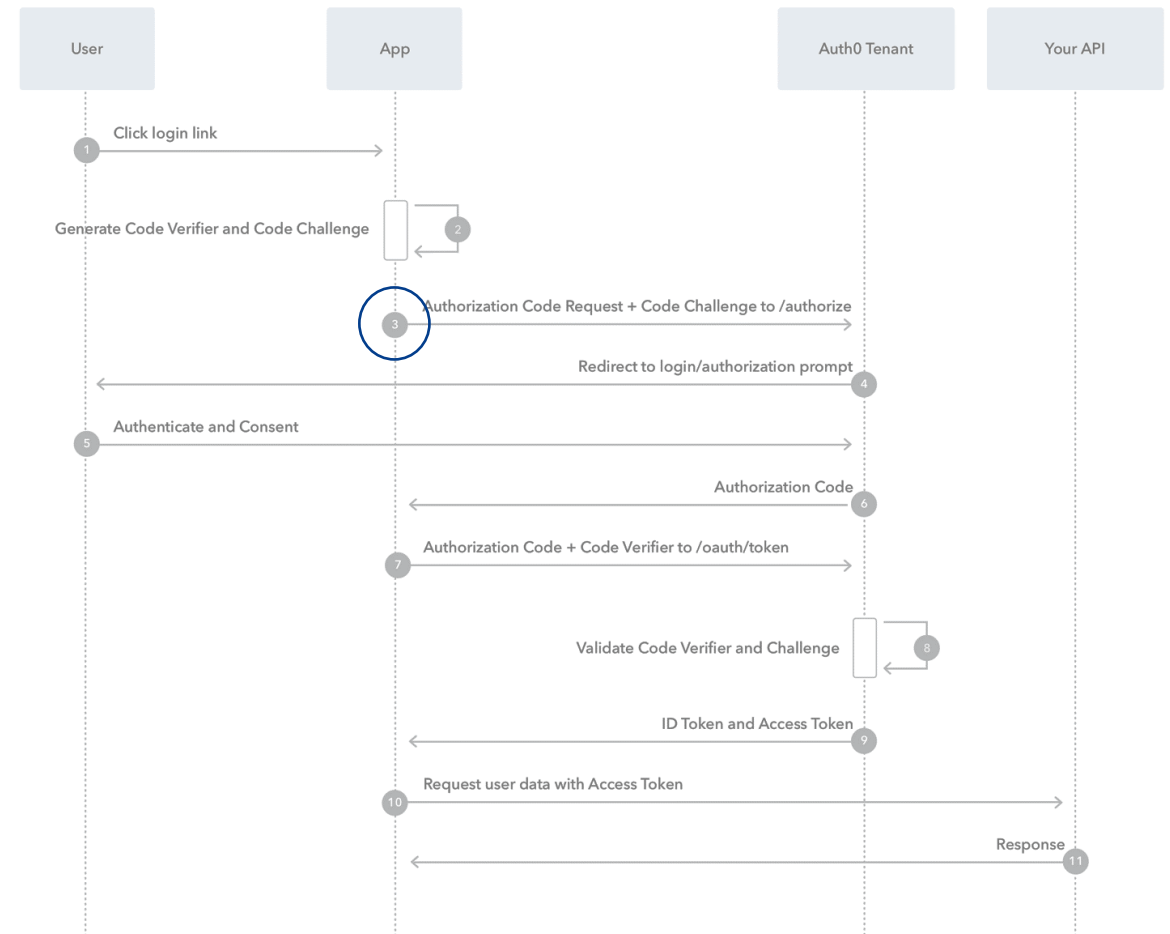
Proof Key for Code Exchange

2. Generazione

3. Invio code challenge

Durante la richiesta dell'autorization code viene inviato il code challenge che viene salvato dall'auth server

7. Invio code verifier



PKCE

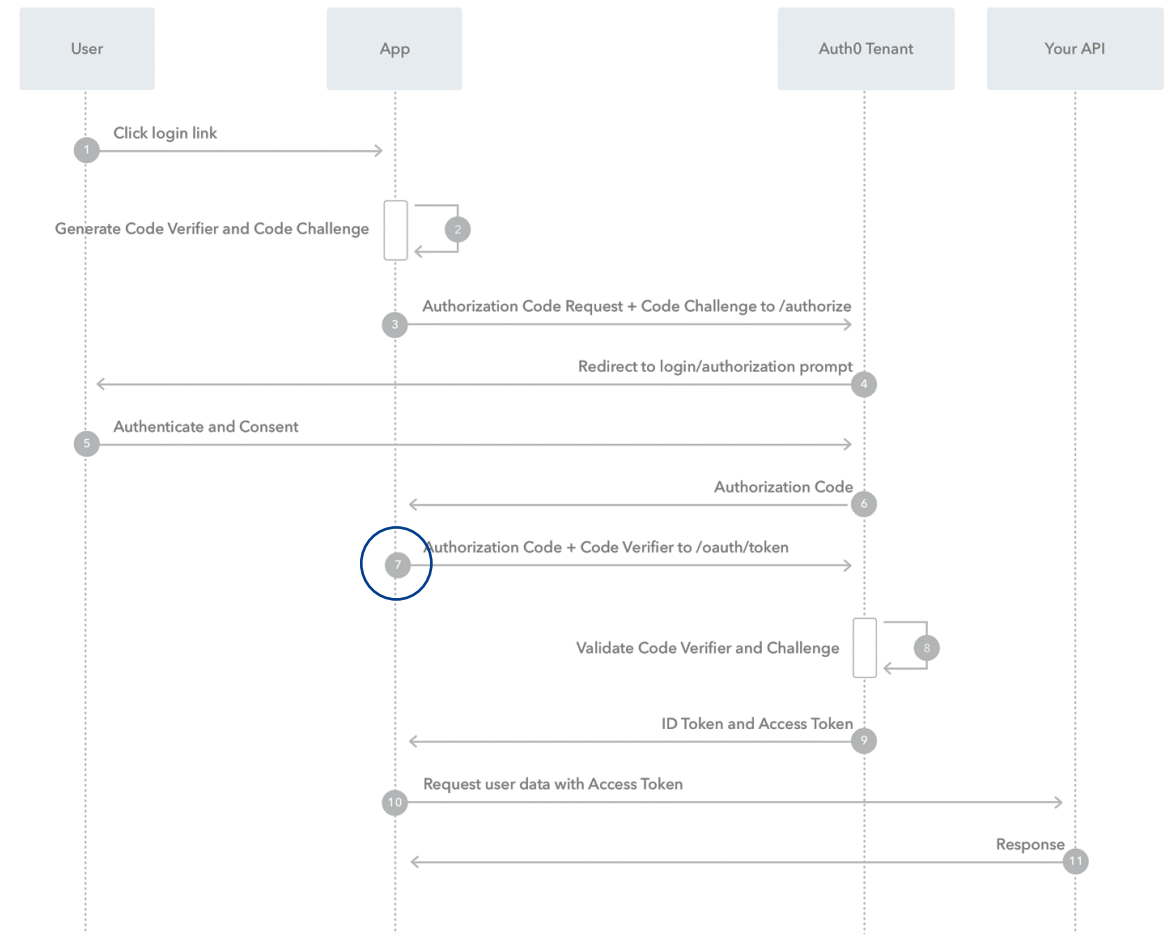
Proof Key for Code Exchange

2. Generazione

3. Invio code challenge

7. Invio code verifier

Durante la richiesta per l'access token viene inviato il code verifier, utilizzato per ricalcolare il code challenge e confrontarlo con quello iniziale



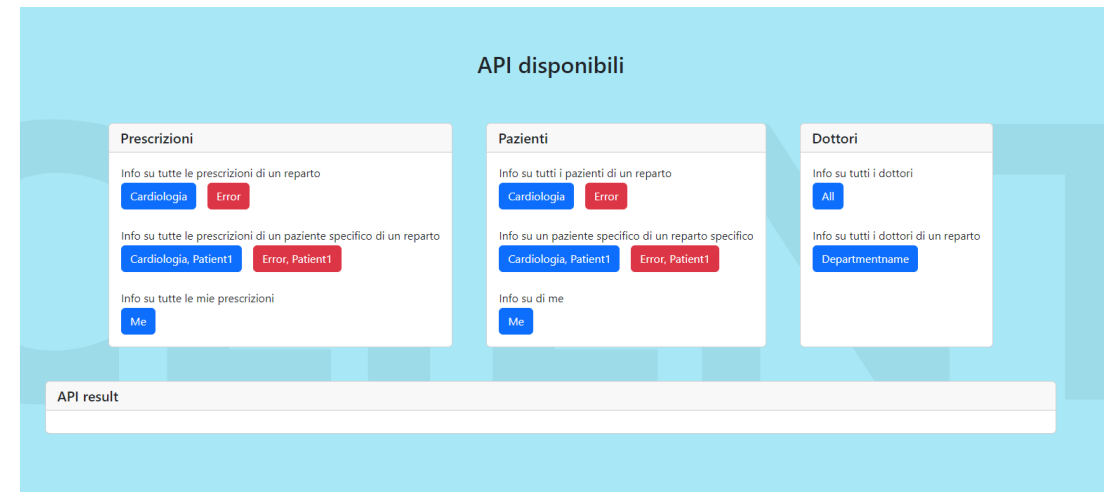
Servizi

Client

Utilizzatore dell'API server. È riconosciuto dall'Authorization server, al quale richiede l'autorizzazione per accedere alle risorse.

Authorization Server

API Server



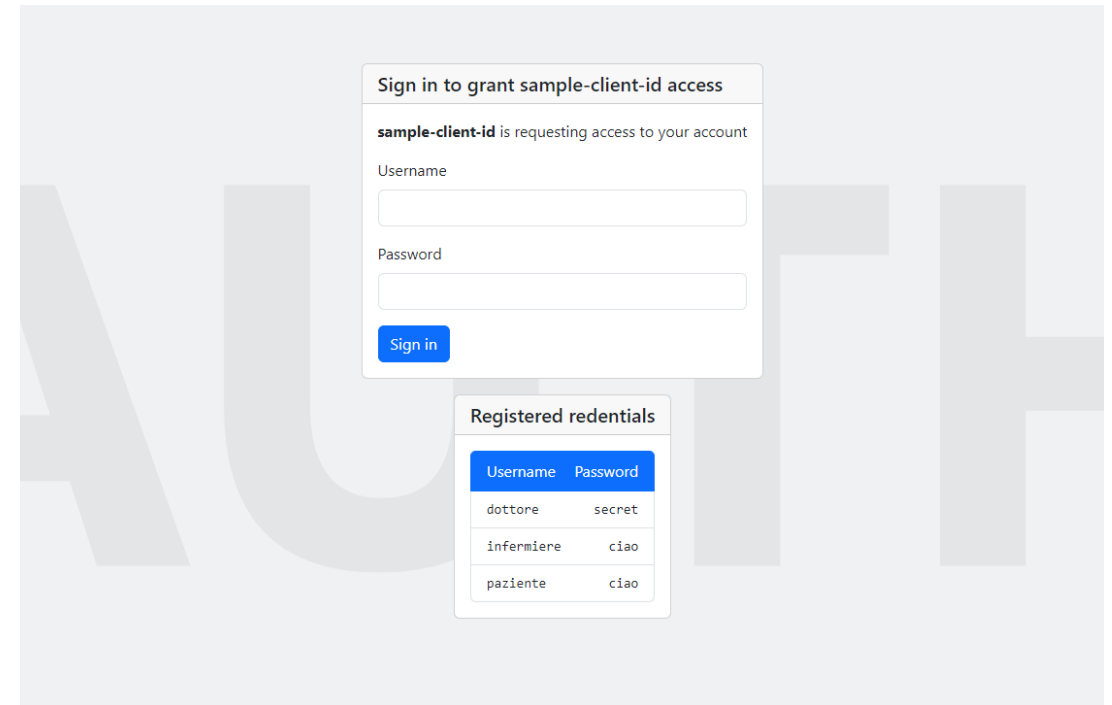
Servizi

Client

Authorization Server

Permette l'identificazione dell'utente. In cambio dell'Authorization Code fornisce l'Access Token al client per l'accesso alle risorse.

API Server



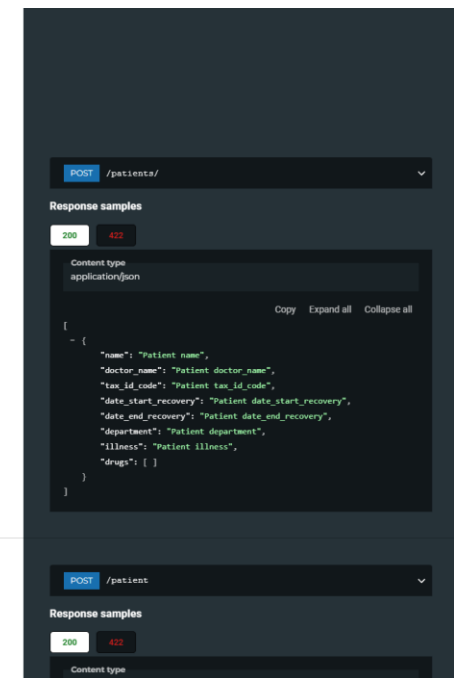
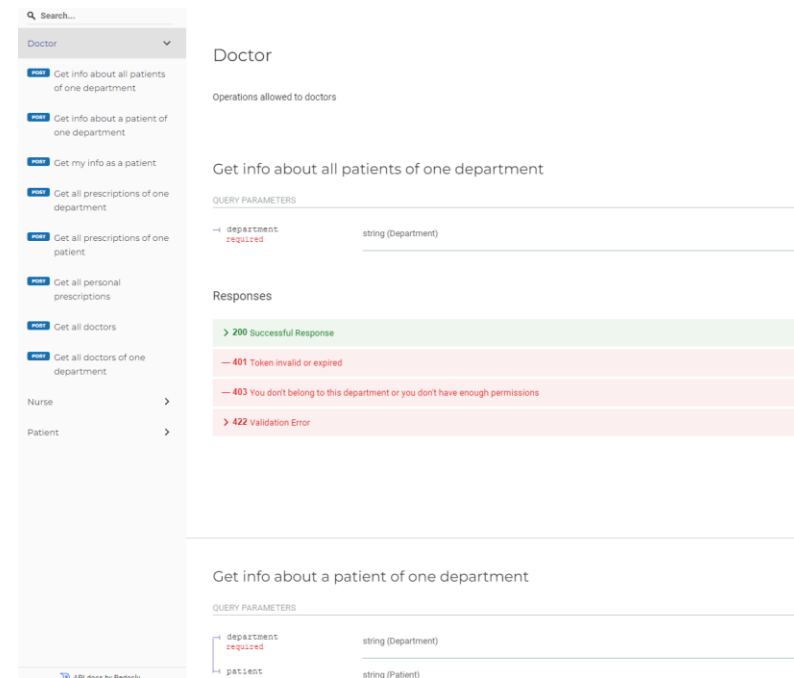
Servizi

Client

Authorization Server

API Server

Fornisce le risorse API utilizzate dal client, corrisponde all'entità più critica del sistema. Come da requisiti soddisfa le specifiche OpenAPI.










Utilizzo di Docker

Struttura

Suddivisione in tre container, ognuno contenente un diverso componente.

`docker-compose`

	restapiserver	Running (3/3)	0.46%	
	apiclient de6b846f1fb9	Running	0.1%	5000:5000 
	apiserver 34b5ab925089	Running	0.27%	8000:8000 
	authserver f5cb6f97ddec	Running	0.09%	5001:5001 

Utilizzo di Docker

Struttura

docker-compose

Singolo file per l'avvio dell'intero sistema e creazione network.
Immagini reperibili su Docker hub.

```
version: '3'
services:
  client:
    image: zacken1999/apiclientimage:1.0
    container_name: apiclient
    ports:
      - 5000:5000
    environment:
      - TOKEN_PATH=http://authserver:5001/token
      - RES_PATH=http://apiserver:8000/
      - VERIFY_TOKEN_PATH=http://authserver:5001/user
  authserver:
    image: zacken1999/authserverimage:1.0
    container_name: authserver
    ports:
      - 5001:5001
  apiserver:
    image: zacken1999/apiserverimage:1.0
    container_name: apiserver
    ports:
      - 8000:8000
    environment:
      - VERIFY_TOKEN_PATH=http://authserver:5001/user
```

Demo

