CHINMOY SHARMA

BACKEND DEVELOPER

Chinmoysharma001@gmail.com | +91 60031-21707

LinkedIn | GitHub

ABOUT ME

Backend developer with hands-on experience building scalable backend systems, distributed applications, and a real-time communication platform using Node.js, TypeScript, Redis, PostgreSQL, Prisma ORM, Docker, Socket.IO and Flask. Experienced in designing REST APIs, authentication systems, asynchronous Job processing and modular backend architectures. Passionate about backend engineering, distributed systems and solving real-world software problems.

SKILLS

- **LANGUAGES:** Python, JavaScript, TypeScript
- **DATABASES:** PostgreSQL, MongoDB, Redis
- **BACKEND:** Flask, Node.js, Express.js, Socket.IO, REST APIs
- **TOOLS:** Docker**,** Git, Github, Postman, Prisma ORM

EDUCATION

Bachelor of Science: Information Technology(July 2021 - Jul 2024)
B.Borooah College(Gauhati University): Guwahati, India

- CGPA: 6.12

PROJECTS

DISTRIBUTED JOB QUEUE | Node.js, Redis, Docker

- Built a distributed background job processing system with support for multiple concurrent workers and asynchronous task execution.
- Implemented visibility timeouts and retry mechanisms to prevent duplicate processing and recover failed/stale jobs.
- Designed fault-tolerance queue coordination using Redis to handle worker crashes and ensure reliable job delivery.
- Developed stale job recovery logic to re-queue unacknowledged tasks and improve system resiliency.

REAL-TIME CHAT BACKEND | Node.js, TypeScript, Socket.IO, PostgreSQL, Prisma ORM, Redis

- Engineered a scalable real-time chat backend supporting authenticated room-based messaging.
- Implemented JWT based authentication and authorization.
- Built online message tracking and unread message counting.
- Integrated Redis Pub/Sub to support scalable real-time communication.
- Used Prisma ORM with PostgreSQL for efficient database access.

FLASK BLOG-API

- Developed a RESTful blog API with JWT based authentication and role-based access.
- Implemented full CRUD functionality for posts, comments and user-interactions.
- Designed and optimized APIs with Pagination & Filtering to efficiently handle large datasets.
- Integrated Swagger/OpenAPI documentation for testing and clear API usage.
- Containerized the application using Docker and Docker Compose for consistent deployment.
- Structured the application using modular routing and scalable backend architecture.