SECURITY

you know / you are / you have

AUTHENTICATION (AUTH) – are you who you say you are?

AUTHORIZATION – now that we know who you are, what are you allowed to do?
- role-based

RESTful Apps = Stateless (forget after each request)
↳ we need ways to maintain auth between req's

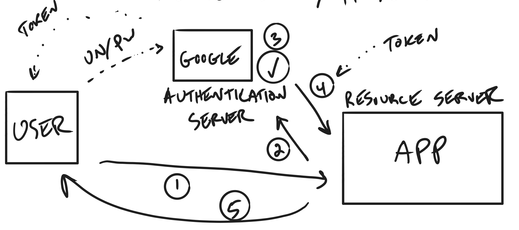
3 MAIN FLOWS

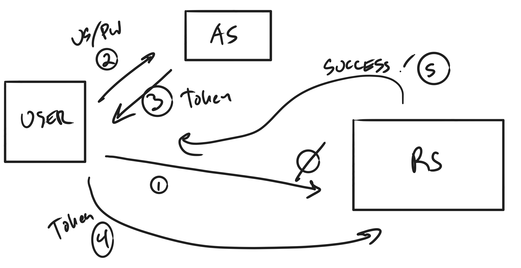
Basic Auth – each req contains username/password
JWT-Based – JSON Web Token
OAuth – (also uses these)

BA → must send credentials each time
insecure due to repeated exposure
easily decrypted
BUT → simple, easy to set up

JWT → user logs in w/ username/password
receive a token containing username but NOT pw
subsequent requests ONLY include the token
- encrypted server-side with a "secret"
- only the encryptor can decrypt
- reduced risk of credential exposure
- ease of resecuring
- must manage the token on the FE
  ↳ in general, this much more complex
- APP STILL HAS YOUR LOGIN INFORMATION!!

OAuth → a third-party company provides authentication of username/password
- and provides a token you then use to get authorized on the new server
- the new server/app NEVER sees PW





CORS → Cross-Origin Resource Sharing

CSRF → Cross-Site Resource Forgery

Passport JS → JS library for various security "strategies"