Phase 3: High-Level System Architecture and Technology Stack

This section proposes a robust, scalable, and secure three-tier architecture for the Property Management Document CRM. The proposed technology stack is based on modern, widely-adopted, and cost-effective open-source technologies, ensuring long-term maintainability and performance.

1. Architectural Overview

The system will follow a standard **Three-Tier Architecture**:

1. **Presentation Tier (Frontend):** The user interface accessed by the Property Manager via a web browser.
2. **Application Tier (Backend):** The core logic, API endpoints, and business rules.
3. **Data Tier (Storage):** The databases and file storage systems for both structured metadata and unstructured documents.

Tier	Primary Function	Proposed Technology Stack	Rationale
Presentation	User Interface (UI) and User Experience (UX)	**React** or **Vue.js** (JavaScript Framework)	Provides a fast, responsive, and modern single-page application (SPA) experience, which is ideal for a heavy-use professional tool like a CRM.
Application	Business Logic, API, Authentication, Alerts	**Python (FastAPI/Django)**	Python is excellent for rapid development, has strong libraries for data processing (e.g., document parsing for full-text search), and is highly maintainable. FastAPI offers high performance, while Django provides a comprehensive

			"batteries-included" framework.
Data	Storage and Retrieval	**PostgreSQL** (Relational Database) & **AWS S3** (Object Storage)	PostgreSQL is robust, supports complex querying (essential for advanced search), and handles structured metadata reliably. AWS S3 (or equivalent cloud storage like Azure Blob Storage or Google Cloud Storage) is the industry standard for secure, highly available, and cost-effective storage of large volumes of unstructured files (documents).

2. Component Breakdown

2.1. Presentation Tier (Frontend)

The frontend will be a Single Page Application (SPA) that communicates with the backend exclusively through RESTful APIs.

- **Key Features:**
  - **Dashboard:** Overview of critical dates, recent activity, and pending tasks.
  - **Property/Tenant Views:** Dedicated pages for managing all data and documents related to a specific property or tenant.
  - **Document Viewer:** An integrated viewer for common file types (PDF, images) to avoid unnecessary downloads.
  - **Responsive Design:** Ensures usability on desktop and tablet devices.

2.2. Application Tier (Backend)

The backend is the brain of the system, handling all data operations, security, and automation.

- **API Gateway:** All frontend requests pass through a secure API layer.
- **Business Logic:** Manages the creation, updating, and deletion of Property, Tenant, and Document records, enforcing all business rules (e.g., linking a document to a valid Property ID).
- **Authentication & Authorization:** Handles user login (Property Manager) and ensures only authorized actions are performed.
- **Alerts & Scheduling Service:** A dedicated service (e.g., using Celery with Redis) to run scheduled tasks, such as checking for upcoming lease expirations and sending automated email/in-app alerts to the Property Manager.
- **Document Processing Service:** A microservice responsible for:
  - Uploading files to S3.
  - Extracting text from documents (OCR for images, text extraction for PDFs) to feed the Full-Text Search index.

2.3. Data Tier (Storage)

The separation of structured and unstructured data is crucial for performance and cost efficiency.

Storage Type	Data Stored	Technology	Purpose
Structured Data	Property/Tenant Metadata, Document Metadata (Type, Expiration Date, Tags), User Accounts, Audit Logs.	**PostgreSQL**	Provides transactional integrity, complex relational querying, and built-in support for advanced features like JSONB for flexible data and full-text indexing.
Unstructured Data	The actual document files (PDFs, JPEGs, DOCX).	**AWS S3** (or equivalent)	Offers massive scalability, high durability (99.999999999% data durability), and cost-effective storage. Access is managed via pre-signed URLs generated by the backend, ensuring

			secure, time-limited access.
Search Index	Indexed text content from all documents.	**Elasticsearch** or **PostgreSQL's Full-Text Search**	Enables the required fast and powerful full-text search capability across all document content.

3. Integration Points

While the user indicated no old integrations, the architecture is designed with future integration in mind:

- **Property Management Website:** The CRM's API can be used to securely expose non-sensitive data (e.g., property status) to the main property management website.
- **Accounting Software:** Integration with platforms like QuickBooks or Xero can be achieved via dedicated API connectors to sync maintenance invoices and financial documents.
- **Email Service:** Integration with a transactional email service (e.g., SendGrid, Mailgun) is necessary for sending automated alerts and notifications.

This architecture provides a solid, modern foundation that can be scaled horizontally as the number of properties and documents grows. The next phase will detail the non-functional requirements, including security and compliance.