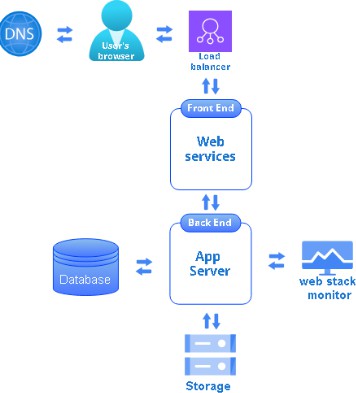
# Project Design Phase-II Technology Stack (Architecture & Stack)

|  |  |
| --- | --- |
| Date | 26 June 2025 |
| Team ID | LTVIP2025TMID53123 |
| Project Name | ResolveNow: Your Platform for Online  Complaints |
| Maximum Marks | 4 Marks |

**Technical Architecture – ResolveNow**

ResolveNow is built using a **client-server architecture**, ensuring smooth interaction between users, agents, and admins. The system is divided into three main layers: **Frontend**, **Backend**, and **Database**. RESTful APIs connect the layers, enabling secure and efficient data exchange. Real-time chat and notifications are supported using **Socket.IO**.



# Architecture Guidelines – ResolveNow

* The system includes core blocks:
  1. **Frontend:** React.js (Material UI, Bootstrap)
  2. **Backend:** Node.js + Express.js (REST APIs)
  3. **Database:** MongoDB (User, Complaint, Chat, Feedback data)

# Infrastructure:

* 1. Local setup for development
  2. Cloud deployment via Vercel (frontend), Render or Railway (backend), MongoDB Atlas (database)

# External Interfaces:

* 1. Gmail SMTP for emails
  2. Google OAuth for login
  3. Optional: Twilio for SMS

# Data Storage:

* 1. All structured data in MongoDB
  2. Files/images stored via Firebase or AWS S3 (optional)

# ML Model (Optional):

* 1. Future-ready for smart routing or auto-prioritization using ML

**Table-1: Components & Technologies**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | **User Interface** | How user interacts with the application (Web UI, etc.) | **React.js**, **HTML**, **CSS**, **JavaScript**, **Material UI**, Bootstrap |
| 2 | **Application Logic-1** | Logic for complaint submission, status update, and routing | **Node.js**, **Express.js** |
| 3 | **Application Logic-2** | Voice-to-text service for verbal complaints (future enhancement) | **IBM Watson Speech to Text (STT)** |
| 4 | **Application Logic-3** | Chatbot to guide users during complaint submission (optional) | **IBM Watson Assistant** |
| 5 | **Database** | Data storage for users, complaints, chats, etc. | **MongoDB (NoSQL)** |
| 6 | **Cloud Database** | Cloud-hosted data services (optional) | **MongoDB Atlas**, **IBM Cloudant** |
| 7 | **File Storage** | Image or document uploads for complaints | **Local Filesystem**, **Cloudinary**, or **AWS S3** |
| 8 | **External API-1** | Weather integration if complaint is weather-related | **OpenWeatherMap API** (example) |
| 9 | **External API-2** | For user identity verification | **UIDAI Aadhar API** |
| 10 | **Machine Learning Model** | Predict complaint urgency or sentiment (future scope) | **Sentiment Analysis Model**, **ML Classifier** |
| 11 | **Infrastructure** | Hosting backend/frontend on cloud/local | **Localhost**, **Render**, **Railway**, **Cloud Foundry**, **Kubernetes** |

**Table-2: Application Characteristics**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology Used** |
| 1 | **Open-Source Frameworks** | Libraries and frameworks used | **React.js**, **Express.js**, **Node.js**, **Mongoose**, **Socket.io** |
| 2 | **Security Implementations** | Authentication, Authorization, Data Protection | **JWT**, **bcrypt.js**, **CORS**, **HTTPS**, **SHA-256**, **Helmet** |
| 3 | **Scalable Architecture** | Modular design for performance and growth | **3-tier architecture**, **Microservices-ready**, **REST APIs** |
| 4 | **Availability** | Ensures uptime, handles traffic | **Load Balancers**, **Cloud Deployment**, **Clustered MongoDB** |
| 5 | **Performance** | Optimized code for response time and user experience | **Axios**, **CDN**, **Caching (Redis optional)**, **Lazy Loading** |

**References**

1. C4 Model for Visualising Software Architecture. Retrieved from: <https://c4model.com/>
2. IBM Developer: Online Order Processing System During Pandemic. Retrieved from: [https://developer.ibm.com/patterns/online-order-](https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/) [processing-system-during-pandemic/](https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/)
3. IBM Cloud Architecture Center. Retrieved from: <https://www.ibm.com/cloud/architecture>
4. AWS Architecture Center. Retrieved from: <https://aws.amazon.com/architecture>
5. “How to Draw Useful Technical Architecture Diagrams” – Medium Article. Retrieved from: [https://medium.com/the-internal-startup/how-to-](https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d) [draw-useful-technical-architecture-diagrams-2d20c9fda90d](https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d)