edWisor

Chat application - User management

- Aditya Kumar
Chief Technology Officer, edwisor.com

We will cover with the following concepts

1) Signup Page

2) Concept of JWT

3) Login Page

Authentication Vs Authorization

- Authentication is the process in which the system verifies who is using it

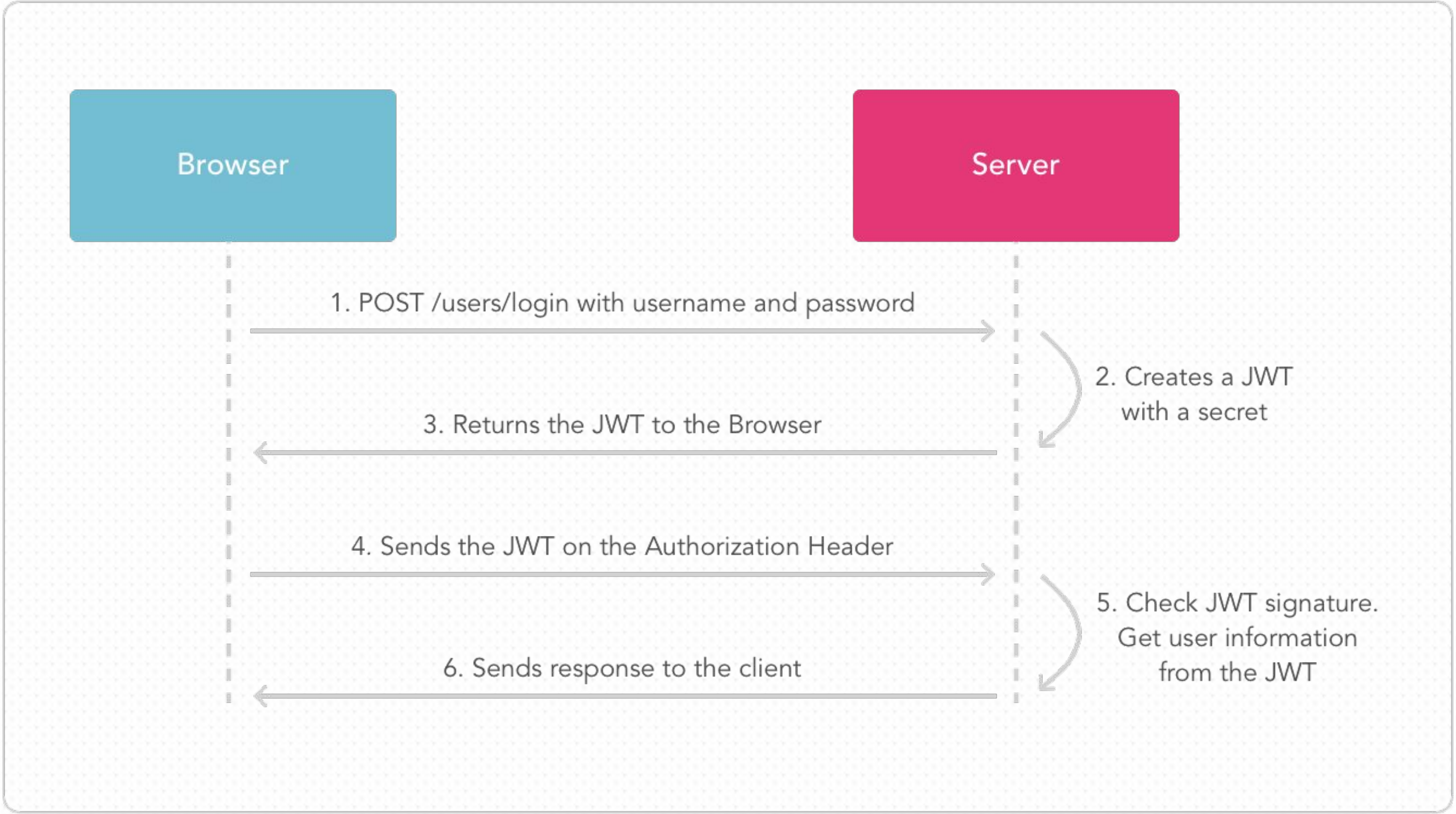
- In most systems, this is done using a loginId(such as username or email etc) and password combination.

- If the combination is correct, user is issued an access token which then is used for authorization

- Authorization is the process in which the system verifies whether the person logged in has the permission to perform an action.

- This is done by attaching the authorization token along with the request of the action. If the user is allowed to do that action, he is moved forward or else gets an error message.

JWT is the most prominent methodology of authorization followed in Web



Browser
Server
1. POST /users/login with username and password
2. Creates a JWT with a secret
3. Returns the JWT to the Browser
4. Sends the JWT on the Authorization Header
5. Check JWT signature. Get user information from the JWT
6. Sends response to the client

Let write code for generating and verifying claims

- We will use the official library for Node.JS - jsonwebtoken
- Separate the code into a library and write functions for generating and verifying claim.
- Write the code for Signup and Login API

Too many callbacks can lead to Hell

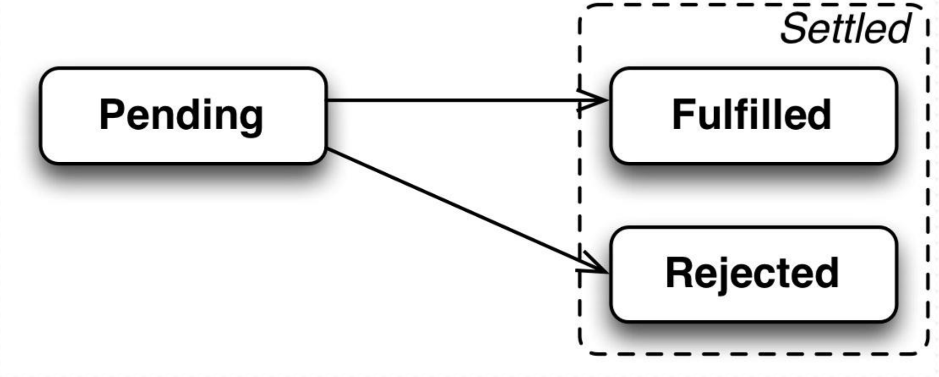
- We love callbacks, they are the fuel of asynchronous programming, but they become a problem when they are too many of them.

- This problem is famously called the **Callback Hell.** Check out callbackhell.com

- There are many solutions suggested for solving this problem like using Async library or promises.

- Async was quite popular, but Promises are the more recommended approach. So we will use promises when dealing with multiple action in same API.

Promises will be fulfilled or rejected



- **Pending state -** Initial state of the promise.
- **Fulfilled** or resolved state - Promise was successful.
- **Rejected state -** Promise operation failed.

- Promises look more like synchronous code but are purely asynchronous in nature.
- They can be used to write better, clean and understandable code.
- They are a part of core Javascript

Promises come with great Advantage

- They can be chained easily using **then.**
- Controlled flow is more “controlled”
- Error handling is built in. You can catch the error.
- Lets try and write out code using promises now.

Writing your own Auth model?

- Gives us an extra layer of security and verification.

- Helps us to invalidate the tokens to prevent unauthorized use of tokens.

- Can encapsulate additional information about the user

The next steps are ...

Encrypting your password

Writing authorization middleware

Logout