Welcome to the CoGrammar

Task 26: Authentication with JWT

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

Full Stack Web Development Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Full Stack Web Development Session Housekeeping cont.

- For all **non-academic questions**, please submit a query: **www.hyperiondev.com/support**
- Report a **safeguarding** incident: **www.hyperiondev.com/safeguardreporting**
- We would love your **feedback** on lectures: **Feedback on Lectures**

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Lesson Objectives

- Define authentication and its importance in web development
- Discuss common authentication methods such as username/password, OAuth and JWT.
- Highlight the role of JWT in authentication and its benefits
- Implementation of JWT as an authentication method.

Authentication vs Authorization.

CoGrammar

Authentication

Definition and importance

- **Authentication** involves verifying the identity of users to access an application (or website).
- This ensures the security and integrity of online systems by allowing only authorized users to access protected resources.

Authentication

Definition and importance

- Importance of Authentication:
  - **Security:** protection against unauthorized access ensures only authorized individuals can access sensitive information.
  - **User Trust and reputation:** strong authentication builds trust with customers demonstrating an organisation's commitment to security.
  - **Compliance:** Many regulations and laws require organisations to protect sensitive information.

Authentication

Common authentication methods

- Authentication methods:
  - **Username/Password based auth:** Most traditional method where users give the username/email and password for identification.
  - **OAuth:** Use of third party applications to access a user's resources without sharing their credentials.
  - **Token Based Authentication:** Using a unique token to authenticated users to include in subsequent requests to access protected routes.
  - **Multi-factor Authentication (MFA):** Adding an extra layer of security by requiring users to provide multiple forms of verification.

Token based Authentication (JSON Web Tokens)

CoGrammar

JSON Web Tokens (JWT)

Definition and comparison to other authentication methods

- How basic authentication with tokens work:
  - The client sends the username and password to an authentication endpoint
  - The auth endpoint checks the data and if legit, generates an auth token which is relevant to the requesting user's session
  - The client stores the token and adds it to the header of further requests
  - The server checks the token every time it receives a request and uses it to determine which user is making the request.

JSON Web Tokens (JWT)

Definition and comparison to other authentication methods

- In basic authentication, where the username and password were passed in the headers of the url, the password becomes interceptable as it is passed as plain text when you use **(http)** instead of **(https).**
- The use of JWT ensures safety as it transmits information between parties securely in a JSON object.
- JWTs are usually signed, this means you can be certain that the senders are who they say they are.
- Additionally, the structure of a JWT allows you to verify that the content hasn't been tampered with.

Structure of a JWT

- **Header:** Contains the signing algorithm and type of token (JWT)
- **Payload:** Contains the claims or the JSON object
- **Signature:** String generated by cryptographic algorithm to verify integrity.

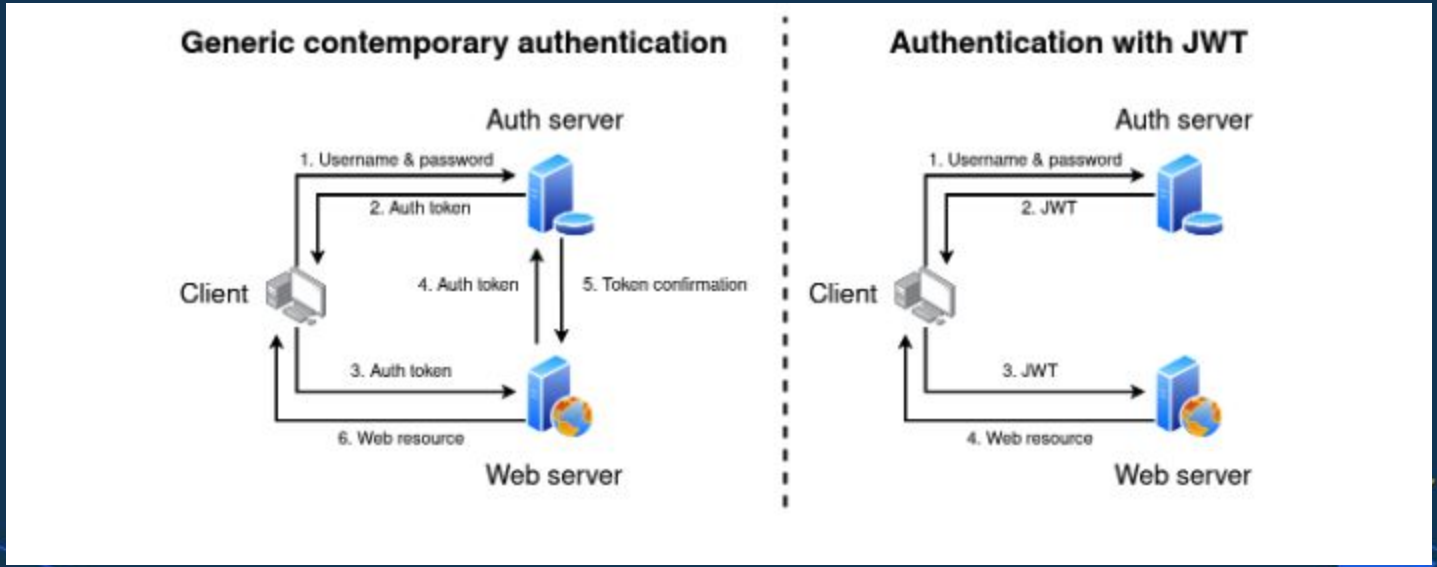
Structure of a JWT

- Combining the JSON objects previously shown creates our JWT, but before combining, we first need to base64 encode the information of the header and payload and concatenate them with full stops together with the secret key. The signature will be made by the HMACSHA256() function.

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

header = 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9'
payload = 'eyJpZCI6MTIzNCwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9'
msg = header + '.' + payload
sig = HS256('secret-key', msg).digestBase64()
```

How JWT performs over basic authentication mechanism



Source: Radix

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar