

# Feature #6: Ready to Implement! 🚀

## Planning Complete ✅

All documentation, architecture, and planning for Feature #6: Smart Document Processing & Compliance is complete!

## What's Been Created

### 1. Comprehensive Documentation

- **Feature Overview** ( `FEATURE_6_DOCUMENT_PROCESSING.md` ) - Business value, features, architecture
- **API Specification** ( `DOCUMENT_API_SPEC.md` ) - All endpoints documented
- **Implementation Checklist** ( `FEATURE_6_CHECKLIST.md` ) - Week-by-week tasks
- **Database Schema** - Complete data model in Prisma schema

### 2. Technical Design

- ✅ Database schema with Document and DocumentTemplate models
- ✅ API endpoint structure planned
- ✅ Component architecture defined
- ✅ Integration points (Cloudinary, OpenAI, OCR) specified


### 3. Code Structure Created

```
src/
├── app/api/documents/
│   ├── upload/
│   ├── search/
│   ├── compliance/
│   ├── templates/
│   └── generate/
├── components/documents/
├── lib/documents/
│   ├── cloudinary.ts      ✅ Created
│   ├── ocr.ts            ✅ Created
│   ├── extraction.ts     ✅ Created
│   ├── classification.ts  ✅ Created
│   ├── compliance.ts     ✅ Created
│   └── generation.ts     ✅ Created
└── types/documents/
    └── index.ts          ✅ Created
```

### 4. Permissions & RBAC

- ✅ Document permissions added to `src/lib/permissions.ts`
- ✅ Role mappings configured:
- **ADMIN**: Full document management
- **OPERATOR**: View, create, update, delete, extract, classify, generate
- **CAREGIVER**: View and create documents
- **FAMILY**: View and upload documents for their resident

## 5. Database Schema Updates

-  Prisma schema updated with:
- `Document` model with OCR and compliance tracking
- `DocumentTemplate` model for generation
- `DocumentType`, `ExtractionStatus`, `ComplianceStatus` enums
- Relations to User, Resident, and Inquiry models

## Ready to Start

---

### Phase 1A: Database & Storage (Days 1-3)

#### Tasks:

1. Run Prisma migration to create Document tables
2. Test Cloudinary upload configuration
3. Create upload API endpoint
4. Test file upload flow

**Estimated Time:** 2-3 days

### Phase 1B: Upload UI (Days 4-7)

#### Tasks:

1. Build DocumentUpload component
2. Implement drag-and-drop
3. Add file validation
4. Create document list view
5. Test end-to-end upload flow

**Estimated Time:** 3-4 days

## Dependencies to Install

---





Before starting implementation, install these packages:

```
npm install tesseract.js
npm install pdf-lib
npm install react-pdf
npm install @types/pdf-lib
npm install cloudinary
npm install @google-cloud/vision # Optional for fallback OCR
```

## Environment Variables

---

Already configured:

-  `CLOUDINARY_CLOUD_NAME=dygtsnu8z`
-  `CLOUDINARY_API_KEY=328392542172231`
-  `CLOUDINARY_API_SECRET` (existing)
-  `OPENAI_API_KEY` (existing)






Optional to add:

- `GOOGLE_CLOUD_VISION_API_KEY` (for fallback OCR)






## Success Criteria

---

### Technical Metrics

-  Upload success rate > 99%
-  OCR accuracy > 95%
-  Field extraction accuracy > 90%
-  Classification accuracy > 85%
-  Processing time < 30 seconds per document

### Business Metrics

-  Time saved: 5-10 hours/week per operator
-  Data entry errors reduced by 80%
-  Compliance rate increased to 100%
-  Onboarding time reduced by 50%
-  User satisfaction > 4.5/5

## Implementation Timeline

---

### Week 1: Foundation (Dec 19-25, 2025)

- Phase 1A: Database & Storage
- Phase 1B: Upload UI

### Week 2: Extraction (Dec 26-Jan 1, 2026)

- Phase 2: OCR Implementation
- Phase 3: AI Field Extraction

### Week 3: Classification & Compliance (Jan 2-8, 2026)

- Phase 4: Document Classification
- Phase 5: Compliance Tracking






### Week 4: Generation & Polish (Jan 9-15, 2026)

- Phase 6: Document Generation
- Phase 7: Testing & Deployment

**Total:** 4 weeks to completion

## Next Steps

---

1.  Review all documentation
2.  Run Prisma migration: `npx prisma migrate dev --name add_document_processing`
3.  Install dependencies
4.  Start Phase 1A: Database setup
5.  Implement upload functionality

## Files Created

---

### Documentation

- docs/features/FEATURE\_6\_DOCUMENT\_PROCESSING.md
- docs/features/DOCUMENT\_API\_SPEC.md
- docs/features/FEATURE\_6\_CHECKLIST.md
- docs/features/FEATURE\_6\_READY.md (this file)

### Database

- prisma/schema.prisma (updated with Document models)

### Code Structure

- src/types/documents/index.ts
- src/lib/documents/cloudinary.ts
- src/lib/documents/ocr.ts
- src/lib/documents/extraction.ts
- src/lib/documents/classification.ts
- src/lib/documents/compliance.ts
- src/lib/documents/generation.ts

### Permissions

- src/lib/permissions.ts (updated with document permissions)
- 

**Status:**  Planning Complete - Ready to Code!

**Date:** December 19, 2025

**Created By:** DeepAgent

 **All planning complete! Ready to start building Feature #6!**