Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	23 February 2026
Team ID	LTVIP2026TMIDS90304
Project Name	ONLINE COMPLAINT REGISTRATION AND MANAGEMENT SYSTEM
Maximum Marks	4 Marks

Technical Architecture:

The application follows a **client-server model**. The **Frontend** acts as the client, utilizing the **Axios** library to communicate with the **Backend** server via **RESTful APIs**. The Backend handles business logic and manages data flow to and from the **MongoDB** database.

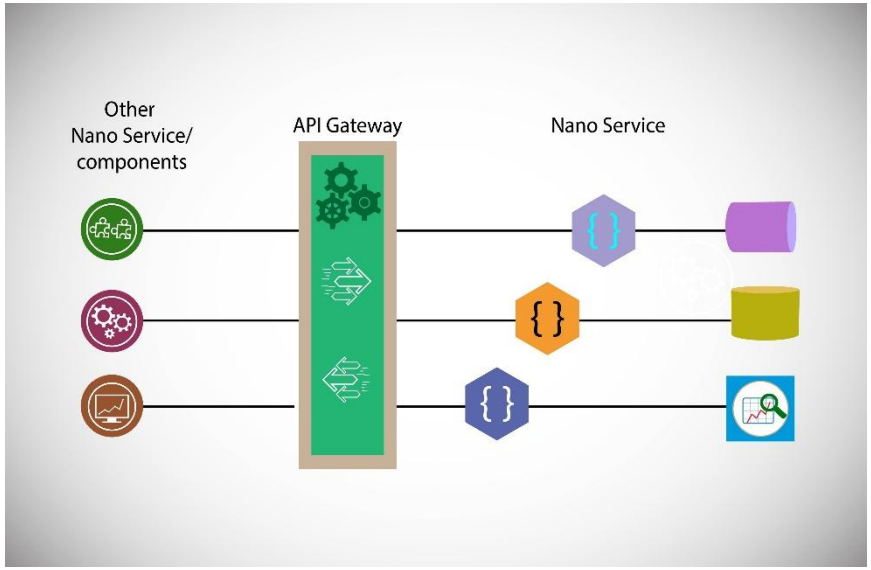

Table-1: Components & Technologies

S.No	Component	Description	Technology
1.	User Interface	Responsive Web UI for three roles: Ordinary User, Agent, and Admin.	React.js, Bootstrap, Material UI .
2.	Application Logic-1	Server-side logic for routing, middleware, and API development.	Node.js, Express.js.
3.	Application Logic-2	Real-time bidirectional communication for the built-in messaging feature.	Socket.io.
4.	Application Logic-3	Object-Document Mapping (ODM) to perform CRUD operations on the database.	Mongoose.
5.	Database	NoSQL database for flexible storage of user profiles, complaints, and messages.	MongoDB.
6.	Cloud Database	Fully managed cloud database service for reliable data access.	MongoDB Atlas.
7.	File Storage	Storage for complaint attachments like images and documents showcasing defects.	Local Filesystem or Cloudinary/AWS S3
8.	External API-1	Library used for making HTTP requests from the frontend to backend endpoints.	Axios.
9.	External API-2	Real-time communication API mentioned for advanced conferencing features.	WebRTC API.
10.	Infrastructure	Local and cloud deployment configurations for the development and testing environment.	Local Server (Node), GitHub (Version Control).

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Utilizes widely supported open-source technologies for full-stack development.	MERN Stack (MongoDB, Express, React, Node).
2.	Security Implementations	Password hashing for protection and secure cross-origin resource sharing.	Bcrypt, CORS, JWT.
3.	Scalable Architecture	Designed with a modular directory structure (Models, Routes, Controllers) for growth.	3-Tier Architecture (Client-Server-DB).
4.	Availability	Accessible through local servers and deployable to highly available cloud environments.	Express Server, MongoDB Atlas.
5.	Performance	Minimalist framework usage ensuring fast server-side routing and efficient data exchange.	Express.js, JSON-based NoSQL.