# How To Send JWT Token as Header

In the previous articles on [Postman Tutorial](https://www.softwaretestingmaterial.com/postman-tutorial/), we have covered “[Testing OAuth2 authorization in Postman](https://www.softwaretestingmaterial.com/testing-oauth2-authorization-in-postman/)“

In this “How To Send JWT Token As Header” article, I will be demonstrating as to how you can implement this concept and get a tight grip over this.

JSON Web Tokens are basically an industry standard and this basically describes how you can send data between parties.

**Example:**

{

first\_name: ‘John’,

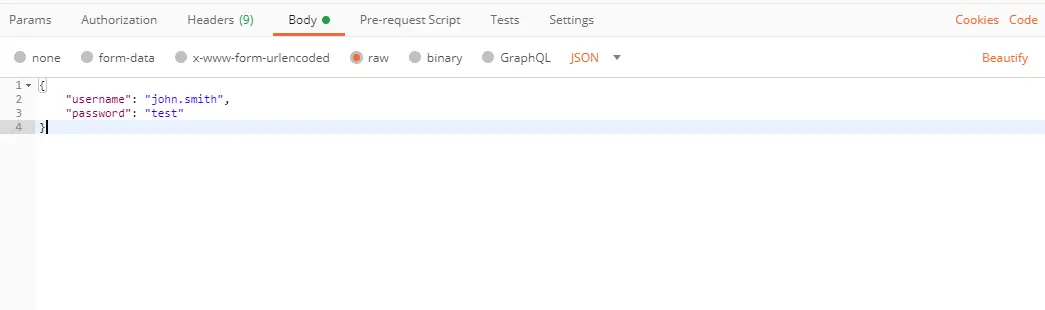
last\_name: ‘Smith’,

Email: ‘john.smith@example.com’,

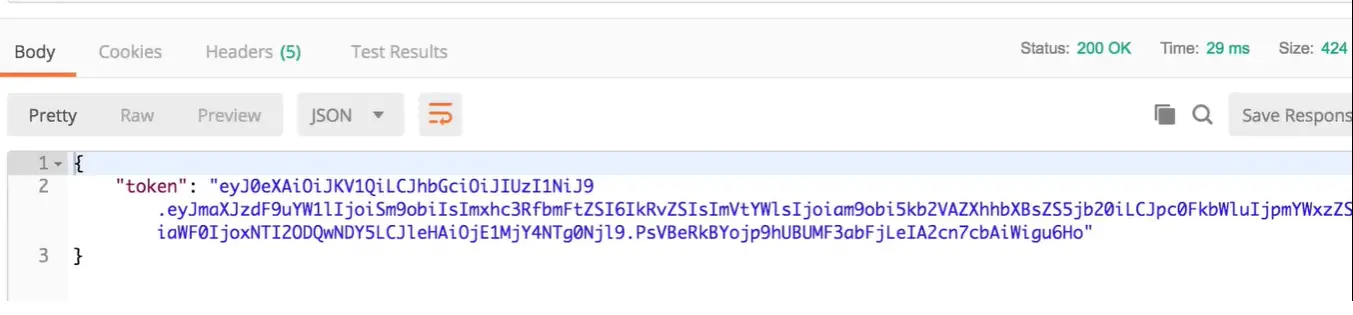
isAdmin: false

}

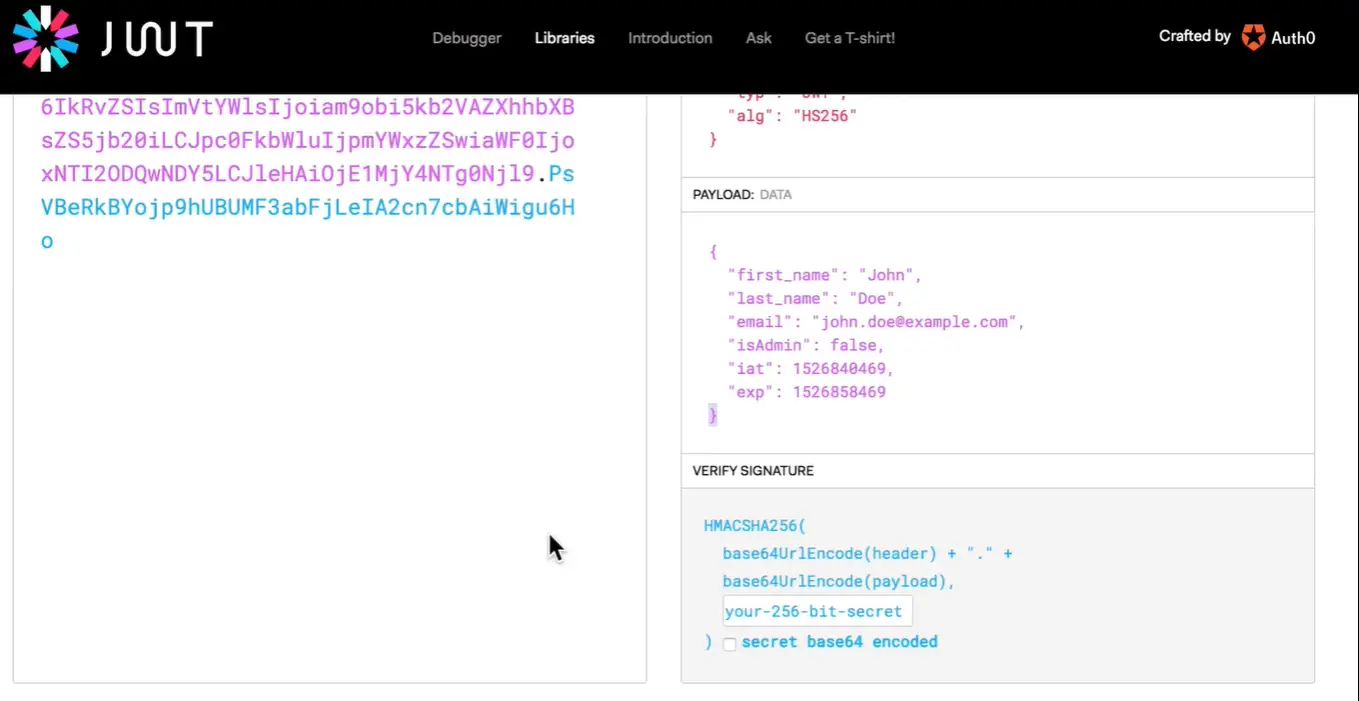
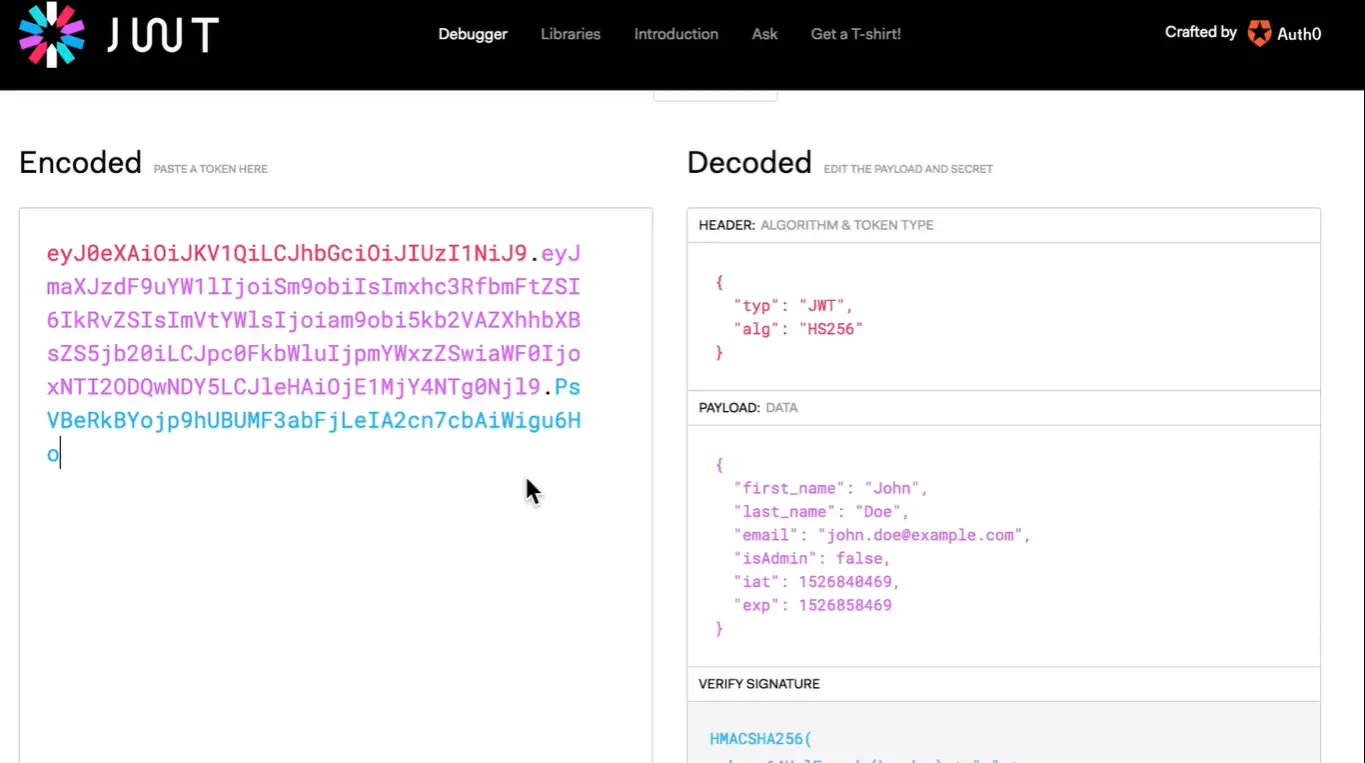
We need to allow third-party to verify that this information is authentic so that it doesn’t get changed in transit. So for example, once a token is generated with this information it should be signed so that we don’t have the possibility for example changing from isAdmin false to true without the other party being able to detect it.



When we submit this request, we get a JSON token as a response.



Go to jwt.io and in the editor paste the token value.



JSON web token is divided into three parts. Each of these parts is delimited by a dot symbol.

1. The first part is talking about what kind of a token is this and what encryption algorithm was used so that the party receiving that will be able to know that this is a jwt token and this specific encryption algorithm has been used.
2. The second part is the payload. So this part says that the person who has this token is John smith and his email address in is not an admin and additionally it has an expired date so this token is not valid for forever but it’s only valid for a limited period of time.
3. The third part is the important part of the token. It is the signature itself and the server that generated this signature and this token basically took the header, took the payload and with a very long secret generated a signature.

Any changes to this payload will be visible because the signature of this entire message will not be the same and anybody having this secret key will be able to tell that the message the payload has been manipulated.

Additionally, this string is base64 encoded and this is just a way of encoding this data but the data is signed and not encrypted.

We can send this token to other endpoints. This can be done easily. We have to add an authorization header in our request and this will be a Bearer TOKEN.



To avoid any manual copy-pasting of JWT token, we can use variables to add a script in the Tests tab of API request which is generating token.

const response = pm.response.json();

pm.globals.set(“jwt\_token”, response.token);

In the request that needs this token, edit value of Authorization header as {{jwt\_token}}