

**COLLEGE CODE :** 8203

**COLLEGE NAME :** AVC

COLLEGE OF ENGINEERING

**DEPARTMENT** **:** CSE

**STUDENT NM ID :**

0E3997591F3B2B683F3423F6F0660B10

**ROLL NO :** 820323104072

**DATE :** 15-09-25

**COMPLETED THE PROJECT**

**NAMED AS :** PHASE 1 TECHNOLOGY

# PROJECT NAME : RESTFUL CONTACT

MANAGEMENT API

**SUBMITTED BY,**

NITHYASRI P

**MOBILE NO :** 9345570080

# IBM-NJ-RESTful Contact Management API

**Phase 2 –** SOLUTION DESIGN AND ARCHITECTURE

## 1. Tech Stack Selection

* **Backend Framework:** Node.js with Express.js (for building REST APIs)
* **Database:** MongoDB (NoSQL) or PostgreSQL (SQL), depending on preference
* **Authentication:** JSON Web Tokens (JWT)
* **API Documentation:** Swagger / OpenAPI
* **Testing Tools:** Postman, Jest / Mocha
* **Hosting Environment:** IBM Cloud or AWS for deployment
* **Version Control:** Git and GitHub for collaboration

## 2. UI Structure / API Schema Design

Although the main focus is the API, a simple front-end (React / basic HTML-CSS) will be used for testing**.**

**UI Structure (for Demo)**

* Login / Registration page
* Contact list page
* Add / Edit contact page

**API Schema (Contact Object)**

*{*

*"ID": "STRING",*

*"NAME": "STRING",*

*"EMAIL": "STRING",*

*"PHONE": "STRING",*

*"ADDRESS": "STRING",*

*"CREATEDAT": "TIMESTAMP",*

*"UPDATEDAT": "TIMESTAMP"*

*}*

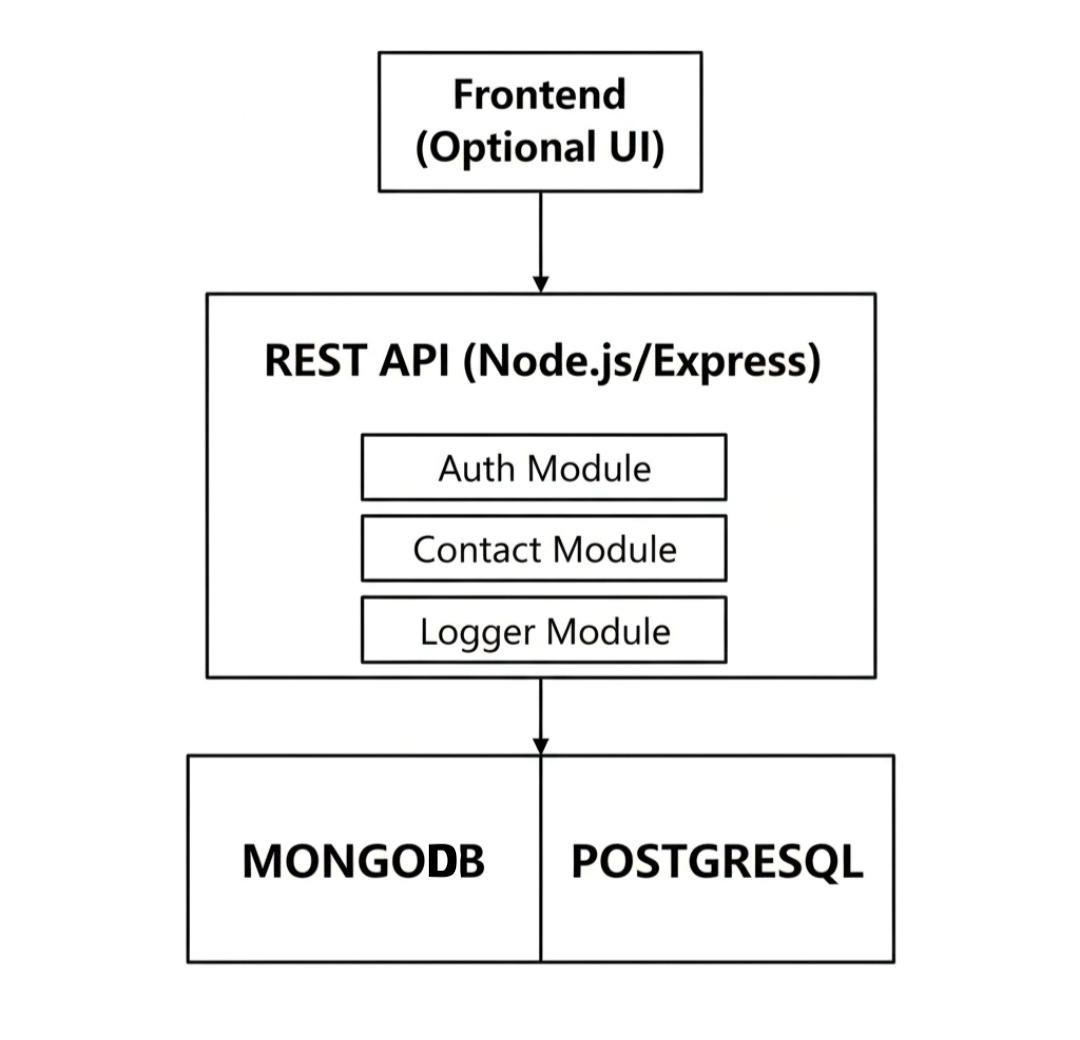
## 3. Data Handling Approach

* **Request Validation:** Use middleware (e.g., Joi or Yup) to validate user inputs.
* **Business Logic Layer:** Handles CRUD logic separately from routing for clean code.
* **Database Layer:** Encapsulates data storage and retrieval functions.
* **Error Handling:** Centralized middleware to send standardized error responses.
* **Security:** Token verification on protected routes and password hashing for users

## 4. Component / Module Diagram

**Modules:**

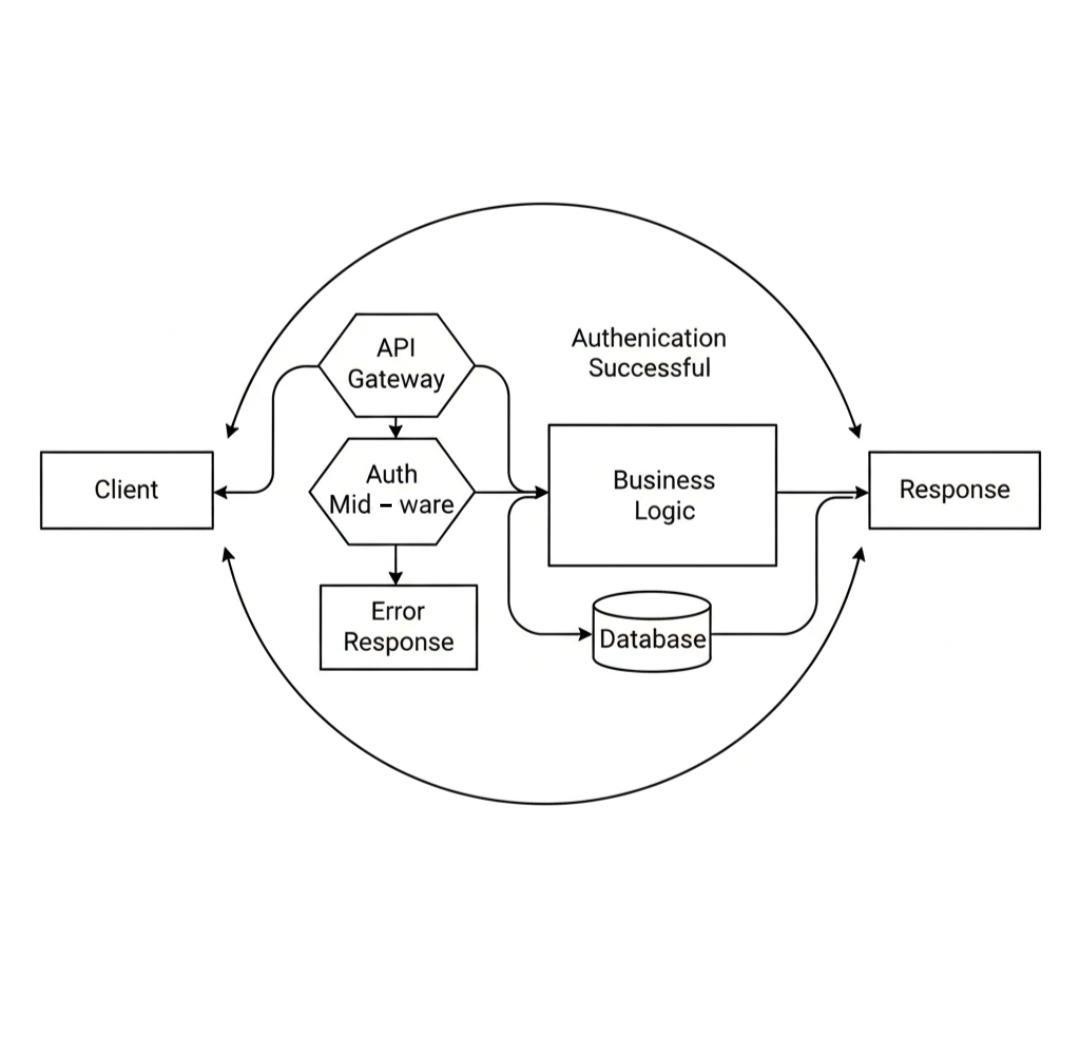
* **Auth Module:** Handles login, registration, token generation.
* **Contact Module:** Handles CRUD operations for contacts**.**
* **Database Module:** Interacts with MongoDB/PostgreSQL**.**
* **Logger Module:** Logs API calls and errors.



## 5. Basic Flow Diagram

**Workflow:**

1. User signs up or logs in → gets token.
2. Front-end or client sends token with each request.
3. API validates token and processes the request.
4. Data is fetched/modified in the database.
5. API returns response to client**.**



## 6. Security & Performance Considerations

* Use HTTPS to secure data in transit.
* Limit API requests per minute (rate limiting).
* Encrypt stored passwords.
* Paginate contact lists for better performance.
* Implement basic monitoring (logs, response times).