Authenticate

Why?

After users go to website and generate data, to associate each user with data, we need to create an account for them so that all of the data that is generated can be accessed when they come back after some time.

Restrict access: This allows people to access some data is available to some specific set of people such as Netflix, Spotify premium memberships.

Authentication has a lot of parts and there are a lot of levels to specify the security of website.

Level 1:

Just creating accounts and storing in database and then checking later on.

Level 2:

Encrypting database.

Encryption: changing message into some form that can decoded.

Caesar Cipher is very common. But it is crackable nowadays.

Mongoose-encryption

Use environment variables to store secrets and API keys.

Level 3:

Hashing

Usually when we use ciphers, we need keys for encryption and decryption. But hashing removes the need to have a key. Using a hashing function converts our data into encrypted data directly which is impossible to decrypt (using current computing standards). A hashing function is a mathematical function.

Data is converted into hash and that hash is stored into the database. On login, the login password is again converted into hash and those two hashes are compared.

Making a hash table:

All words from a dictionary (150000)

All numbers from a telephone book (5,000,000)

All combinations of characters up to 5 places (19,770,609,664)

About 19 billion

Level 4: Salting and Hashing

To prevent dictionary attacks

In addition to the password, we add some random letters and add it to password, then the hashing is done which is more secure.

Salt rounds: after the hash is generated after salting and hashing, another salt is added to the hash generated after first round and then again hashing is done. This is 1 salt round.

We have username stored, salt stored, hash stored.

Level 5:

Cookies & sessions

Cookies are made for saving information about how someone interacted on the website and using that interaction data for other uses such as targeted ads etc. Cookies save a variety of information which is dependent on what developer wants to save using cookies such as items in a cart, or the state whether the user was logged in or not.

A session is the time period during which the user interacted with the website.

Passport.js is a very common library that is used with Node.js

For passport.js install these packages:

passport passport-local passport-local-mongoose express-session

Passport local package automatically does salt and hash.

Serialize allows us to make the cookie and store the information there and deserialiase allows us to read the data from our cookie.

Level 6:

OAuth Open authorization

Allows to login to application using other social apps or services like Google and Facebook

Use case:

A request will be made to Facebook for example to GET all friend list and then they will be returned.

Why OAuth?  
1. Granular Level Access: specific things can be accessed such as email, name and other data.

2. Read/Read-Write Access: We can either just take information from Facebook or we can also post things to their facebook account.

3. Revoke Access: User should be able to remove access from account anytime from ex. Facebook.

Step 1: set up your app on the OAuth platform.

Step 2: Redirect to authenticate.

Step 3: User logs in on OAuth Platform.

Step 4: User grants permission.

Step 5: Receive authorisaton code that yes user has logged in.

Step 6: Exchange AuthCode for Access Token, this token can be then used for accessing user data (email) and has a longer validity than authorization code.

Using passport google oauth 2.0 which is latest.