**1. Cookies & JWT Tokens**

So, whenever a user logs into the app, we need some way to remember who they are for the next requests. Instead of making them log in every time, we use something called a JWT (JSON Web Token). Once the user is authenticated, we create this token and store it in the browser — not just anywhere though — we use a secure, HTTP-only cookie.

Why? Because it’s safer. This kind of cookie can’t be accessed by JavaScript running on the frontend, which helps protect against some attacks like XSS (cross-site scripting). Once it’s there, it automatically gets sent with every request, so the backend knows, “yep, this user is logged in.” Clean, safe, and no extra code needed on the frontend to keep track of it.

**2. OTP Verification (One-Time Password)**

When someone signs up or logs in but hasn’t been verified yet, we don’t just trust them immediately. Instead, we generate a random 6-digit OTP and store it in the database along with an expiry time — think of it like a mini-password that only lasts a few minutes.

This OTP is linked with either their email or mobile number (whatever they used). For testing, we just log it to the console, but ideally, we’d send it via email or SMS.

Then, when the user types in the OTP they got, we:

* Check if it’s correct for that email or number
* Make sure it hasn’t expired
* And if all is good, we update their status in the database to say they’re verified
* Finally, we delete the OTP so it can’t be used again

This way, we make sure only real users with access to the actual email/phone get through.

**3. Middleware**

Think of middleware like a security guard at the door. Whenever someone tries to visit a protected page (like their dashboard or profile), our requireAuth middleware steps in. It checks if the incoming request has a valid JWT token in the cookie.

If there’s no token or it’s fake/expired, the request is denied — like, “sorry buddy, you can’t come in.” But if it’s valid, the middleware lets the request pass and even adds the user’s ID to the request so the next function knows who’s logged in.

It’s a super simple way to protect parts of your app that should only be accessed by logged-in users.