




Your First MEAN Stack Project

Full Stack Challenge

Building a Contact Form for Dreaserous Productions

 Duration	 Team Size	 Difficulty
3 Weeks	Solo or Team	Intermediate

What You'll Build

You'll create a contact website for Dreaserous Productions featuring:

- **Contact Form System** - Users submit name, email, subject, and message. All submissions automatically save to a database with timestamps and unique IDs
- **Dynamic FAQ Section** - Questions and answers stored in the database are retrieved and displayed with expandable/collapsible section for each answer
- **Responsive Web Design** - Mobile-responsive layout that adapts from phone screens to desktop screens seamlessly

AI-First Approach

Don't panic! Even if you've never coded before, you'll use AI tools like ChatGPT, Claude, or GitHub Copilot to generate code pieces. Think of it like building with LEGO bricks – AI creates the bricks, you assemble them into your project.

Why This Challenge Matters

Don't get left behind in the AI-powered development revolution. While others struggle with outdated tutorials, you'll master the skills companies actually want in 2025.

- **Industry demand:** Even Fortune 500 companies need developers who can build contact systems and database apps
- **AI-first advantage:** Learn the modern workflow 73% of professional developers now use
- **Portfolio edge:** Production-ready business app beats basic projects every time
- **Future-proof skills:** MEAN stack powers Google, Netflix, and WhatsApp

Reality check: Junior developers with full-stack projects get hired 3x faster. This challenge gives you that edge in 3 weeks.

Step-by-Step Guide

3.1 Phase 1: Environment Setup

1. **Install Node.js:** Download LTS version from <https://nodejs.org>, verify with `node -version`
2. **Install Angular CLI:** Run `npm install -g @angular/cli`, verify with `ng version`
3. **Create project structure:**
 - Create main folder: `mkdir dreaserous-contact`
 - Generate Angular app: `ng new frontend -routing -style=css`
 - Create backend folder: `mkdir backend`
4. **Test installation:** Run `ng serve` in frontend folder, verify localhost:4200 loads Angular welcome page

AI Prompt Example: "Walk me through installing Node.js, Angular CLI, and creating a new Angular project with routing enabled. Include verification steps."

3.2 Phase 2: Backend Development

1. **Initialize backend:** In backend folder, run `npm init -y` then install dependencies: `npm install express sqlite3 cors`
2. **Create Express server:** Build `server.js` with CORS enabled, listening on port 3000
3. **Setup SQLite database:** Create `database.db` file with tables for contacts (id, name, email, subject, message, created_at) and faqs (id, question, answer, display_order)
4. **Build API endpoints:**
 - POST `/api/contacts` - Save contact form submissions to SQLite
 - GET `/api/faqs` - Retrieve all FAQ entries from database
 - GET `/api/contacts` - List all contacts (admin feature)
5. **Test with Postman:** Send POST request to contact endpoint, verify data saves to SQLite database file

AI Prompt Example: "Create a complete Express.js server with SQLite database. Include CORS setup, contact form endpoint that saves to SQLite, and FAQ retrieval endpoint. Provide the full server.js file with database table creation."

3.3 Phase 3: Frontend Development

1. **Generate Angular components:** Use `ng generate component contact-form` and `ng generate component faq-section`
2. **Build contact form:** Create reactive form with validation for name (required), email (required, email format), subject dropdown, and message (required, min 10 characters)
3. **Create FAQ component:** Display questions from SQLite via API with click-to-expand functionality using Angular animations
4. **Setup HTTP service:** Create service to handle API calls, implement form submission and FAQ data retrieval
5. **Add responsive styling:** Use CSS Grid/Flexbox for mobile-first design, test on 320px and 1920px viewports

AI Prompt Example: "Create an Angular reactive form component with name, email, subject dropdown, and message fields. Include form validation, error messages, and HTTP service to submit data to Express API."

3.4 Phase 4: Integration & Deployment

1. **Connect frontend to backend:** Configure Angular proxy for development, test form submission saves to SQLite database
2. **Populate FAQ data:** Insert 5-10 sample questions directly into SQLite database, verify they display and expand correctly
3. **Mobile testing:** Test on actual mobile device or browser dev tools, fix any layout issues
4. **Create documentation:** Write README.md with setup instructions, API endpoints, and feature descriptions
5. **Prepare submission:** Create project demo video (2-3 minutes), zip project files including SQLite database file

AI Prompt Example: "Help me write a comprehensive README.md for my MEAN stack contact form project using SQLite. Include setup instructions, API documentation, and how to run both frontend and backend."

✔ What Your Project Must Have

★ Your Finished Website Will Include

A Working Contact Form:

- Visitors can fill out their name, email, pick a message category, and write their message
- Won't let people submit empty forms or incorrect email addresses
- When someone clicks "Send," their message gets saved permanently
- Shows a "Message sent successfully!" confirmation

A Smart FAQ Section:

- Displays at least 5 common questions about Dreaserous Productions
- Click any question to see the answer appear below it
- Click again to hide the answer (keeps the page tidy)
- Looks professional and easy to read

Build Quality:

- Looks great on phones, tablets, and computers
- Contact form actually connects to your database
- Visitors won't see any error messages or broken features
- Clean, modern design that a real business could use

4.1 Bonus Features (Impress Us for Extra Points)

Only try these after everything above works perfectly:

- **Email alerts:** You get an email notification whenever someone submits the contact form (+5 points)
- **Admin dashboard:** Special page where you can add new FAQ questions without touching code (+10 points)
- **FAQ search:** Visitors can type keywords to find specific questions quickly (+5 points)

- **Dark mode:** Toggle button that switches the whole site to dark colors (+3 points)

Think of it this way: You're building a digital business card for Dreaserous Productions. People can learn about them (FAQ section) and get in touch (contact form). Everything they send gets saved so nothing gets lost, and it works perfectly whether someone visits on their phone or laptop.

🔧 Tools You'll Use

Core Development Tools (Must Have):

- **Node.js (LTS 20+):** JavaScript runtime that powers your server
- **Angular CLI (18+):** Creates the website visitors see - the contact form and FAQ pages
- **Express.js:** Web framework that handles form submissions and API requests
- **SQLite (3.40+):** Lightweight database that stores contact messages and FAQ content
- **VS Code (latest):** Free code editor with excellent extensions support

AI Coding Assistants (Highly Recommended):

- **ChatGPT, Claude, or Gemini:** Generate code snippets and debug issues
- **GitHub Copilot:** Real-time code completion and suggestions
- **Cursor IDE:** AI-powered code editor alternative to VS Code

Testing & Debugging Tools:

- **Postman or Thunder Client:** Test your API endpoints without using the frontend
- **Browser Developer Tools:** Built-in debugging tools for frontend issues

Installation tip: Always choose LTS (Long Term Support) versions when available. These are the most stable and widely supported releases.

📁 Your Project Structure

Here's what your project folder should look like:

```
dreaserous-contact/
|-- frontend/                (Angular website)
|   |-- src/app/
|   |   |-- contact-form/    (Contact form component)
|   |   |-- faq-section/     (FAQ component)
|   |   +-- app.component.*  (Main page)
|   +-- ...
|-- backend/                 (Node.js server)
|   |-- server.js            (Main server file)
|   |-- database.db          (SQLite database)
|   +-- package.json         (Backend dependencies)
+-- README.md                (Setup instructions)
```

📁 How to Submit

7.1 Submission Requirements

- **Where:** Upload to <https://dreaserous.tech/submit> (Will be available during submission period)
- **Deadline:** October 1, 2025 at 11:59 PM IST
- **File format:** .zip file or GitHub repository link
- **Size limit:** 50MB maximum (exclude node_modules folders)
- **Naming convention:** YourTeamName_FullStackChallenge_2025.zip

7.2 What to Include in Your Submission

1. **Complete source code:** Both frontend and backend folders with all files (Exclude node_modules folder)
2. **SQLite database file:** Include database.db with at least 5 sample FAQ entries
3. **README.md file:** Setup instructions and project documentation (see template below)
4. **Demo video:** 2-3 minute screen recording showing your project working
5. **Package files:** Both package.json files (but exclude node_modules folders)

7.3 Demo Video Requirements

- **Duration:** 2-3 minutes maximum
- **Content:** Show contact form submission, FAQ expansion/collapse, mobile responsiveness
- **Format:** MP4, MOV, or AVI (under 25MB)
- **Audio:** Optional narration explaining your features

7.4 README Template

```
# Dreaserous Contact Page

## How to Run
1. Backend: cd backend && npm install && node server.js
2. Frontend: cd frontend && npm install && ng serve
3. Visit: http://localhost:4200

## Features
- Contact form with validation
- FAQ section with expand/collapse
- Mobile-responsive design

## AI Tools Used
- [List the AI tools you used]

## What I Learned
[Brief reflection on your experience]
```

How You'll Be Graded

Category	Weight	What We're Evaluating
Core Functionality	40%	Contact form saves to SQLite, FAQs load from database, form validation works, no console errors
Code Organization	25%	Well-structured components, clean API endpoints, effective AI tool usage, readable code
User Experience	20%	Professional appearance, mobile responsiveness, intuitive navigation, working features
Documentation	10%	Clear setup instructions, feature descriptions, AI tools acknowledgment
Project Demo	5%	Complete walkthrough showing all features working as intended

Pro Tip: A simple, working contact system scores higher than a beautiful but broken interface. Get your core features working first, then polish the design.

Prizes & Recognition

1st Place	INR 1000 + Certificate of Merit + Portfolio Showcase
All Finishers	Certificate of Completion

When You Get Stuck

10.1 Start with AI (Your First Line of Defense)

1. **Copy exact error messages** and paste them into ChatGPT, Claude, or Gemini
2. **Be specific with questions:** Instead of "it's broken," ask "Why isn't my Angular form submitting to the Express server?"
3. **Share relevant code snippets** (10-20 lines max) rather than your entire project
4. **Request step-by-step solutions** when you're completely stuck

Pro tip: AI works best with specific problems. "My contact form won't save to database" gets better help than "my project doesn't work."

10.2 Human Support (When AI Isn't Enough)

- **MDIC 2025 Community:** Join the WhatsApp group for peer discussions and quick help
- **Technical Support:** Email outreach@dreaserous.tech for installation or setup issues

10.3 Quick Fixes for Common Issues

"Angular CLI won't install": Ensure Node.js 18+ is installed, run terminal as administrator (Windows) or use sudo (Mac/Linux)

"Frontend can't reach backend": Add CORS middleware to Express server, check both servers are running on correct ports

"Database not saving data": Verify SQLite table exists, check console for SQL errors, test API endpoints with Postman first

"Everything is broken": Take a break, then start with the simplest working example and add features one at a time

♥ You Can Do This!

Final words: Three weeks from now, you'll have built something real that businesses actually need. Don't let imposter syndrome stop you - every expert was once a beginner.

Remember: AI is your coding mentor, not just a code generator. Ask it to explain everything you don't understand: "What does this function do?" "Why is this necessary?" The more questions you ask, the more you'll learn.

Start simple, finish strong. Your first version doesn't need to be perfect - it just needs to work. You've got this!

🌐 Additional Support & Resources

12.1 Technical Glossary

📖 Essential Terms Explained

Node.js: JavaScript runtime that runs JavaScript code outside browsers. Like having a JavaScript interpreter that can create servers and manage files. *Think of it as the engine that powers your car - Node.js is the engine that powers your backend server.*

Angular: Google's frontend framework for building dynamic web apps. Uses TypeScript and creates single-page applications where content updates without page reloads. *Like a smart TV interface - everything responds instantly without switching channels.*

Express.js: Minimal Node.js framework that simplifies creating web servers and APIs. Reduces hundreds of lines of HTTP handling code to just a few lines. *Like a restaurant waiter - takes orders from customers (frontend) and delivers food from kitchen (database).*

SQLite: File-based database that requires zero configuration. Stores your entire database in a single .db file, unlike complex database servers. *Like a digital filing cabinet in a single folder instead of a whole office building.*

API (Application Programming Interface): Rules that allow different software to communicate. Your contact form sends data to API endpoints - specific URLs that perform database actions. *Like a drive-through menu - you order specific items at specific windows.*

REST API: Web API style using standard HTTP methods. GET retrieves data, POST creates, PUT updates, DELETE removes. Each endpoint has specific URL and purpose. *Like postal service - different types of mail (letters, packages) go to specific addresses for specific actions.*

CORS (Cross-Origin Resource Sharing): Browser security that blocks cross-domain requests unless explicitly allowed. Needed for frontend (port 4200) to talk to backend (port 3000). *Like apartment building security - visitors need permission to enter different floors.*

Component (Angular): Reusable UI piece with HTML structure, CSS styling, and TypeScript logic. Contact form and FAQ are separate components that can be nested and reused. *Like LEGO blocks - each piece has specific design and function, combine them to build bigger structures.*

Reactive Forms (Angular): Angular's form handling with built-in validation, error handling, and data binding. More powerful than template-driven forms for complex validation rules. *Like smart forms that check your answers in real-time and highlight mistakes as you type.*

HTTP Methods: Standard communication methods for web apps. GET retrieves, POST sends, PUT updates, DELETE removes data. Contact form uses POST to submit. *Like different types of phone calls - information requests, placing orders, updating details, canceling services.*

Package Manager (npm): Tool that installs and manages code libraries for your project. Comes with Node.js and handles downloading dependencies like Express, Angular. *Like an app store for code - automatically downloads and installs the tools you need.*

12.2 Terminal Basics for Complete Beginners

>_ Essential Terminal Commands

The terminal (also called command prompt on Windows) lets you interact with your computer using text commands instead of clicking icons. Don't worry - you only need a few basic commands!

Navigation Commands:

- `pwd` - Shows your current location (Print Working Directory)

- `ls` (Mac/Linux) or `dir` (Windows) - Lists files and folders in current location
- `cd foldername` - Enter a folder (Change Directory)
- `cd ..` - Go back to parent folder
- `cd` (without anything after) - Go to your home folder

File/Folder Operations:

- `mkdir foldername` - Create a new folder
- `touch filename` (Mac/Linux) or `ni filename` (Windows) - Create empty file
- `clear` - Clear the terminal screen

Pro Tips:

- Use Tab key to auto-complete folder/file names
- Use Up/Down arrow keys to repeat previous commands
- `Ctrl+C` cancels current command if you make a mistake
- If you get lost, type `pwd` to see where you are

12.3 Complete Setup & Verification Guide

Step-by-Step Installation

Step 1: Install Node.js

1. Visit <https://nodejs.org> in your web browser
2. Click "Download" for the LTS (Long Term Support) version - this is the most stable
3. Run the downloaded installer and click "Next" through all default settings
4. **Test installation:** Open Command Prompt (Windows) or Terminal (Mac/Linux)
 - **Windows:** Press Windows key + R, type `cmd`, press Enter
 - **Mac:** Press Cmd + Space, type "Terminal", press Enter
 - **Linux:** Press Ctrl + Alt + T
5. Type `node -version` and press Enter
6. You should see something like `v18.17.0` - this confirms Node.js is installed

Step 2: Install VS Code (Code Editor)

1. Visit <https://code.visualstudio.com>
2. Click "Download" for your operating system (Windows/Mac/Linux)
3. Run installer with default settings
4. Optional but recommended: Install "Angular Language Service" extension for better Angular support

Step 3: Install Angular CLI

1. Open terminal/command prompt (same as Step 1)
2. Type: `npm install -g @angular/cli` and press Enter
3. Wait 2-3 minutes for installation (downloads many files)

4. If you get permission errors:

- **Windows:** Run Command Prompt as Administrator (right-click → "Run as administrator")
- **Mac/Linux:** Add `sudo` before the command: `sudo npm install -g @angular/cli`

5. Test: Type `ng version` - you should see Angular CLI version information

12.4 Project Creation & Testing

📁 Creating Your Project

Create Project Structure:

1. Open terminal and navigate to where you want your project (like Desktop or Documents)
2. Create main project folder: `mkdir dreaserous-contact`
3. Enter the folder: `cd dreaserous-contact`
4. Create Angular frontend: `ng new frontend -routing -style=css`
 - When prompted "Would you like to add Angular routing?": Type `Y` and press Enter
 - When prompted about stylesheet format: Choose `css` (default)
 - Wait 3-5 minutes while Angular downloads and installs dependencies
5. Create backend folder: `mkdir backend`
6. Enter backend folder: `cd backend`
7. Initialize backend project: `npm init -y` (creates `package.json` automatically)
8. Install backend dependencies: `npm install express sqlite3 cors`

Test Frontend (Angular):

1. Navigate to frontend folder: `cd ../frontend` (go back one level, then into frontend)
2. Start development server: `ng serve`
3. Wait for "compiled successfully" message
4. Open web browser and go to `http://localhost:4200`
5. You should see Angular welcome page with logo and links
6. Success! Leave this terminal window open (server keeps running)

Test Backend (Express):

1. Open a NEW terminal window (don't close the Angular one)
2. Navigate to backend folder: `cd path/to/your/project/backend`
3. Create a simple `server.js` file with this content:

```
const express = require('express');
const app = express();
const PORT = 3000;

app.get('/', (req, res) => {
  res.send('Hello from Dreaserous backend!');
});

app.listen(PORT, () => {
  console.log(`Server running on port ${PORT}`);
});
```

4. Save the file and run: `node server.js`
5. Open browser to `http://localhost:3000`
6. You should see "Hello from Dreaserous backend!"
7. Success! Both your frontend and backend are working

Questions? Join the MDIC 2025 group or email outreach@dreaserous.tech
Remember: Every expert was once a beginner. You've got this!