📋 Detailed Task Breakdown – CloudBlitz Enquiry Management System

🎯 Project Goal

Build a Fullstack Enquiry Management System with React + Vite (Frontend) and Node.js + Express + MongoDB (Backend).

⚡ Important Instruction:
We need a developer who is an expert in AI-powered development tools (Cursor, GitHub Copilot, Codeium, Mintlify, Tabnine, etc.).

- AI tools should be the primary approach for scaffolding, refactoring, documentation, and testing.
- Manual coding should only be done to refine or optimize AI-generated output.
- The developer should proactively explore and apply the maximum possible AI assistance throughout the project lifecycle.

🔷 EPIC 1: Project Setup

Story 1.1 – Repository & Workflow Setup

- Create GitHub repo with frontend and backend folders.
- Setup branching strategy (Gitflow).
- Add .gitignore, README.md, LICENSE.
- Configure ESLint + Prettier with TypeScript support.
- Add Husky + lint-staged for pre-commit hooks.

Story 1.2 – Frontend Initialization

- Initialize React 18 + Vite + TypeScript project.
- Install and configure Tailwind CSS, Radix UI, Lucide React, Class Variance Authority, Tailwind Animate.
- Setup folder structure (components, pages, layouts, routes).
- Add React Router DOM and create starter layout.

Story 1.3 – Backend Initialization

- Initialize Node.js (TypeScript) + Express project.

- Setup folder structure (controllers, models, routes, middlewares).
- Install dependencies: express, mongoose, zod, cors, dotenv, jsonwebtoken, bcryptjs.
- Create base Express server with /api/health route.

🔹 EPIC 2: Authentication & Authorization

Story 2.1 – Backend Auth APIs

- Create User schema: name, email, passwordHash, role, timestamps.
- Implement routes:
  - POST /auth/register
  - POST /auth/login
  - GET /auth/me
- Implement JWT authentication & role-based access middleware.

Story 2.2 – Frontend Auth Pages

- Create Login & Register pages.
- Handle token storage (localStorage/sessionStorage).
- Implement Protected Routes with React Router.

🔹 EPIC 3: Enquiry Management

Story 3.1 – Backend CRUD APIs

- Create Enquiry schema: customerName, email, phone, message, status, assignedTo, timestamps.
- Implement routes:
  - POST /enquiries
  - GET /enquiries (with filters)
  - GET /enquiries/:id
  - PUT /enquiries/:id
  - DELETE /enquiries/:id (soft delete).

Story 3.2 – Frontend Dashboard

- Create Dashboard with tabs for: All, New, In Progress, Closed.
- Add search + filter functionality.
- Display enquiries in a table/list view.
- Add actions: View, Edit, Delete.

Story 3.3 – Enquiry Forms & Modals

- Create Add Enquiry form with validation.
- Create Edit Enquiry modal for status/assignee updates.
- Add Toast notifications for actions.

🔹 EPIC 4: User Management (Admin Only)

Story 4.1 – Backend User APIs

- Implement routes:
  - GET /users
  - POST /users
  - PUT /users/:id
  - DELETE /users/:id

Story 4.2 – Frontend User Page

- Create User Management page with user table.
- Add actions: Create, Edit, Delete users.
- Allow assigning enquiries to staff users.

🔹 EPIC 5: Infrastructure & Deployment

Story 5.1 – Dockerization

- Create Dockerfile for backend.
- Create Dockerfile for frontend (with Nginx).
- Create docker-compose.yml for local dev with MongoDB.

Story 5.2 – Deployment

- Setup .env files for dev/prod.
- Deploy backend (Node + MongoDB) on cloud.
- Deploy frontend on Vercel/Netlify/Cloud.
- Configure proper environment-based API base URLs.

🔹 EPIC 6: Documentation & Testing

Story 6.1 – API Documentation

- Auto-generate API documentation.
- Provide Postman collection or API doc site.

Story 6.2 – Testing

- Setup Jest + Supertest for backend.
- Write unit tests for models & controllers.
- Write integration tests for auth + enquiry workflows.

Story 6.3 – Final Documentation

- Complete README.md with setup steps.
- Add developer guide (how to run, test, deploy).
- Include system architecture diagram.

✅ Final Deliverables

1. Full source code (frontend + backend).
2. Dockerized setup with MongoDB.
3. API documentation.
4. Test suite (unit + integration).
5. Deployment-ready build.
6. Developer documentation.

Credentials & Access

- Share the following after deployment:
  - **Public URL** to access the CRM.
  - **Admin login credentials** (default admin user).
  - **Server login credentials** (EC2/Lightsail SSH).
  - **Database connection details** (if self-hosted, not Atlas).

⚡ Reminder for Developer:
This project is AI-first. We are not just looking for a coder, but for someone who can:

- Use AI development tools expertly.
- Save time by generating, refactoring, and documenting code with AI.
- Treat AI as a collaborator in development, not just a helper.