1. Define the Core Features

From the document, the system should support:

- **File Upload & Storage**
  - Drag-and-drop or browse files
  - Secure cloud storage (AWS S3, Firebase Storage, or another provider)
  - File metadata management (title, description, version, status, etc.)
- **Document Collaboration & Version Control**
  - Allow team members to edit and track changes
  - Show differences between two document versions
  - Revert or compare versions
- **Electronic Signatures**
  - Allow customers, vendors, and employees to sign documents digitally
  - Track signature status (pending, signed, rejected)
- **Permissions & Sharing**
  - Define roles (Owner, Editor, Viewer, Signer)
  - Share docs with internal/external users with specific permissions
- **File Management & UI**
  - File preview (PDF, images, Office files)
  - Filtering and sorting (by date, status, name)
  - Bulk actions (share, move, delete)
  - Archiving and retention policies

---

2. Tech Stack Recommendation

Since you're using **React, Node.js, AWS, PostgreSQL, MongoDB, and Firebase**, here's a suggested approach:

Frontend (React)

- **Component-based UI**: Use React with **Tailwind CSS** or **Material UI**.
- **Drag-and-Drop File Upload**: Use a library like **react-dropzone**.
- **File Preview Support**: Integrate **PDF.js** for PDFs and **react-file-viewer** for Office files.
- **Role-based Access Control**: Show/hide UI elements based on user permissions.

Backend (Node.js + Express)

- **File Upload API** (`POST /documents/upload`)
  - Use **AWS S3 Pre-Signed URLs** for secure uploads.
  - Validate file type & size before storing.
  - Store metadata in **PostgreSQL** (for structured data) or **MongoDB** (for flexibility).

- **Version Control**
  - Track versions in **PostgreSQL** with references to previous versions.
  - Compare file versions with a diffing tool for text-based documents.
- **E-Signatures**
  - Use **DocuSign API** or **Eversign API** for digital signatures.
  - Store signature status in the database.
- **Access Control**
  - Implement **JWT Authentication** for secure access.
  - Define **roles and permissions** in PostgreSQL.

---

3. Development Roadmap

Phase 1: Core Document Management

✅ Set up **file upload API** with AWS S3 or Firebase
✅ Build **React UI** for file uploads, previews, and listing
✅ Implement **basic permissions** (Owner, Editor, Viewer)

Phase 2: Collaboration & Version Control

✅ Create a **versioning system** (store file history)
✅ Add **diff comparison** for text-based files
✅ Implement **sharing & role management**

Phase 3: E-Signatures & Advanced Features

✅ Integrate **DocuSign/Eversign API**
✅ Track **signature requests & statuses**
✅ Enable **bulk document actions** (share, delete, move)
✅ Implement **filters & sorting**

Next Steps

1. **Decide on Cloud Storage** (AWS S3 vs Firebase)
2. **Start with File Upload API & Metadata Storage**
3. **Build the React UI with File Management Features**
4. **Integrate Version Control & Diff Comparison**
5.