

# Authentication and authorization



JWT



Authentication  
vs  
Authorization



Prototypes

## JSON Web Tokens



- JSON Web Tokens (or JWTs) provide a means of transmitting information from the client to the server in a stateless, secure way.
- Generated by signing user information via a secret key

Flow

Example

## How?



- Client provides email and password, which is sent to the server
- Server then verifies that email and password are correct and responds with an auth token
- Client stores the token and sends it along with all subsequent requests to the API
- Server decodes the token and validates it
- This cycle repeats until the token expires or is revoked. In the latter case, the server issues a new token

## JSON Web Tokens



- JSON Web Tokens (or JWTs) provide a means of transmitting information from the client to the server in a stateless, secure way.
- Generated by signing user information via a secret key

Flow

Example

## **pip install pyjwt**

```
def encode_token(self):
    try:
        payload = {
            'exp': datetime.utcnow() + timedelta(days=2),
            'sub': self.id
        }
        return jwt.encode(
            payload,
            key=config('SECRET_KEY'),
            algorithm='HS256'
        )
    except Exception as e:
        raise e
```

## **pip install pyjwt**

```
def encode_token(self):
    try:
        payload = {
            'exp': datetime.utcnow() + timedelta(days=2),
            'sub': self.id
        }
        return jwt.encode(
            payload,
            key=config('SECRET_KEY'),
            algorithm='HS256'
        )
    except Exception as e:
        raise e
```

## JSON Web Tokens



- JSON Web Tokens (or JWTs) provide a means of transmitting information from the client to the server in a stateless, secure way.
- Generated by signing user information via a secret key

Flow

Example

# Authentication and authorization



JWT



Authentication  
vs  
Authorization




Prototypes



## Definition

- While authentication and authorization are often used interchangeably, they are separate processes used to protect an organization from cyber-attacks.
- As data breaches continue to escalate in both frequency and scope, authentication and authorization are the first line of defense to prevent confidential data from falling into the wrong hands.
- As a result, strong authentication and authorization methods should be a critical part of every application's overall security strategy

## Differences

- 
- Authentication is the process of verifying who someone is.
  - Authentication works through passwords, one-time pins, biometric information, and other information provided or entered by the user.
  - Authentication is the first step of a good identity and access management process
  - Authentication is visible to and partially changeable by the user.

- Authorization is the process of verifying what specific applications, files, and data a user has access to
- Authorization works through settings that are implemented and maintained by the organization.
- Authorization always takes place after authentication.
- Authorization isn't visible to or changeable by the user.

## Definition

- While authentication and authorization are often used interchangeably, they are separate processes used to protect an organization from cyber-attacks.
- As data breaches continue to escalate in both frequency and scope, authentication and authorization are the first line of defense to prevent confidential data from falling into the wrong hands.
- As a result, strong authentication and authorization methods should be a critical part of every application's overall security strategy

## Differences

# Authentication and authorization



JWT



Authentication  
vs  
Authorization



Prototypes

## Prototypes

```
@app.get("/clothes/", dependencies=[Depends(oauth2_scheme)])
async def get_all_clothes():
    return await database.fetch_all(clothes.select())

@app.post(
    "/clothes/",
    response_model=ClothesOut,
    status_code=201,
    dependencies=[Depends(oauth2_scheme), Depends(is_admin)]
)
async def create_clothes(clothes_data: ClothesIn):
    id_ = await database.execute(clothes.insert().values(**clothes_data.dict()))
    return await database.fetch_one(clothes.select().where(users.c.id == id_))
```