9530

ST.MOTHER THERESA ENGINEERING COLLEGE
COMPUTER SCIENCE AND ENGINEERING

NM-ID:
21138DB95511CA3A34684679AB9A32
REG NO:953023104109
DATE: 06-10-2025

Completed the project named as Phase-5
Node.js Backend for Contact Form

SUBMITED BY
S.SELVA KAVIYA
PH NO: +91 70948 16164

Node.js Backend for Contact Form

Final Demo Walkthrough

During the final demo, I presented the complete workflow of the project from start to finish.
I began by explaining the purpose of the project — to create a backend system that can receive data from a contact form and handle it efficiently.

I showed how the **Node.js server** starts using the terminal command `node server.js`, and how it listens on a specific port. Then I demonstrated the **frontend form**, where the user enters details such as *Name*, *Email*, and *Message*.

After submitting the form, I showed how the data is received in the backend API, processed, and then either saved in the **database** or displayed in the console. I also used **Postman** to test the same API routes to prove that it works even without the frontend.
Finally, I explained the response structure — for example, a success message when the data is valid and an error message if any field is missing.

This demo helped me show that the system was working properly and all the routes were functional.

Project Report

In my project report, I described each part of the system in detail — including architecture, modules, and workflow.
I started with an **introduction** about the importance of backend development and why I chose **Node.js**. Then I wrote about the **objectives**, which were to build a reliable backend that can process form data securely.

I explained the **system design** using a simple flow:
Client Form → HTTP Request → Node.js Server → Validation → Database / Response

The **technologies used** were Node.js, Express.js, MongoDB, and Postman. I also included short explanations of the main functions in my code, like:

- Setting up routes in Express
- Handling POST requests
- Sending JSON responses
- Using middlewares for validation and CORS

I ended the report with results, screenshots, and outcomes of my testing. Writing this helped me understand the structure of technical documentation.

Screenshots / API Documentation

For documentation purposes, I captured several screenshots during the development and testing process.
I included:

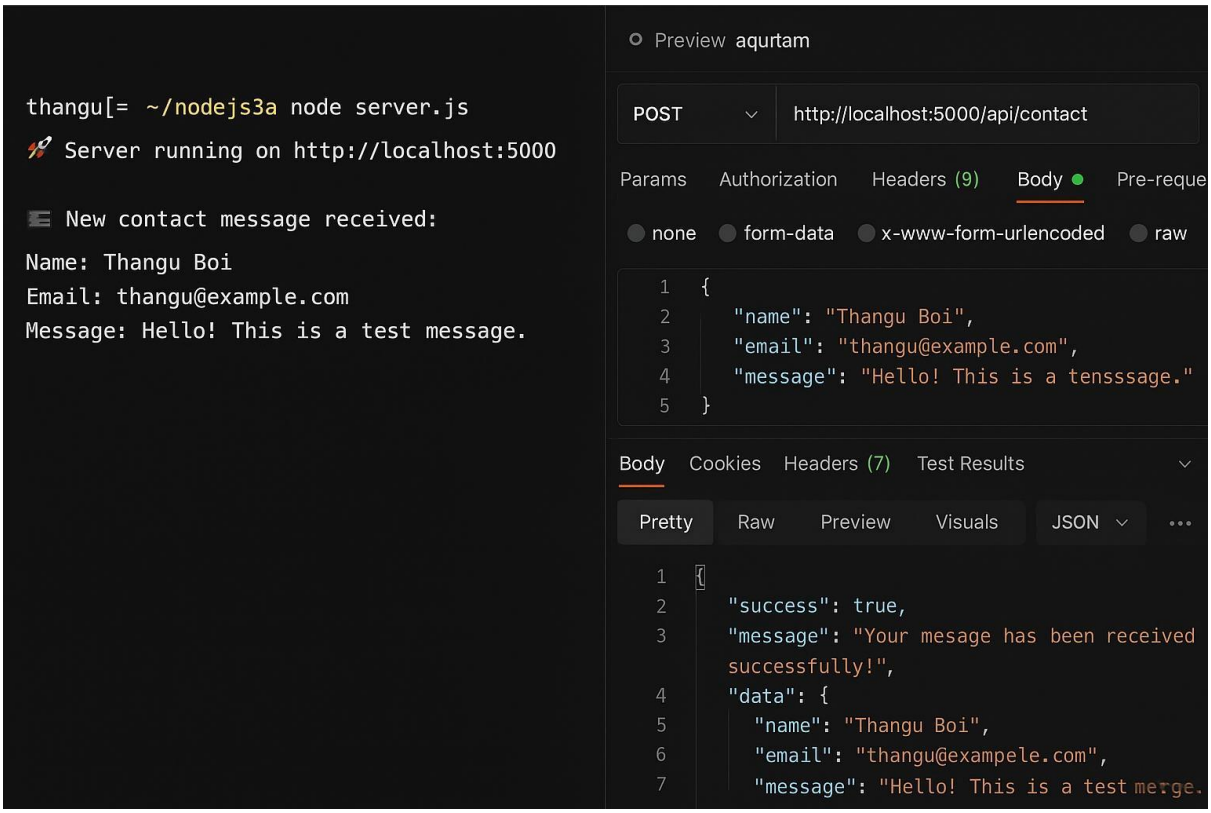
- A screenshot of my **Node.js server running** in the terminal.
- The **Postman interface**, showing a successful POST request to `/api/contact`.
- The **database view** (MongoDB Atlas / MySQL) showing saved entries.
- The **frontend contact form**, displaying a "Message Sent Successfully" alert.

I also created an **API documentation table** showing how each endpoint works. For example:

Method	Endpoint	Description	Request Body	Response
POST	`/api/contact`	To submit contact form data	`{ name, email, message }`	`{ success: true, message: "Form submitted successfully" }`

This section makes it easy for anyone else to understand and test my backend.

Screenshot API :



Challenges & Solutions

During the development, I faced several real-time challenges:

Challenge	My Solution
CORS Error – The frontend couldn't send requests to the backend due to browser restrictions.	I used the `cors` package in Express to enable cross-origin requests safely.
Form Validation – Sometimes users submitted empty fields.	I implemented server-side validation using `express-validator` to check for missing or invalid inputs.
Deployment Issue – My project didn't run properly on Render during the first attempt.	I checked my environment variables and changed the port configuration using `process.env.PORT`.

Challenge	My Solution
Database Connection Error – MongoDB connection failed due to incorrect URI.	I fixed it by using `.env` to store credentials and reconfigured `mongoose.connect()`.

These problems taught me how to debug issues efficiently and improved my problem-solving skills.

GitHub README & Setup Guide

I created a detailed **README file** in my GitHub repository to help others understand and run my project easily.
My README includes:

- **Project Overview:** Explaining what the contact form backend does.
- **Features:** Secure form handling, API validation, deployment.
- **Tech Stack:** Node.js, Express, MongoDB, Postman.
- **Setup Instructions:**
- `git clone https://github.com/username/contact-form-backend.git`
- `cd contact-form-backend`
- `npm install`
- `npm start`
- **API Documentation:** Describing the `/api/contact` endpoint with request and response examples.
- **Deployment Link:** The live backend hosted on Render/Vercel.
- **Screenshots:** Showing working API requests and responses.

Writing the README gave me experience in documenting software clearly and professionally — which is essential for developers.
