**🏗️** Technology Stack – Lease Management System

**📅 Date:** 31 January 2025  
**🧾 Project Name:** Lease Management System  
**🧑‍💻 Team ID:**  LTVIP2025TMID30244  
**🎯 Maximum Marks:** 4 Marks

**📌 Technical Architecture**

The Lease Management System is built using a **3-tier architecture** consisting of:

* **Presentation Layer** (User Interface – Web & Mobile)
* **Application Layer** (Business Logic)
* **Data Layer** (Database & File Storage)

A cloud-hosted model is preferred for scalability and availability, supported by microservices and REST APIs.

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

| **S.No** | **Component** | **Description** | **Technology Used** |
| --- | --- | --- | --- |
| 1. | **User Interface** | Web & Mobile App | React.js (Web), Flutter (Mobile), HTML/CSS |
| 2. | **Application Logic-1** | Backend for user, lease & rent modules | Node.js / Express.js |
| 3. | **Application Logic-2** | Notification engine | Python (Scheduler) + Firebase Cloud Messaging |
| 4. | **Application Logic-3** | Authentication, roles & access control | JWT, OAuth 2.0 |
| 5. | **Database** | Relational DB for leases, users, payments | PostgreSQL / MySQL |
| 6. | **Cloud Database** | Managed DB Service | AWS RDS / Google Cloud SQL |
| 7. | **File Storage** | Lease documents & scanned files | AWS S3 / Google Cloud Storage |
| 8. | **External API-1** | Payment gateway integration | Razorpay / Stripe API |
| 9. | **External API-2** | SMS & Email notifications | Twilio, SendGrid |
| 10. | **Machine Learning Model** | Predict rent delays (optional) | Python + Scikit-learn (Predictive Analytics) |
| 11. | **Infrastructure** | Deployment | Docker + Kubernetes on AWS EC2 or GCP GKE |

**📋 Table-2: Application Characteristics**

| **S.No** | **Characteristic** | **Description** | **Technology Used** |
| --- | --- | --- | --- |
| 1. | **Open-Source Frameworks** | React.js, Node.js, PostgreSQL, Docker | All open-source |
| 2. | **Security Implementations** | JWT auth, AES-256 encryption, HTTPS, IAM policies | OAuth 2.0, HTTPS, IAM, OWASP compliance |
| 3. | **Scalable Architecture** | Microservices, containerized services | Docker, Kubernetes, REST APIs |
| 4. | **Availability** | Cloud deployment with auto-scaling, load balancing | AWS/GCP Load Balancer, Kubernetes Horizontal Pod Autoscaler |
| 5. | **Performance** | Redis cache for faster queries, CDN for static files | Redis, Cloudflare CDN, NGINX reverse proxy |

**🌐 Architecture Diagram (Suggested Layout)**

less

CopyEdit

[User Interface]

(React / Flutter Apps)

|

[API Gateway / Load Balancer]

|

-------------------------------

| | |

[Auth Service] [Lease Mgmt] [Payment Engine]

| | |

PostgreSQL AWS S3 Razorpay API

| |

Redis Cache Notification Service

|

Firebase / Twilio

Infrastructure: Docker + Kubernetes (AWS/GCP)