**NodeJs Concepts:**

1. [**https://nodejs.org/en/learn/getting-started/introduction-to-nodejs**](https://nodejs.org/en/learn/getting-started/introduction-to-nodejs)
2. [**https://www.geeksforgeeks.org/nodejs/**](https://www.geeksforgeeks.org/nodejs/)
3. [**https://www.geeksforgeeks.org/node-js-eventemitter/**](https://www.geeksforgeeks.org/node-js-eventemitter/)
4. [**https://dev.to/meddy672/node-js-the-eventemitter-102o**](https://dev.to/meddy672/node-js-the-eventemitter-102o)
5. [**https://www.freecodecamp.org/news/node-js-streams-everything-you-need-to-know-c9141306be93/**](https://www.freecodecamp.org/news/node-js-streams-everything-you-need-to-know-c9141306be93/)
6. **Cron in node js**

cron is a robust tool for running jobs (functions or commands) on schedules defined using the cron.schedule( ‘\* \* \* \* \* ’, () => {}) syntax. Perfect for tasks like data backups, notifications, and many more!

npm install node-cron

[**https://javascript.plainenglish.io/how-to-schedule-cron-jobs-in-node-js-54fbec910664**](https://javascript.plainenglish.io/how-to-schedule-cron-jobs-in-node-js-54fbec910664)

1. **Webpack**A webpack is a scalable , configurable static module bundler which bundles the files (js, css, img) and generate a single static modules (one js, one css, one img, one video). By default, from entry entry point it starts creating tree of ddpendencies and root module is bundled into one.

[**https://www.smashingmagazine.com/2021/06/getting-started-webpack/**](https://www.smashingmagazine.com/2021/06/getting-started-webpack/)

1. **Validation joi or express-validator -> used for Object Schema validation**Joi is an object schema description language and validator for JavaScript objects which helps us to accept objects of specific object

 The most well-known, effective, and often utilised package for object schema definitions and validation is called Joi. Joi gives programmers the ability to create Javascript blueprints and ensure that the application will accept properly structured input.

Npm install joi

<https://medium.com/@Arnav-mah/joi-validation-with-node-js-and-postman-598fb0af8c61>

<https://medium.com/@Arnav-mah/joi-validation-with-node-js-and-postman-2-8e06227d76b6>

**validation zod in nest:**

[**https://medium.com/@maciejkoch90/angular-data-service-with-schema-parsing-using-zod-js-f320b0e8d2c8**](https://medium.com/@maciejkoch90/angular-data-service-with-schema-parsing-using-zod-js-f320b0e8d2c8)

**Class Validator**[**https://www.npmjs.com/package/class-validator**](https://www.npmjs.com/package/class-validator)

1. **Protecting API’S**

<https://medium.com/@developerom/protect-your-rest-apis-in-node-js-253c11a2dfd7>

HTTPS Protection:

<https://adamtheautomator.com/https-nodejs/>

**express-rate limit:**  
<https://www.npmjs.com/package/express-rate-limit>

**nests/throtller:**

<https://docs.nestjs.com/security/rate-limiting>

1. Authentication and authorization check below
2. Worked thread and clustering see below
3. Node js word doc by Sumit in pc
4. State management
5. View engines Handlebar <https://stackabuse.com/guide-to-handlebars-templating-engine-for-node/>
6. Health checkup using http-terminator and express-acurator

<https://expressjs.com/en/advanced/best-practice-security.html>

1. Securing through Helmet and CORS  
   <https://blog.logrocket.com/using-helmet-node-js-secure-application/>

<https://expressjs.com/en/resources/middleware/cors.html>

1. Db connection   
   Tedious and mssql  
   <https://www.npmjs.com/package/mssql>

**ORM:**

Any one ORM like mongoose, Sequelize, **TypeORM**, prisma  
  
  
**Docker -**

Docker compose

how to secure all the password in docker compose

docker network

docker service  
docker logs

**AWS Cloud:**  
how to deploy static site on GCP function, AWS lambda or Azure functions

**MicroServices:**  
check any messaging system like RabbitMQ or apache kafka (RabbitMQ is easy)  
<https://scanairobi.hashnode.dev/microservices-101-with-nodejs-and-rabbitmq>

**Single thread , Worker thread and cluster in node**

<https://alvinlal.netlify.app/blog/single-thread-vs-child-process-vs-worker-threads-vs-cluster-in-nodejs>

**Difference b/w thread and cluster:**Note clusters are build on top of child\_process which uses spawn(), fork() and exec()

Basically, you can think of the cluster module as allowing us to run multiple instances of Node in separate processes, while the worker thread module allows us to run multiple instances of Node in the same process.

<https://www.linkedin.com/pulse/ultimate-guide-nodejs-performance-feat-clustering-pm2-sutradhar#:~:text=Basically%2C%20you%20can%20think%20of,Node%20in%20the%20same%20process>.

**Thread:**

[**https://www.geeksforgeeks.org/node-js-worker-threads/**](https://www.geeksforgeeks.org/node-js-worker-threads/)

[**https://medium.com/@Trott/using-worker-threads-in-node-js-80494136dbb6**](https://medium.com/@Trott/using-worker-threads-in-node-js-80494136dbb6)

[**https://snyk.io/blog/node-js-multithreading-with-worker-threads/**](https://snyk.io/blog/node-js-multithreading-with-worker-threads/)

**Clustering:**

<https://dev.to/tawseef/clustering-in-nodejs-4pgn>

<https://blog.logrocket.com/optimizing-node-js-app-performance-clustering/>

<https://medium.com/@mjdrehman/implementing-node-js-cluster-for-improved-performance-f800146e58e1>

npm packages to build clustering:

<https://www.npmjs.com/package/pm2>

<https://pm2.keymetrics.io/docs/usage/cluster-mode/>

https://medium.com/yavar/pm2-cluster-mode-c7ab29c4b10d

with Nest:

https://levelup.gitconnected.com/nodejs-clustering-maximizing-nestjs-application-performance-16af18926abe

<https://www.optimum-web.com/pm2-cluster-mode-with-node-js/>

**Authentication and authorization :**

Authentication is the process of check who the user is. (sign in /login, sign up)

Authorization is the process of check what the user can do.( for each request sent we verify the token)

To verify the jwt token we use jwt.Verify(req.token, api\_token)

**Passport js**[**https://www.npmjs.com/package/passport**](https://www.npmjs.com/package/passport)

<https://medium.com/@vikramgyawali57/7-best-approach-of-protecting-routes-in-node-js-every-senior-engineer-suggests-fc32b7777827>

**Authencation logic:  
  
during signup:**

1. Check if user entered all mail and name or else throw 400 error as all fields required
2. If user data is correct then  
   a) check if user exists in db redirect to login page or

b) store it in db with password encypt by using bcrypt.hash(password, bcrypt.salt(10))

3) After storing generate jwt token using Jwt.sign( userObject, token, { expiresIn : ‘1h’})

1. Send response with jwt tokens assigned to response.

**During login:**

1. Check if user enterd all mail name and password or else thow 400 error as all fields required
2. If user data is their then compare password provided by user and in db by using bcrypt.compare(req.password, user.password)
3. If all good create jwt tokem using Jwt.sign( userObject, token, { expiresIn : ‘1h’})
4. Send response with jwt token assigned to response.

**Authorization:**

**In express use middleware to check authorization, role acces and permission access**

**In nest use guards such as AuthGuard, RolesGuard, and PermissionGuard  
to sepecify public route use decorators createete Public() decorator and for roles Role([role.admin]) decorator**

For each subsequent request check the following:

(checking of token needs to be done in verifyToken middleware)

1. Check if the request sent has token in body or params or query or header (x-access-token) if token not found throw error
2. If token fond verify the token with existing and new using jwt.verify( req.token , aptoken)
3. If token is true then proceed or else send 401 unauthorized token exception.

**In Node and Express :**

We used npm packages bcrypt and jwt token  
<https://medium.com/@xiaominghu19922/authentication-and-authorization-with-nodejs-and-react-part-1-8dd0ea7f79ca>

<https://medium.com/@xiaominghu19922/authentication-and-authorization-with-nodejs-react-and-typescript-part-2-ae9d320e4f74>

<https://www.toptal.com/json/jwt-nodejs-security>  
  
<https://dvmhn07.medium.com/jwt-authentication-in-node-js-a-practical-guide-c8ab1b432a49>

**In Node and Nest js :**

We use npm package : Passport.local for user and password check and passpot.jwt to use with jwt

1. Create authgaurd that implements passport authguard
2. Use it over controller by using @UseGuards(authquard)
3. Implemet stratergy that extends passport stratergy.

https://medium.com/att-israel/authentication-authorization-using-react-nestjs-jwt-token-55f52070a3f2

**with asp.net core:**

need packages   
dotnet add package Microsoft.AspNetCore.Authentication.JwtBearer  
dotnet add package System.IdentityModel.Tokens.Jwt  
<https://www.c-sharpcorner.com/article/authentication-and-authorization-in-asp-net-core-web-api-with-json-web-tokens/>

<https://medium.com/@susithapb/implementing-jwt-authentication-and-authorization-in-net-core-web-api-bf438d7ab9a7>

**Single Sign On with outh2.o**

[https://medium.com/javarevisited/single-sign-on-sso-saml-oauth2-oidc-simplified-cf54b749ef39#:~:text=The%20OAuth2%20protocol%20allows%20the,provided%20by%20the%20resource%20server.](https://medium.com/javarevisited/single-sign-on-sso-saml-oauth2-oidc-simplified-cf54b749ef39%23:~:text=The%20OAuth2%20protocol%20allows%20the,provided%20by%20the%20resource%20server.)

**State Management**

1. **ASP .NET CORE**

[**https://code-maze.com/state-management-in-asp-net-core-mvc/#:~:text=ASP.NET%20Core%20maintains%20the,to%20fetch%20the%20session%20data**](https://code-maze.com/state-management-in-asp-net-core-mvc/#:~:text=ASP.NET%20Core%20maintains%20the,to%20fetch%20the%20session%20data)**.**

[**https://www.infoworld.com/article/3680070/6-state-management-techniques-for-aspnet-core-mvc.html**](https://www.infoworld.com/article/3680070/6-state-management-techniques-for-aspnet-core-mvc.html)

[**https://medium.com/simform-engineering/caching-in-net-core-7c759a5bc3c6**](https://medium.com/simform-engineering/caching-in-net-core-7c759a5bc3c6)

1. **JavaScript**

[**https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP\_StateManagement.html**](https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_StateManagement.html)

[**https://javascript.info/cookie**](https://javascript.info/cookie)

[**https://javascript.info/localstorage**](https://javascript.info/localstorage)

[**https://medium.com/@codebyamir/a-web-developers-guide-to-browser-caching-cc41f3b73e7c**](https://medium.com/@codebyamir/a-web-developers-guide-to-browser-caching-cc41f3b73e7c)

1. **Angular**

<https://dev.to/chintanonweb/angular-state-management-a-comparison-of-the-different-options-available-100e>

<https://levioconsulting.com/insights/intro-to-rxjs-in-angular-observables-subjects-and-behaviorsubjects/#:~:text=BehaviorSubjects%20are%20a%20type%20of,last%20emitted%20value%20upon%20subscription>

<https://designtechworld.medium.com/managing-state-in-angular-with-ngrx-store-19c4d99e83bc>

<https://www.apriorit.com/dev-blog/angular-state-management-ngrx>

<https://stackoverflow.com/questions/52471796/what-is-state-management-in-angular-and-why-should-i-use-it>

1. **Node JS**

[**https://dev.to/saint\_vandora/how-to-implement-session-management-in-nodejs-applications-5emm**](https://dev.to/saint_vandora/how-to-implement-session-management-in-nodejs-applications-5emm)

[**https://www.section.io/engineering-education/session-management-in-nodejs-using-expressjs-and-express-session/**](https://www.section.io/engineering-education/session-management-in-nodejs-using-expressjs-and-express-session/)

For Production use store as mssql-session-store

[**https://www.npmjs.com/package/mssql-session-store**](https://www.npmjs.com/package/mssql-session-store)

<https://pankaj-kumar.medium.com/how-to-handle-cookies-in-node-js-express-app-b16a5456fbe0>

<https://stackoverflow.com/questions/16209145/how-can-i-set-cookie-in-node-js-using-express-framework>

<https://www.linkedin.com/pulse/caching-nodejs-oleabhiele-daniel-donald>

<https://medium.com/@muhammadtaifkhan/cache-your-express-nodejs-api-1e9f80b87c88>

<https://www.npmjs.com/package/apicache>

<https://levelup.gitconnected.com/caching-strategies-in-node-js-improving-performance-and-efficiency-6a7fd929e165>

<https://www.honeybadger.io/blog/nodejs-caching/>

<https://www.geeksforgeeks.org/how-to-access-cache-data-in-node-js/>

-------------Interview Questions -------------------------------------------------

**NodeJs Interview Questions:**

<https://www.toptal.com/nodejs/interview-questions>

<https://gist.github.com/paulfranco/9f88a2879b7b7d88de5d1921aef2093b>

https://www.geeksforgeeks.org/node-js-questions-complete-reference/?ref=lbp

**Nestjs:**

<https://www.linkedin.com/pulse/5-interview-questions-related-nestjs-backend-engineer-ali-hassan-llohf>

<https://climbtheladder.com/nestjs-interview-questions/>

<https://anywhere.epam.com/en/blog/advanced-node-js-interview-questions-answers>

<https://www.shanumittal.com/blogs/technology/nestjs-interview-questions-answers>

<https://welovedevs.com/app/tests/questions-techno-nest>

**Not required:**

<https://adevait.com/nodejs/interview-questions>

https://www.toptal.com/nodejs/interview-questions

<https://www.tutorialspoint.com/nodejs/nodejs_interview_questions.htm>

<https://blog.risingstack.com/node-js-interview-questions/>

<https://www.guru99.com/node-js-interview-questions.html>

<https://www.techbeamers.com/top-30-node-js-interview-questions-answers/>

<https://www.interviewbit.com/node-js-interview-questions/>

**Javascript Interview questions:**

<https://www.toptal.com/javascript/interview-questions>

[https://github.com/sudheerj/javascript-interview-questions#what-are-the-possible-ways-to-create-objects-in-javascript](https://github.com/sudheerj/javascript-interview-questions%23what-are-the-possible-ways-to-create-objects-in-javascript)

<https://medium.com/@SilentHackz/4-javascript-design-patterns-you-should-know-9b8e6c3abdaf>

<https://github.com/lydiahallie/javascript-questions>

**Angular:**

1. **Angular 17**

[**https://academind.com/tutorials/angular-17-the-renaissance-continues**](https://academind.com/tutorials/angular-17-the-renaissance-continues)

[**https://medium.com/angular-in-depth/implementing-custom-component-decorator-in-angular-4d037d5a3f0d**](https://medium.com/angular-in-depth/implementing-custom-component-decorator-in-angular-4d037d5a3f0d)

**Element Ref , Template ref and view container ref:**

[**https://plainenglish.io/blog/angular-dom-manipulation-elementref-templateref-and-viewcontainerref**](https://plainenglish.io/blog/angular-dom-manipulation-elementref-templateref-and-viewcontainerref)

[**https://stackoverflow.com/questions/53374430/difference-between-elementref-and-templateref-in-angular4**](https://stackoverflow.com/questions/53374430/difference-between-elementref-and-templateref-in-angular4)

[**https://angularindepth.com/posts/1052/exploring-angular-dom-manipulation-techniques-using-viewcontainerref**](https://angularindepth.com/posts/1052/exploring-angular-dom-manipulation-techniques-using-viewcontainerref)

[**https://website-development.ch/blog/angular-viewcontainerref-templateref-elementref**](https://website-development.ch/blog/angular-viewcontainerref-templateref-elementref)

[**https://medium.com/angular-in-depth/exploring-angular-dom-abstractions-80b3ebcfc02**](https://medium.com/angular-in-depth/exploring-angular-dom-abstractions-80b3ebcfc02)

[**https://www.youtube.com/watch?v=uQqp1z7FpJY**](https://www.youtube.com/watch?v=uQqp1z7FpJY)

1. **Interview questions**

[**https://github.com/sudheerj/angular-interview-questions**](https://github.com/sudheerj/angular-interview-questions)

[**https://www.turing.com/interview-questions/angular**](https://www.turing.com/interview-questions/angular)

[**https://flexiple.com/angular/interview-questions**](https://flexiple.com/angular/interview-questions)

[**https://www.toptal.com/full-stack/interview-questions**](https://www.toptal.com/full-stack/interview-questions)

**Typescript:**

[**https://www.onlineinterviewquestions.com/typescript-interview-questions/**](https://www.onlineinterviewquestions.com/typescript-interview-questions/)

**.net:**

[**https://www.talkingdotnet.com/asp-net-core-interview-questions/**](https://www.talkingdotnet.com/asp-net-core-interview-questions/)

[**https://www.dotnetcurry.com/dotnetinterview/70/dotnet-interview-questions-answers-beginners**](https://www.dotnetcurry.com/dotnetinterview/70/dotnet-interview-questions-answers-beginners)

[**https://stackoverflow.com/questions/365489/questions-every-good-net-developer-should-be-able-to-answer**](https://stackoverflow.com/questions/365489/questions-every-good-net-developer-should-be-able-to-answer)

[**https://www.hanselman.com/blog/what-great-net-developers-ought-to-know-more-net-interview-questions**](https://www.hanselman.com/blog/what-great-net-developers-ought-to-know-more-net-interview-questions)

[**https://www.careerride.com/ASPNet-Questions.aspx**](https://www.careerride.com/ASPNet-Questions.aspx)**s**

**C#:**

[**https://www.toptal.com/c-sharp/interview-questions**](https://www.toptal.com/c-sharp/interview-questions)

[**https://www.dotnetfunda.com/interviews/cat/6/csharp**](https://www.dotnetfunda.com/interviews/cat/6/csharp)

[**https://www.guru99.com/c-sharp-interview-questions.html**](https://www.guru99.com/c-sharp-interview-questions.html)

[**https://www.techbeamers.com/csharp-coding-interview-questions-developers/**](https://www.techbeamers.com/csharp-coding-interview-questions-developers/)

[**https://www.techbeamers.com/csharp-oops-interview-questions-answers/**](https://www.techbeamers.com/csharp-oops-interview-questions-answers/)

[**https://www.interviewbit.com/c-sharp-interview-questions/**](https://www.interviewbit.com/c-sharp-interview-questions/)

**Youtube:**

<https://www.youtube.com/@piyushgargdev>

all above most links from those git:

<https://github.com/DopplerHQ/awesome-interview-questions/blob/master/README.md>