JSON Web Token (JWT)

- **JWT** - открытый стандарт (RFC 7519) передачи защищенной информации (токенов) для целей аутентификации (claims authentication) и обмена информацией.

- Официальный сайт https://jwt.io/

- **Авторизация**: все запросы аутентифицированного пользователя включают в себя JWT, что позволяет ему получать доступ к запрашиваемым ресурсам. Благодаря использованию единой точки авторизации (Single Sign On) возможно выполнение кросс-доменных запросов.

- **Обмен информацией**: JWT хорошо подходя для надежного и защищенного обмена информацией между участниками, с проверкой цифровых подписей.

Структура JSON Web Token

- Структура JWT состоит из 3 частей разделенных точками
- xxxxx.yyyyy.zzzzzz
  - Заголовок (header)
  - Нагрузка (payload)
  - Подпись (signature)
- **Заголовок**

```
{
        "alg": "HS256",
        "typ": "JWT"
}
```

- **Нагрузка** (состоит из набора данных claims, кодируется в base64):

```
{
        “sub”     :        “1224”
        “name”    :        “admin”
}
```

Цифровая подпись (генерируется например так:

HMACSHA256(base64UrlEncode(header) + "." + base64UrlEncode(payload), secret))

Наборы данных (Claims)

Registered claims: определенные заранее наборы данных, которые рекомендуется использовать, например: **iss** (издатель), **exp** (срок жизни), **sub** (сущность), **aud** (аудитория) и т.д.

Public claims: определяются создателями JWT для целей использования. Чтобы избежать дублирования именований, их следует регистрировать в IANA JSON Web Token Registry или именовать в виде URI.

Private claims: собственные данные используемые только участниками и нигде не регистрируемые.

Пример JWT

Encoded

```
eyJhbGciOiJIUzI1NiIsInR5cCI6
IkpXVCJ9.eyJzdWIiOiIxMjM0NTY
3ODkwIiwibmFtZSI6IkpvaG4gRG9
lIiwiYWRtaW4iOnRydWV9.TJVA95
OrM7E2cBab30RMHrHDcEfxjoYZge
FONFh7HgQ
```

Decoded

HEADER:

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD:

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret
) ☐secret base64 encoded
```

Виды токенов

- **Access token** – используется для доступа владельцем к защищенному ресурсу, обычно имеет короткий срок жизни
- **Refresh token** – позволяет клиентам запрашивать новые access token, обычно длительный срок жизни

Схема работы

1. Клиент проходит аутентификацию в приложении (например с использованием логина и пароля).
2. В случае успешной аутентификации сервер отправляет клиенту access- и refresh-токены.
3. При дальнейшем обращении к серверу клиент использует (отправляет в заголовке **Authorization: Bearer …**) access-токен. Сервер проверяет токен на валидность и предоставляет доступ к ресурсам.
4. В случае, если access-токен становится не валидным, клиент отправляет refresh-токен, в ответ на которые сервер предоставляет два новых токена.
5. В случае если refresh-токен становится не валидным, клиент должен заново пройти аутентификацию (п. 1).

Реализация на PHP

- Есть различные библиотеки для разных языков и платформ:
- https://jwt.io/#libraries-io

- Например https://github.com/lcobucci/jwt

- Установка: **composer require lcobucci/jwt**