Node.js API

This small Node + Express API is a learning playground. It contains several simple endpoints you can call from the browser, curl, or Postman. The project uses an in-memory array for users so you can focus on HTTP, routing, and JSON without needing a database.

What you'll learn

- How to run a Node.js/Express server
- How to call GET/POST/DELETE endpoints
- How to read and send JSON payloads
- A simple example of input validation and error handling

Prerequisites

- Node.js (recommend v20+)
- npm (comes with Node)

Quick start

1. Install dependencies

```
npm install
```

2. Start the server

```
npm start
# Or you can start directly using node:
node server.js
```

3. Open the API in your browser or use curl: `http://localhost:3000`

Endpoints (for students)

- GET /health — simple health check
- GET /hello — returns a greeting
- GET /random — returns a random number
- GET /time — returns current server time in ISO format
- GET /users — list all users; add `?minAge=18` to filter by age
- GET /users/:id — get a user by id
- POST /users — create a user; JSON body: `{ "name": "Alice", "age": 20 }`
- DELETE /users/:id — delete a user by id

Example responses

GET /hello

```
{ "message": "Hello, World!" }
```

GET /users

```
[
  { "id": 1, "name": "Rick", "age": 23 },
  { "id": 2, "name": "Aadrija", "age": 22 }
]
```

POST /users (example)

Request body:

```
{ "name": "Someone", "age": 21 }
```

Response (201 Created):

```
{ "id": 3, "name": "Someone", "age": 21 }
```

Try it with curl (Windows PowerShell examples)

```
# Get all users
curl http://localhost:3000/users | ConvertFrom-Json

# Create a new user
curl -Method POST http://localhost:3000/users -Headers @{"Content-Type"="application/json"} -Body
(ConvertTo-Json @{name='Someone'; age=21}) | ConvertFrom-Json

# Get a user by id
curl http://localhost:3000/users/1 | ConvertFrom-Json
```

What you can try next

1. Add an endpoint to update a user's name (PATCH /users/:id).
2. Persist users to a JSON file instead of memory.
3. Add request logging middleware that prints method and URL.
4. Add validation so name must be at least 2 characters.

Notes

- This project is intentionally simple to keep focus on HTTP and JSON.
- The in-memory store will reset when the server restarts.