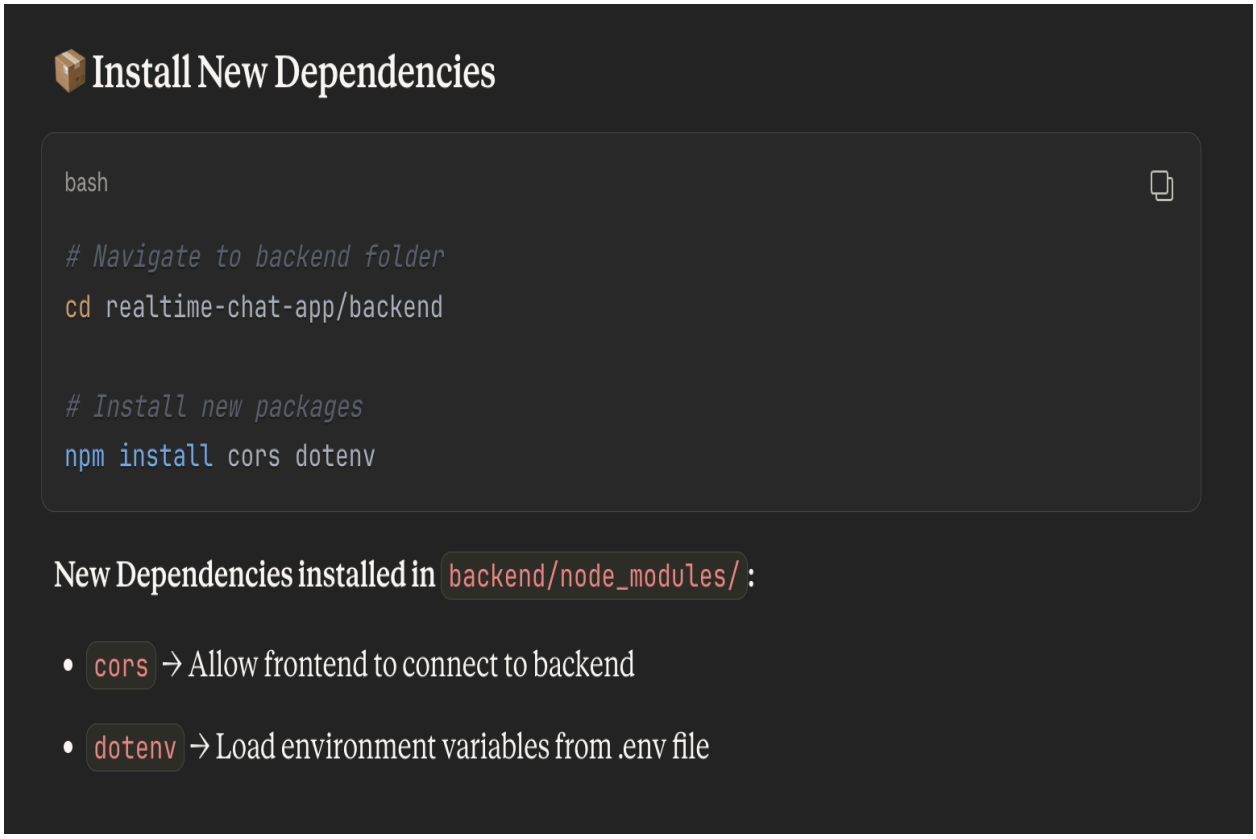
Day 2 Notes: Building REST API with Express

Git Hub->Rudra Arora->https://github.com/RudraArrora/ChitChat-DEMO->
This Link will direct you to the code of the application that we will making

1) Coding Part Of The Session

This is how your file directory should look like

Some other dependencies to installed keep in the mind which folder are installing the dependencies



Part 1: Understanding REST and APIs

What is an API?

API (Application Programming Interface) is a set of rules allowing different software applications to communicate. Defines how requests should be made and what responses to expect.

What is REST?

REST (Representational State Transfer) is an architectural style for designing networked applications using standard HTTP methods.

REST Principles:

- Client-server separation
- Stateless communication (each request independent)
- Cacheable responses
- Uniform interface using standard HTTP methods
- Resource-based (everything is a resource with unique URL)

Part 2: HTTP Methods (Request Types)

GET Method

Purpose: Retrieve data from server without modifying it

Characteristics:

- Safe (no modifications)
- Idempotent (same result when repeated)
- Cacheable
- Parameters in URL

Use Cases: Fetching lists, retrieving details, loading pages

POST Method

Purpose: Send data to create new resource

Characteristics:

- Not safe (modifies server)
- Not idempotent (multiple requests create multiple resources)
- Data in request body
- Not cacheable

Use Cases: Creating accounts, submitting forms, uploading files

---

PUT Method

Purpose: Update existing resource completely

Characteristics:

- Not safe (modifies server)
- Idempotent (same result when repeated)
- Replaces entire resource
- Data in request body

Use Cases: Complete profile updates, replacing documents

PATCH Method

Purpose: Partially update existing resource

Characteristics:

- Not safe (modifies server)
- May be idempotent
- Updates only specified fields
- More efficient than PUT for small changes

Use Cases: Updating single field, changing status

DELETE Method

Purpose: Remove resource from server

Characteristics:

- Not safe (modifies server)
- Idempotent (same result when repeated)
- No request body typically

Use Cases: Deleting accounts, removing posts

Part 3: HTTP Status Codes

Status Code Categories

1xx Informational: Request received, processing continues **2xx Success:** Request successful **3xx Redirection:** Further action needed **4xx Client Error:** Request error from client side **5xx Server Error:** Server failed to fulfill request

---

Common Success Codes (2xx)

200 OK: Request succeeded, data returned **201 Created:** New resource created successfully **204 No Content:** Request succeeded, no content to return

---

Common Client Error Codes (4xx)

400 Bad Request: Invalid request syntax **401 Unauthorized:** Authentication required **403 Forbidden:** Access denied despite authentication **404 Not Found:** Resource does not exist **409 Conflict:** Request conflicts with current state **422 Unprocessable Entity:** Valid syntax but semantic errors

---

Common Server Error Codes (5xx)

500 Internal Server Error: Generic server error **503 Service Unavailable:** Server temporarily unavailable

---

Part 4: Express Middleware

What is Middleware?

Middleware functions have access to request and response objects. They execute during request-response cycle before reaching route handlers.

Purpose:

- Process requests before route handlers
- Modify request/response objects
- End request-response cycle
- Call next middleware in stack

Common Middleware:

- cors: Enable cross-origin requests
- express.json(): Parse JSON request bodies
- express.urlencoded(): Parse URL-encoded bodies
- Static file serving

Part 5: Project Structure

MVC Pattern Basics

Models: Data structure and database interactions **Controllers:** Business logic and request handling **Routes:** URL patterns and HTTP methods

Benefits:

- Organized code
- Separation of concerns
- Easier maintenance
- Reusable components

Part 6: Environment Variables

What are Environment Variables?

Configuration values stored outside code. Used for sensitive data and environment-specific settings.

Common Uses:

- Port numbers
- Database connection strings
- API keys
- Environment mode (development/production)

dotenv Package: Loads environment variables from .env file into process.env. Keeps sensitive data out of codebase.

Part 7: Error Handling

Try-Catch Blocks

Wrap code that may throw errors in try-catch blocks. Prevents server crashes and provides meaningful error responses.

Error Handler Middleware

Special middleware that handles errors. Has four parameters including error object. Catches errors from routes and other middleware.

Part 8: REST API Best Practices

Consistent URL Structure:

- Use nouns, not verbs in URLs
- Use plural names for collections
- Keep URLs simple and intuitive

Proper Status Codes:

- Return appropriate codes for each scenario
- Communicate outcome clearly

Meaningful Responses:

- Include success/failure indicators
- Provide error messages when applicable
- Return relevant data

Validation:

- Validate input data
- Return clear validation errors
- Prevent invalid data from processing