Agenda — 4 class topic — Authentication & Authorisation

1. Authentication
2. Authorisation
3. Auth vs Auth
4. RBAC (Role Based Access control)
5. How Auth flow works
   - → Token

Next class
1. JWT
2. OAuth
3. Implementing Auth Service

Authentication

KYC
- → Know the identity of the person

scaler.com | academy
- → Anyone on the internet can visit.

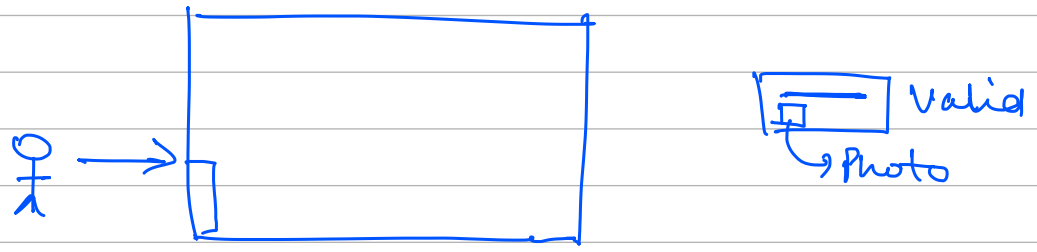
scaler.com | meetings | ——
- → login
- → to access this page we need some kind of permission

scaler.com/instructor/meeting/—

↓

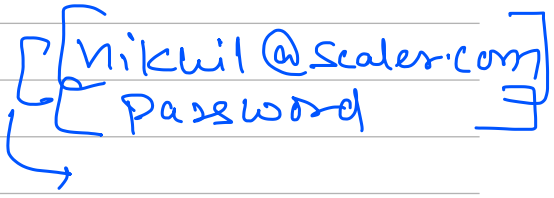
Only user with inst premission can join the above URL

eg



Authentication

↳ Proving to the server that you are someone



scaler.com/inst

DEAN

Authorization

Authentication + Permission to perform an operation

Atlas vs Jira

RBAC Role Based Access Control

- Student —
- Prof — Teaching
- HOD — Department
- DEAN — College

FEST Needs DEAN

New Curriculum Dep HOD

Confluence
- Create Page
  - Move Page to different SPACE
    - ADMIN
- BACKUP Conf
  - SUPER ADMIN

User

Roles
Student
Inst
Mentor
TA

User 1 | M Roles

User M | 1 Roles

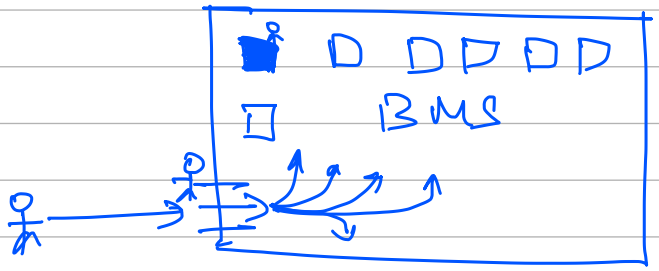
M : M

Mapping table

eg Movie Booking System

	Authentication	Authorization
① Search any Movie	X	X
② Book a ticket	✓	X
③ Cancel a ticket	✓	✓

bms.com/ticket/id=1



How Authentication works

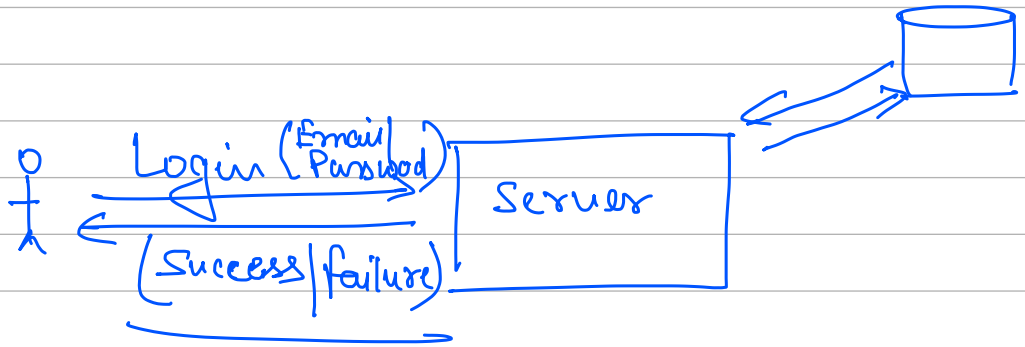
- Email | Password
- Phone | OTP

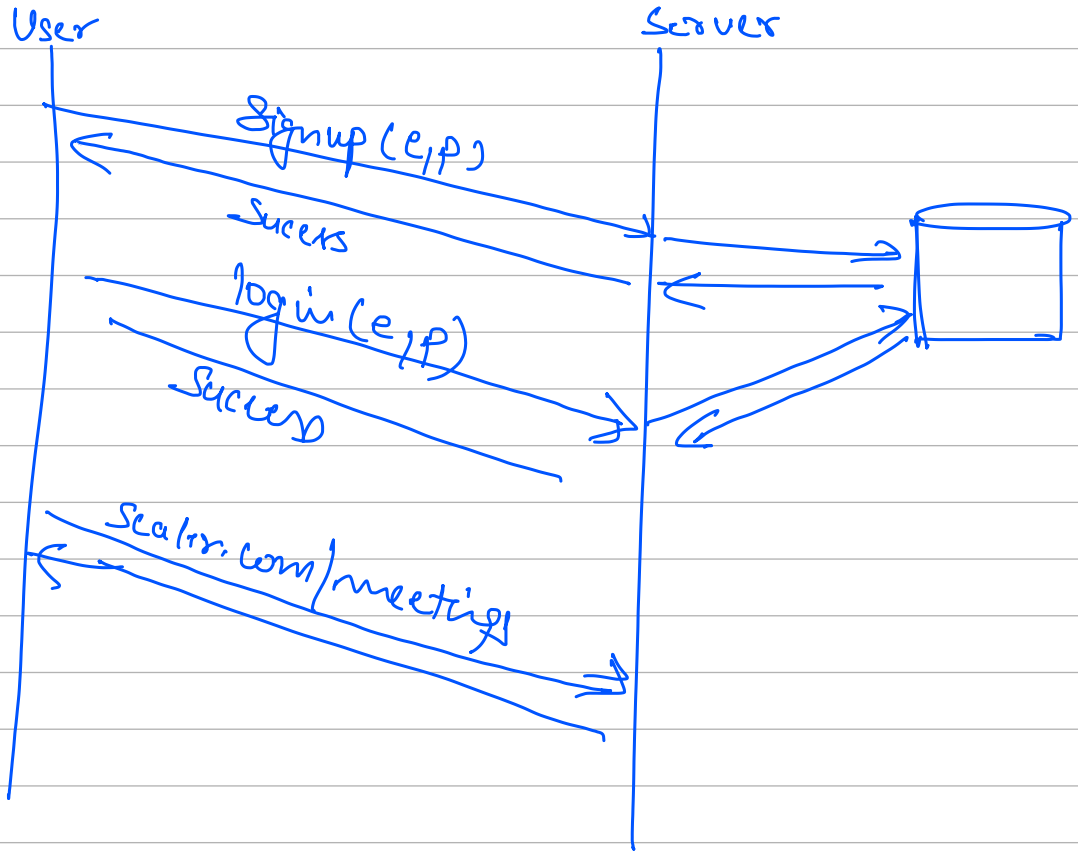
1. Signup or Register

- Registering verification details
- Create user in DB with verification info

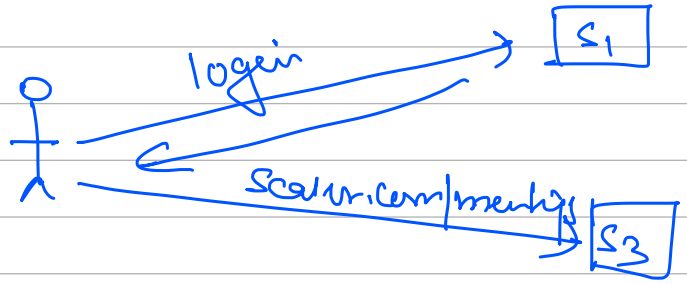
User	
Email	Password

② Login





- Every request should self sufficient
- Stateless



login creds with every request

{ Server needs to verify creds every time with every call

User Table

id	email	Password	

① What if DB get compromised ?

Sol'n We should Not Save the password in raw text format

① only Scaler
② only Scaler/Email
③ Scaler/Email/Bank.

Encryption | Hashing

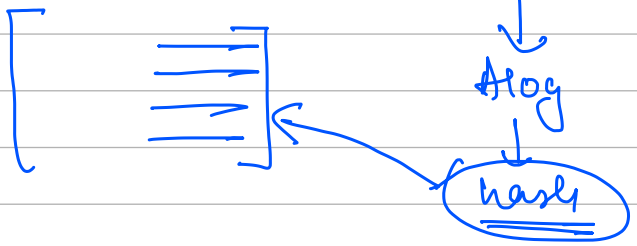
Hash(Password)

User	Password

Strongest Encryption

Top 20 Passwords

Password 123 → 1.2 Million



Hashing + Salt →

BCrypt Encoder → Encode ( )
→ Matches ( )

↓

Boolean

Salt → Timestamp
MacAddress.
Constant

→ Auto Generated Number

encode(Password + Salt)

Signup

↳ (Email, Password)

↓

Email, BCrypt.Encode(Password) → Hash

(Email, hashedPassword) → DB

Login (Email, Password)

↓

Email → fetch hashedPassword DB

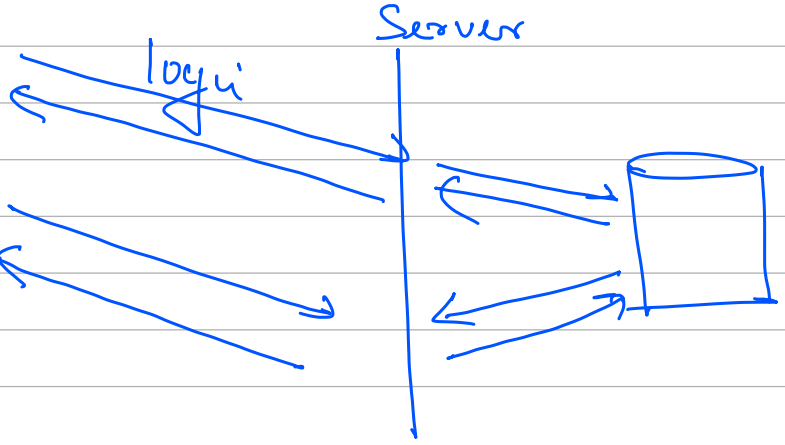
if(BCrypt.matches(HashPwdDB, Password)

loginSuccess → How it works internally

else loginFailed

Token

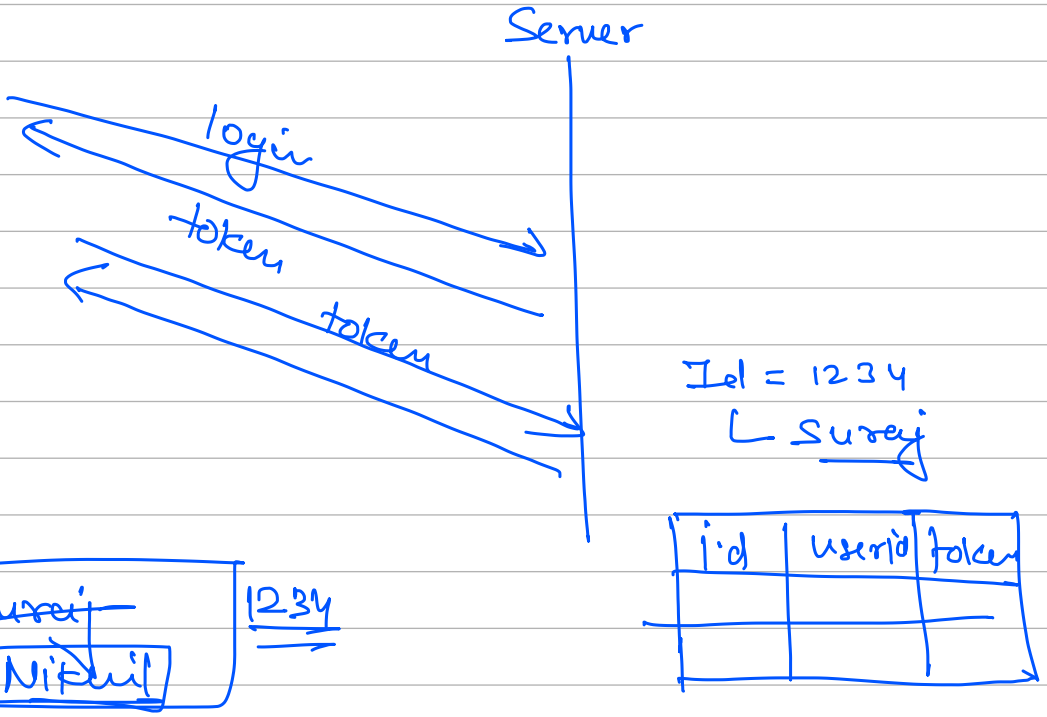
Server HTTP are stateless



Cache

→ Costly

→ Sync Overhead



To validate the token, server needs to call DB

What if token itself contains the info. required to validate the token, then do we need the DB call? NO

Self Validating Tokens

- JWT → JSON Web Token