**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | June 2025 |
| Team ID | LTVIP2025TMID58285 |
| Project Name | Service Desk for Customer Complaint Resolution |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2 Example: Order processing during pandemics for offline mode.

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/

|  |  |  |
| --- | --- | --- |
|  | |  | | --- | | Guidelines:  Include all the processes (As an application logic  Technology Block)  Provide infrastructural demarcation (Local / Cloud)  Indicate external interfaces (third party API’s etc.) Table-1 : Components & Technologies:  Indicate Data Storage components / services  Indicate interface to machine learning models  Table-2: Application Characteristics | |

**Table -1 : Components & Technologies :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | User Interface | How user interacts with the application (e.g., Web UI) | HTML, CSS, JavaScript, React.js |
| 2 | Application Logic-1 | User registration and login process | Node.js, Express.js |
| 3 | Application Logic-2 | Complaint submission and tracking | Node.js, Express.js |
| 4 | Application Logic-3 | Real-time chat between user and agent | Socket.io (optional), Express.js |
| 5 | Database | Stores complaint, user, agent data | MongoDB |
| 6 | Cloud Database | Cloud-hosted scalable database | MongoDB Atlas |
| 7 | File Storage | Store user-uploaded complaint images | MongoDB GridFS / Base64 (optional) |
| 8 | External API-1 | Email service for notifications (if implemented) | Nodemailer |
| 9 | External API-2 | OTP verification (if implemented) | Twilio / Firebase OTP |
| 10 | Machine Learning Model | (Not used currently, optional) | (N/A) |
| 11 | Infrastructure (Server / Cloud) | Local development environment | Node.js server, localhost |

**Table - 2 : Application Characteristics :**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology Used** |
| 1 | Open-Source Frameworks | Use of free, community-maintained frameworks | React.js, Node.js, Express.js |
| 2 | Security Implementations | Password hashing, secure Mongo URI, role-based access | bcrypt.js, dotenv |
| 3 | Scalable Architecture | Modular frontend/backend, MongoDB Atlas handles scaling | Express.js, MongoDB Atlas |
| 4 | Availability | Can be deployed on cloud servers or local servers for 24/7 availability | MongoDB Atlas, Vercel (optional) |
| 5 | Performance | REST APIs, optimized frontend, lazy loading | Axios, React.js, Express.js |

References:

https://c4model.com/ https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/ https://www.ibm.com/cloud/architecture https://aws.amazon.com/architecture https://medium.com/the-internalstartup/how- to-draw-useful-technical-architecture-diagrams-2d20c9fda90d