# Sicurezza e autenticazione

**Non vogliamo che chiunque abbia accesso alle ns API**

**Principali tecniche di protezione endpoint HTTP (ognuno ha PRO e CONTRO)**

**1) Basic HTTP authentication**

**Le credenziali utente (es email e password) sono messe in un header HTTP chiamato Authorization, codificate in Base64. Schema semplice ma non molto sicuro (la password appare in ogni request)**

**2) Cookies**

**Un modo utile per memorizzare dati statici sul client (normalmente il browser) mandati ogni volta sul server.**
**Tipicamente un cookie può contenere un session token (che può essere verificato dal server, legato ad uno specifico utente)**

**3) Token (bearer) authentication nel header Authorization**

**Probabilmente il più usato**
**Semplicemente consiste nel mandare un token nel header Authorization**
**Il nome “Bearer authentication” si può intendere come “dare accesso al portatore di questo token”. Il token può essere verificato lato server e associato a specifico user**

FastAPI

# ... in FastAPI

**Fornite e utilizzabili con il meccanismo delle dependencies**

**Token auth con APIKeyHeader security dependency**
**Progettatata per ottenere un valore dall'header**
**E' una class dependency che può essere instanziata con l'argomento name, il nome dell'header ricercato.**
**Nel ns endpoint inoculeremo questa dipendenza per ottenere il valore della token.**
**Se corrisponde al valore atteso, bene, altrimenti errore 403**
**Usando la dependency, viene anche rilevata in OpenAPI docs**

*Esempi: api_key_header.py e fastcash app*

FastAPI