

## Project Design Phase-II

### Technology Stack (Architecture & Stack)

Date	27 JUNE 2025
Team ID	7264E40C79CF15DA21F41ABA4F4825CD
Topic	Lease Management System

#### Technical Architecture (overview)

The Lease Management System is designed as a modular, cloud-native application that separates user-facing interfaces, business logic, and data storage. Primary actors are **Tenant, Landlord / Property Manager, Admin**, and external services such as **Payment Gateway, Identity Verification**, and **Notification Providers**.

Key architectural principles:

- **Clear infrastructural demarcation:** UI and mobile clients (edge) → API / Backend services (application layer) → Storage & analytics (data layer). Core components run in the cloud; optional on-prem integrations are supported via secure APIs.
- **Resilient integrations:** external services (payments, KYC, SMS/email) are called asynchronously where possible and retried on transient failure.
- **Data separation:** transactional relational store for lease and payment records; object storage for documents and photos; search index for fast property queries; audit log store for compliance.
- **Security & compliance:** encryption in transit & at rest, RBAC, audit trails, configurable data retention for tenancy law / privacy compliance.
- **Extensibility:** microservices / modular services for Payments, Lease Engine, Maintenance, and Notifications; clear APIs for vendor integrations (vendor portals, accounting software).
- **Optional ML:** modules for credit/risk scoring, dynamic pricing, or fraud detection can be plugged in via model endpoints.

Typical data flows include: tenant browses properties → applies for lease → system checks availability and tenant verification → lease generated and e-signed → payment processed via gateway → documents stored in object storage → lease and payment records persisted in DB → notifications sent.

**Table-1 : Components & Technologies**

S.No	Component	Description	Recommended Technology / Example
1.	User Interface (Web & Mobile)	Tenant/Landlord/Admin dashboards, property browsing, application & e-sign flows, responsive design.	React (Web) + React Native (Mobile) or Angular; Tailwind / Material UI for styling
2.	API Gateway / Edge	Single endpoint for clients, request routing, authentication, rate limiting.	AWS API Gateway / NGINX / Kong
3.	Authentication & Identity	User sign-up, social login, MFA, role management (tenant, landlord, admin).	Auth0 / Keycloak / AWS Cognito
4.	Lease Engine (Application Logic 1)	Core business logic: create/approve leases, term calculations, prorations, renewals.	Node.js (Express) / Spring Boot (Java) / .NET Core
5.	Payments Service (Application Logic 2)	Integrates with payment provider to handle deposits, recurring rent, refunds.	Stripe / PayPal / Razorpay; server-side processing in microservice

S.No	Component	Description	Recommended Technology / Example
6.	Maintenance & Vendor Workflow	Manage tickets, assign vendors, track status and invoices.	Backend microservice + third-party vendor APIs / Trello-like workflow
7.	Document Management & e-Signature	Store lease PDFs, photos, inspection reports; support e-signatures.	AWS S3 / Azure Blob + DocuSign / Adobe Sign
8.	Relational Database	Transactional storage for users, leases, payments, audit logs.	PostgreSQL / Amazon RDS / Azure SQL
9.	Object Storage	Photos, signed PDFs, inspection images.	AWS S3 / Azure Blob Storage / Google Cloud Storage
10.	Search & Indexing	Fast property search, filtering and full-text search for listings.	Elasticsearch / OpenSearch
11.	Cache & Session Store	Short-term cache (property pages, sessions), rate limiting.	Redis / Memcached
12.	Notifications & Messaging	Email, SMS and in-app notifications for payments, renewals, maintenance updates.	Twilio (SMS), SendGrid (Email), Firebase Cloud Messaging (push)
13.	Analytics & Reporting	Lease portfolio KPIs, delinquencies, occupancy rates, exportable reports.	Data pipeline to Redshift / BigQuery; Metabase / Superset
14.	Monitoring & Logging	Health checks, centralized logs,	ELK stack (Elasticsearch/Logstash/Kibana) or CloudWatch + Grafana

S.No	Component	Description	Recommended Technology / Example
		alerting for failures and security events.	
15.	CI/CD & Containerization	Infrastructure as code, automated builds & deployments.	Docker, Kubernetes (EKS/GKE/AKS), GitHub Actions / Jenkins / GitLab CI
16.	External Integrations	KYC / identity, accounting software, utility registration, property listing syndication.	REST APIs / Webhooks; providers like Onfido (KYC), QuickBooks (accounting)
17.	Backup & DR	Scheduled backups, snapshot policies, disaster recovery plan with RTO/RPO.	Cloud snapshots (RDS snapshots), cross-region replication

**Table-2 : Application Characteristics**

S.No	Characteristic	Description	Technology / Approach
1.	Security	End-to-end encryption, RBAC, MFA, audit trails, PCI compliance for payments.	TLS, OAuth2 / OpenID Connect, server-side encryption, PCI DSS controls
2.	Scalability	Horizontally scale API and worker services for peak leasing seasons.	Kubernetes autoscaling, managed DB read replicas, CDN for static assets
3.	Availability	Minimize downtime for critical services (auth, payments, document access).	Multi-AZ deployment, load balancers, health checks & failover

S.No	Characteristic	Description	Technology / Approach
4.	Performance	Fast property search and quick transaction responses.	Caching (Redis), indexed DB queries, async background jobs (Celery / Sidekiq)
5.	Maintainability	Modular services, documented APIs, automated tests and CI.	Microservices or modular monolith, OpenAPI (Swagger), unit/integration tests
6.	Compliance & Data Retention	Configurable retention windows, consent capture, right to erasure workflows.	Policy engine, audit logs, data export tools
7.	Reliability	Ensure payment/lease transactions are consistent and retry on failure.	ACID DB transactions, idempotent APIs, durable queues (SQS/Kafka)
8.	Usability & Accessibility	Simple flows for non-technical users and WCAG accessibility.	UX design, WCAG AA testing, mobile responsive layouts
9.	Observability	Centralized logs, traces and metrics for debugging and SLA monitoring.	OpenTelemetry, Grafana, ELK / Cloud observability tools
10.	Extensibility	Ability to add new integrations (utility registration, insurance, analytics).	Well-versioned APIs, plugin integration patterns

### Integration & External Interfaces (summary)

- **Payment Gateway:** Stripe / Razorpay — process deposits, recurring payments, webhooks for payment status.

- **Identity / KYC:** Onfido, Jumio, or government ID verification APIs for tenant screening.
- **Notification Providers:** Twilio (SMS), SendGrid (email), Push services for mobile.
- **Accounting / ERP:** Optional sync to QuickBooks / Xero for landlord payouts and reconciliation.
- **Property Listing Syndication:** APIs to push listings to partner portals (Zillow, MagicBricks etc.) if required.

### **Deployment options & recommended stack (example deployment)**

- **Cloud provider:** AWS (recommended) or GCP / Azure.
- **Container orchestration:** Kubernetes (EKS/GKE/AKS) for microservices.
- **DB:** Managed PostgreSQL (RDS) + read replicas.
- **Object storage:** S3 with lifecycle rules (archive to Glacier).
- **CI/CD:** GitHub Actions pipelines → build → run tests → push Docker images → deploy to clusters.
- **Infra as Code:** Terraform (or CloudFormation) for reproducible environments.