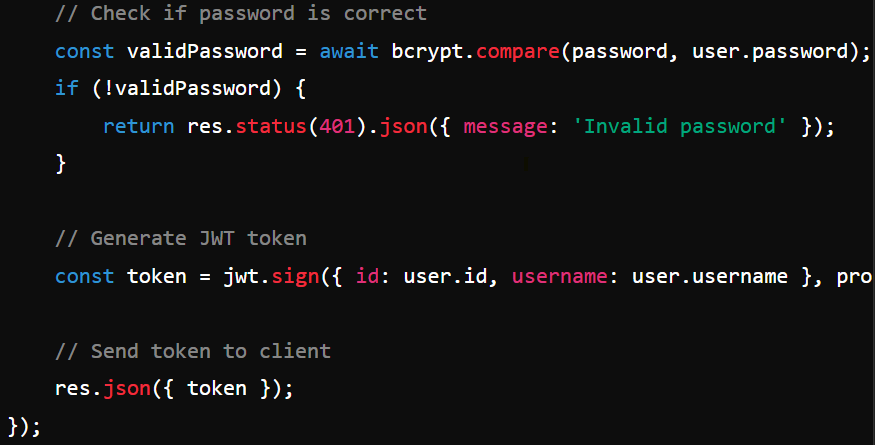
Certainly! Here’s a simple breakdown of how **authentication** and **authorization** can be implemented in a mobile app using **Express.js** (backend) and **Flutter** (frontend):

**Authentication (Login)**

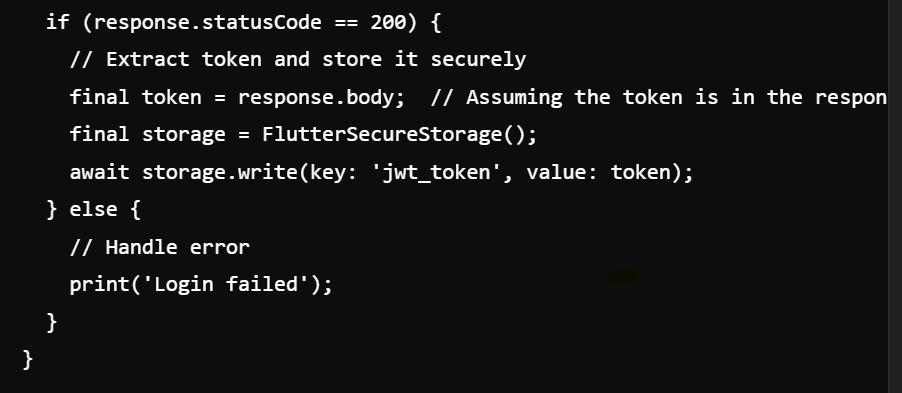
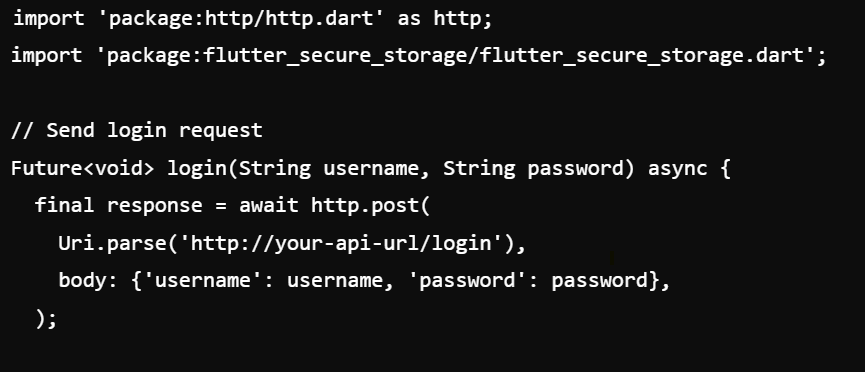
**Authentication** ensures that the **user is who they say they are**.

**Backend (Express.js)**

* **User sends login credentials (username, password)**.
* **Backend checks credentials** against the database.
* If valid, a **JWT token** is generated and returned to the user.
* The token contains encoded information (e.g., user ID) and is **signed** to prevent tampering.
* **Code Example (Express.js)**

**Frontend (Flutter)**

* **User enters credentials** and sends them to the backend.
* If the credentials are valid, the backend returns a **JWT token**.
* The Flutter app stores the token securely (e.g., in **secure storage** or **shared preferences**).
* The token is sent along with every future request to **prove the user's identity**.

**Code Example (Flutter)**

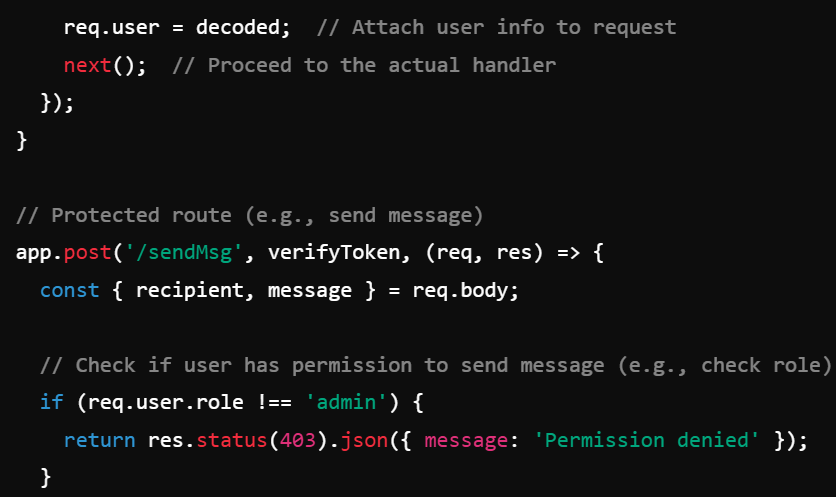
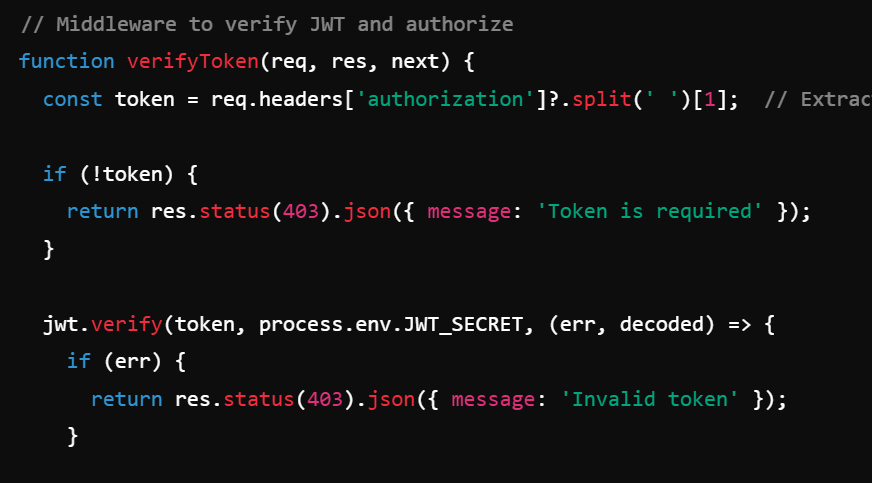
**Authorization (Access Control)**

**Authorization** ensures that the **authenticated user has permission** to access specific resources or perform certain actions.

**Backend (Express.js)**

* Every time the user tries to access a protected route; they must provide the **JWT token** in the **Authorization header**.
* The backend verifies the token using **JWT** and checks the **user’s role or permissions**.
* If valid, the request is allowed; otherwise, it’s rejected (e.g., **401 Unauthorized** or **403 Forbidden**).

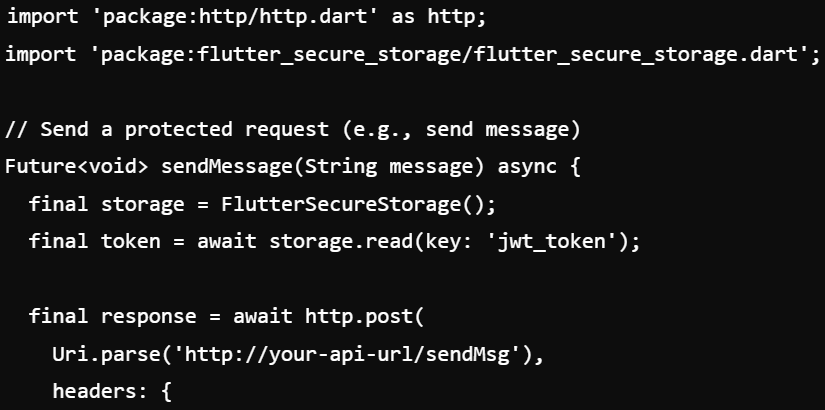
**Code Example (Express.js)**

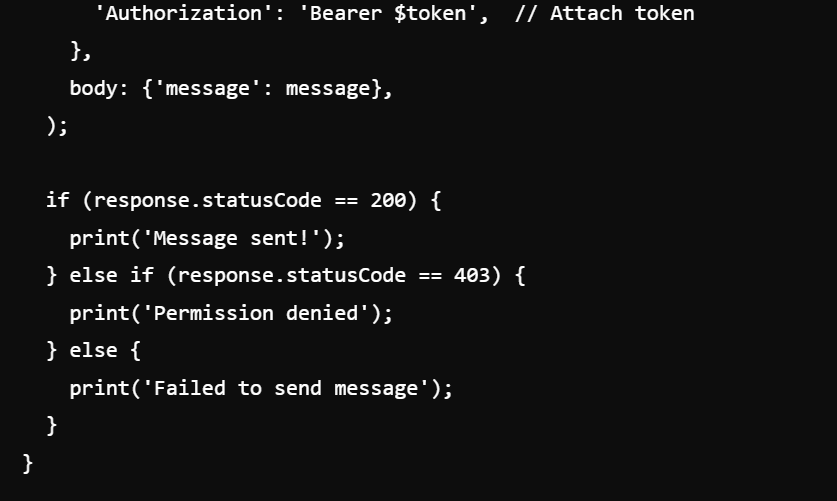


**Frontend (Flutter)**

* When sending a request to a protected endpoint (e.g., send message), the app retrieves the **JWT token** from secure storage and attaches it in the **Authorization header**.
* The backend will then validate the token and authorize the user based on their role or permissions.

**Code Example (Flutter)**





**Summary of the Flow**

1. **Authentication (Login)**:
   * User logs in by sending their credentials (username, password) to the backend.
   * Backend verifies credentials and generates a **JWT token**.
   * The token is sent to the frontend (Flutter app), which stores it securely.
2. **Authorization (Access Control)**:
   * Whenever the user accesses a protected route (like sending a message), the **JWT token** is sent with the request.
   * Backend verifies the token, checks the user’s role/permissions, and either allows or denies access to the resource.

**Key Points**

* **Authentication** proves the **user’s identity** (login and token generation).
* **Authorization** checks if the **authenticated user has permission** to access certain resources (role-based access).

Top of Form

Bottom of Form