Enterprise Document Signing API - Technical Specification

1. Project Overview

Tech Stack: Node.js, Express.js, Prisma ORM, PostgreSQL, Redis

Language: JavaScript (ES6+)

Purpose: Enterprise-grade RESTful API for electronic document signing with multi-party support, audit trails, and webhook notifications.

2. Core Features

2.1 Authentication & Authorization

- **API Key Authentication** - For developer access
- **OAuth 2.0** - Client credentials flow for enterprise integrations
- **JWT Tokens** - Session management with refresh tokens
- **Role-Based Access Control (RBAC)** - Admin, Developer, Signer roles
- **Rate Limiting** - Tiered based on subscription plans

2.2 Document Management

- **Upload Documents** - PDF, DOCX support (convert to PDF)
- **Document Storage** - S3-compatible storage with encryption at rest
- **Document Templates** - Reusable templates with merge fields
- **Version Control** - Track document revisions
- **Bulk Operations** - Upload and process multiple documents

2.3 Signature Workflows

- **Sequential Signing** - Ordered signer flow

- **Parallel Signing** - All signers receive simultaneously
- **Hybrid Workflows** - Mix sequential and parallel stages
- **Signature Types** - Click-to-sign, draw, type, upload image
- **Multi-party Support** - Up to 50 signers per document
- **Expiration Dates** - Auto-expire signing requests
- **Reminders** - Automated email/SMS reminders

2.4 Security & Compliance

- **AES-256 Encryption** - For stored documents
- **TLS 1.3** - For data in transit
- **Digital Signatures** - PKI-based cryptographic signatures
- **Audit Trails** - Immutable logs of all actions
- **IP Tracking** - Record signer IP addresses
- **Geolocation** - Optional GPS coordinates
- **Certificate of Completion** - Tamper-proof completion certificates
- **ESIGN & UETA Compliance** - US electronic signature laws
- **GDPR Ready** - Data privacy controls

2.5 Notifications & Webhooks

- **Email Notifications** - Customizable templates
- **SMS Notifications** - Optional for critical events
- **Webhook Events** - Real-time event delivery
- **Retry Logic** - Exponential backoff for failed webhooks
- **Event Types** - Document sent, viewed, signed, completed, expired, declined

3. Database Schema (Prisma Models)

Key Models

```
// User/Account Management
- Organization (multi-tenant support)
- User (developers/admins)
- ApiKey (API authentication)
- OAuthClient (OAuth credentials)
```

```
// Document Management
- Document (metadata, status, file reference)
- DocumentVersion (version history)
- Template (reusable templates)
- StorageFile (S3 references)

// Signing Workflow
- SigningRequest (tracks overall signing process)
- Signer (individual signer details)
- SignatureField (position and type)
- Signature (captured signature data)
- AuditLog (comprehensive audit trail)

// Notifications
- Webhook (webhook configurations)
- WebhookEvent (event delivery tracking)
- EmailLog (email delivery status)

// Billing (optional)
- Subscription (plan management)
- Usage (API usage tracking)
```

4. API Endpoints Structure

4.1 Authentication

```
POST   /api/v1/auth/register
POST   /api/v1/auth/login
POST   /api/v1/auth/refresh
POST   /api/v1/auth/logout
POST   /api/v1/auth/api-keys
GET    /api/v1/auth/api-keys
DELETE /api/v1/auth/api-keys/:id
```

4.2 Documents

```
POST   /api/v1/documents/upload
GET    /api/v1/documents
GET    /api/v1/documents/:id
DELETE /api/v1/documents/:id
GET    /api/v1/documents/:id/download
POST   /api/v1/documents/:id/versions
GET    /api/v1/documents/:id/audit-log
```

4.3 Templates

```
POST   /api/v1/templates
GET    /api/v1/templates
GET    /api/v1/templates/:id
PUT    /api/v1/templates/:id
DELETE /api/v1/templates/:id
POST   /api/v1/templates/:id/use
```

4.4 Signing Requests

```
POST   /api/v1/signing-requests
GET    /api/v1/signing-requests
GET    /api/v1/signing-requests/:id
PUT    /api/v1/signing-requests/:id
DELETE /api/v1/signing-requests/:id
POST   /api/v1/signing-requests/:id/send
POST   /api/v1/signing-requests/:id/remind
POST   /api/v1/signing-requests/:id/cancel
GET    /api/v1/signing-requests/:id/status
GET    /api/v1/signing-requests/:id/certificate
```

4.5 Signatures (Public Signing Interface)

```
GET    /api/v1/sign/:token (get signing page data)
POST   /api/v1/sign/:token/signature
```

```
POST   /api/v1/sign/:token/decline
GET    /api/v1/sign/:token/document
```

4.6 Webhooks

```
POST   /api/v1/webhooks
GET    /api/v1/webhooks
GET    /api/v1/webhooks/:id
PUT    /api/v1/webhooks/:id
DELETE /api/v1/webhooks/:id
GET    /api/v1/webhooks/:id/events
POST   /api/v1/webhooks/:id/test
```

4.7 Analytics & Reporting

```
GET    /api/v1/analytics/dashboard
GET    /api/v1/analytics/documents
GET    /api/v1/analytics/signers
GET    /api/v1/reports/usage
GET    /api/v1/reports/audit
```

5. Key Technical Implementation

5.1 Document Processing Pipeline

1. **Upload** → Validate file type and size
2. **Virus Scan** → ClamAV integration
3. **Convert** → Convert DOCX to PDF (using LibreOffice)
4. **Encrypt** → AES-256 encryption
5. **Store** → Upload to S3/MinIO
6. **Generate Thumbnail** → PDF preview generation
7. **Extract Metadata** → Page count, dimensions

5.2 Signature Capture Process

1. Signer receives email with unique token
2. Token validates identity and document access
3. Signer reviews document
4. Signature placement on designated fields
5. Cryptographic hash generation
6. Signature storage with timestamp
7. Certificate generation upon completion
8. Final document assembly with signatures

5.3 Security Implementation

- **Helmet.js** - Security headers
- **Express Rate Limit** - DDoS protection
- **CORS** - Configurable cross-origin policies
- **Input Validation** - Joi/Express-validator
- **SQL Injection Prevention** - Prisma ORM protection
- **XSS Protection** - Content sanitization
- **Secrets Management** - Environment variables + Vault

5.4 Background Job Processing

Using Bull Queue (Redis-based)

- Document conversion jobs
- Email sending queue
- Webhook delivery queue
- Reminder scheduling
- Document expiration cleanup
- Analytics aggregation

6. Project Structure

```
project-root/
├── src/
│   ├── config/           # Configuration files
```

```
│   ├── controllers/      # Route controllers
│   ├── middleware/       # Auth, validation, error handling
│   ├── services/         # Business logic
│   │   ├── auth.service.js
│   │   ├── document.service.js
│   │   ├── signature.service.js
│   │   ├── storage.service.js
│   │   ├── webhook.service.js
│   │   └── email.service.js
│   ├── routes/           # API routes
│   ├── utils/            # Helper functions
│   ├── validators/       # Request validators
│   ├── jobs/             # Background jobs
│   └── app.js            # Express app setup
├── prisma/
│   ├── schema.prisma     # Database schema
│   ├── migrations/       # Database migrations
│   └── seed.js           # Seed data
├── tests/                # Unit and integration tests
├── docs/                 # API documentation
├── .env.example
├── package.json
└── server.js             # Entry point
```

7. Required Dependencies

```
{
  "express": "^4.18.0",
  "@prisma/client": "^5.0.0",
  "prisma": "^5.0.0",
  "bcrypt": "^5.1.0",
  "jsonwebtoken": "^9.0.0",
  "multer": "^1.4.5",
  "aws-sdk": "^2.1400.0",
  "bull": "^4.11.0",
  "redis": "^4.6.0",
```

```
  "nodemailer": "^6.9.0",
  "pdf-lib": "^1.17.0",
  "sharp": "^0.32.0",
  "helmet": "^7.0.0",
  "cors": "^2.8.5",
  "express-rate-limit": "^6.8.0",
  "joi": "^17.9.0",
  "winston": "^3.10.0",
  "dotenv": "^16.3.0",
  "axios": "^1.4.0"
}
```

8. Environment Configuration

```
# Server
NODE_ENV=production
PORT=3000
API_VERSION=v1

# Database
DATABASE_URL="postgresql://user:password@localhost:5432/docsign"

# Redis
REDIS_HOST=localhost
REDIS_PORT=6379

# JWT
JWT_SECRET=your-secret-key
JWT_EXPIRES_IN=1h
REFRESH_TOKEN_EXPIRES_IN=7d

# Storage (S3/MinIO)
S3_ENDPOINT=https://s3.amazonaws.com
S3_BUCKET=documents
S3_ACCESS_KEY=your-access-key
S3_SECRET_KEY=your-secret-key
```

```
S3_REGION=us-east-1

# Email (SMTP)
SMTP_HOST=smtp.gmail.com
SMTP_PORT=587
SMTP_USER=your-email@domain.com
SMTP_PASS=your-password
EMAIL_FROM="DocSign API <noreply@docsign.com>"

# Encryption
ENCRYPTION_KEY=32-byte-hex-key

# Rate Limiting
RATE_LIMIT_WINDOW=15
RATE_LIMIT_MAX_REQUESTS=100

# URLs
API_BASE_URL=https://api.yourdomain.com
FRONTEND_URL=https://app.yourdomain.com
```

9. Security Best Practices

1. **Never store plain-text passwords** - Use bcrypt with salt rounds ≥12
2. **Rotate API keys regularly** - Implement key expiration
3. **Validate all inputs** - Prevent injection attacks
4. **Implement request signing** - HMAC-SHA256 for webhook verification
5. **Use prepared statements** - Prisma handles this automatically
6. **Enable audit logging** - Log all sensitive operations
7. **Implement CSRF protection** - For web interfaces
8. **Regular security audits** - Use npm audit and Snyk
9. **Principle of least privilege** - Minimal database permissions
10. **Secure file uploads** - Validate MIME types and scan for malware

10. Scalability Considerations

Horizontal Scaling

- **Stateless API** - No server-side sessions
- **Load Balancing** - NGINX/HAProxy
- **Database Connection Pooling** - Prisma connection pool
- **Redis Clustering** - For job queues
- **CDN Integration** - CloudFront for document delivery

Performance Optimization

- **Database Indexing** - Strategic indexes on queries
- **Caching Strategy** - Redis for frequently accessed data
- **Pagination** - Cursor-based for large datasets
- **Lazy Loading** - Load signatures only when needed
- **Query Optimization** - Use Prisma select and include wisely
- **Response Compression** - Gzip middleware

11. Monitoring & Logging

Application Monitoring

- **Winston Logger** - Structured logging with levels
- **Morgan** - HTTP request logging
- **Error Tracking** - Sentry integration
- **APM** - New Relic or DataDog
- **Health Checks** - `/health` and `/readiness` endpoints

Metrics to Track

- API response times
- Document processing durations
- Signature completion rates
- Webhook delivery success rates

- Storage usage
- Active signing requests
- Error rates by endpoint

12. Testing Strategy

Unit Tests

- Service layer functions
- Utility functions
- Validators

Integration Tests

- API endpoint testing
- Database operations
- External service mocks

E2E Tests

- Complete signing workflows
- Multi-signer scenarios
- Error handling paths

Testing Tools: Jest, Supertest, Prisma Test Environment

13. Deployment Checklist

- [ ] Set up production database (PostgreSQL)
- [ ] Configure Redis cluster
- [ ] Set up S3/MinIO storage
- [ ] Configure SMTP service
- [ ] Set up SSL certificates
- [ ] Configure environment variables

- [ ] Run database migrations
- [ ] Set up monitoring and alerting
- [ ] Configure backup strategy
- [ ] Set up CI/CD pipeline
- [ ] Load testing
- [ ] Security audit
- [ ] API documentation (Swagger/Postman)
- [ ] Rate limiting configuration
- [ ] CORS policy setup

14. Future Enhancements

- Mobile SDKs (iOS/Android)
- Biometric signature capture
- Video signing verification
- Blockchain anchoring for audit trails
- AI-powered document analysis
- Multi-language support
- Advanced workflow builder UI
- White-label solutions
- On-premise deployment option

15. Estimated Development Timeline

Phase 1 (4-6 weeks): Core API, authentication, document upload

Phase 2 (4-6 weeks): Signing workflows, signature capture

Phase 3 (3-4 weeks): Webhooks, notifications, audit trails

Phase 4 (2-3 weeks): Security hardening, testing

Phase 5 (2-3 weeks): Documentation, deployment, monitoring

Total: 15-22 weeks for MVP

This specification provides a complete roadmap for building an enterprise-grade document signing API. Each section should be expanded during implementation with specific technical details and business requirements.