AcademiaLens Development Tasks

This document provides a comprehensive, systematic breakdown of development tasks for implementing the AcademiaLens application. Each task is designed to be small, precise, and actionable with clear objectives.

1. Project Setup & Infrastructure

1.1 Next.js Project Initialization

1. Create Next.js 14+ project with TypeScript and App Router `bash npx create-next-app@latest academialens --typescript --tailwind --app --eslint`
2. Set up project directory structure following the file structure document
3. Configure ESLint and Prettier for code quality
4. Set up Git repository with proper .gitignore
5. Configure Husky for pre-commit hooks

1.2 Database Setup

1. Install PostgreSQL locally for development `bash docker run --name academialens-postgres -e POSTGRES_PASSWORD=password -p 5432:5432 -d postgres`
2. Install Prisma ORM and initialize `bash npm install prisma @prisma/client npx prisma init`
3. Create Prisma schema file with all models from database schema document:
4. User, Account, Session models
5. Project, ProjectMember models with relationships
6. Document, DocumentChunk, DocumentEntity models
7. Analysis, Citation, AIInteraction, Glossary models
8. Notification, SystemSetting models
9. Set up environment variables for database connection
10. Generate Prisma client and initial migration
11. Create database utility file for Prisma client singleton

1.3 Authentication System

1. Install NextAuth.js `bash npm install next-auth`
2. Configure NextAuth with Prisma adapter
3. Set up authentication providers (Google, Email/Password)

4. Create authentication API routes
5. Implement protected routes with middleware
6. Create authentication hooks and context
7. Build sign-in, sign-up, and password reset pages
8. Implement email verification system

1.4 Storage Integration

1. Create Cloudinary account and get API credentials
2. Install Cloudinary SDK `bash npm install cloudinary`
3. Create storage utility functions for file uploads
4. Implement secure URL generation for file access
5. Set up storage folder structure for different content types
6. Create file type validation and security checks
7. Implement file deletion and cleanup functionality

1.5 Development Environment

1. Configure Docker Compose for local development
2. Set up development, staging, and production environments
3. Create environment variable templates for each environment
4. Configure build and deployment scripts
5. Set up continuous integration workflow

2. Core Application Framework

2.1 UI Framework Setup

1. Install and configure Tailwind CSS `bash npm install -D tailwindcss postcss autoprefixer npx tailwindcss init -p`
2. Set up Shadcn UI components `bash npx shadcn-ui@latest init`
3. Create theme configuration with light/dark mode support
4. Build responsive layout components
5. Create global styles and typography system
6. Implement icon system with Lucide icons

2.2 State Management

1. Install Zustand for state management `bash npm install zustand`
2. Create store for user session state
3. Implement document management state

4. Create analysis and processing state
5. Build notification and alert state
6. Set up persistent state with local storage

2.3 API Layer

1. Create API client with Axios or SWR `bash npm install axios swr`
2. Implement API request hooks for data fetching
3. Create error handling and retry logic
4. Build response caching system
5. Implement request throttling and debouncing
6. Create API route handlers for all endpoints

2.4 Layout & Navigation

1. Build main application layout with sidebar
2. Create responsive navigation components
3. Implement breadcrumb navigation system
4. Build dashboard layout with widgets
5. Create document viewer layout
6. Implement analysis results layout

3. Universal Input & Ingestion Engine

3.1 Document Upload System

1. Create drag-and-drop file upload component
2. Implement multi-file upload support
3. Build upload progress indicator
4. Create file type validation (PDF, DOCX, TXT)
5. Implement file size limits and validation
6. Build error handling for failed uploads

3.2 PDF Processing

1. Install PDF.js and pdf-parse `bash npm install pdfjs-dist pdf-parse`
2. Create PDF rendering component
3. Implement text extraction from PDFs
4. Build PDF navigation with page controls
5. Create PDF annotation system
6. Implement PDF metadata extraction

3.3 OCR Processing

1. Install Tesseract.js for OCR `bash npm install tesseract.js`
2. Create OCR processing service
3. Implement image preprocessing for better OCR results
4. Build OCR progress indicator
5. Create OCR result validation and correction
6. Implement language detection for multi-language support

3.4 Web Content Extraction

1. Create server-side web scraping with Cheerio and Axios
2. Implement URL validation and security checks
3. Build content extraction rules for academic websites
4. Create metadata extraction from web pages
5. Implement citation information extraction
6. Build error handling for failed extractions

3.5 Document Processing Pipeline

1. Create document processing queue system
2. Implement document chunking algorithm
3. Build text cleaning and normalization
4. Create document metadata store
5. Implement document versioning system
6. Build processing status tracking

4. Insight Extractor Module

4.1 Gemini AI Integration

1. Set up Google AI client for Gemini API `bash npm install @google/generative-ai`
2. Create AI service wrapper for Gemini
3. Implement prompt template system
4. Build response parsing and formatting
5. Create error handling and retry logic
6. Implement token usage tracking and optimization

4.2 ELI-PhD Suite

1. Create multi-level explanation UI controls

2. Implement specialized prompts for different explanation levels
3. Build explanation rendering component
4. Create source citation system
5. Implement chain-of-thought visualization
6. Build explanation comparison view

4.3 Precision Summarizer

1. Create summary parameter controls (length, focus, style)
2. Implement summary generation service
3. Build abstract generation feature
4. Create TL;DR generator for quick scanning
5. Implement source tracking for verifiable summaries
6. Build summary export functionality

4.4 Entity & Keyword Identification

1. Create named entity recognition with spaCy or Gemini
2. Implement keyword extraction algorithm
3. Build entity linking and disambiguation
4. Create entity visualization component
5. Implement entity filtering and search
6. Build entity relationship mapping

4.5 JargonBuster & Acronym Resolver

1. Create jargon detection algorithm
2. Implement acronym extraction and expansion
3. Build domain-specific terminology database
4. Create tooltip system for term explanations
5. Implement custom glossary builder
6. Build terminology export functionality

5. Interactive Knowledge Assistant

5.1 Contextual Q&A System

1. Create question input interface
2. Implement RAG (Retrieval-Augmented Generation) with LangChain `bash npm install langchain`
3. Build document context retrieval system

4. Create answer generation with source citations
5. Implement conversation history management
6. Build answer quality feedback system

5.2 Custom Glossary Builder

1. Create glossary management interface
2. Implement term extraction and definition generation
3. Build glossary editing tools
4. Create glossary export functionality
5. Implement glossary sharing between projects
6. Build glossary search and filtering

5.3 Conversation Management

1. Create conversation thread interface
2. Implement conversation persistence in database
3. Build conversation export functionality
4. Create conversation search and filtering
5. Implement conversation sharing
6. Build conversation analytics

6. Deconstruction Toolkit Module

6.1 Methodology Blueprint

1. Create methodology extraction prompts
2. Implement flowchart generation with Mermaid.js `bash npm install mermaid`
3. Build interactive methodology viewer
4. Create methodology comparison tool
5. Implement methodology export functionality
6. Build methodology template library

6.2 Claims & Evidence Mapper

1. Create claim detection algorithm
2. Implement evidence linking system
3. Build claim-evidence visualization
4. Create counter-argument detection
5. Implement claim strength evaluation
6. Build claim export functionality

6.3 Reproducibility Auditor

1. Create reproducibility checklist generator
2. Implement methodology completeness evaluation
3. Build data availability checker
4. Create code and resource locator
5. Implement reproducibility score calculator
6. Build reproducibility report generator

6.4 Quick Reference Card Generator

1. Create reference card template system
2. Implement key point extraction
3. Build visual card designer
4. Create card export functionality
5. Implement card sharing and collaboration
6. Build card collection management

7. Synthesis & Connection Hub Module

7.1 Cross-Document Weaver

1. Create document selection interface
2. Implement cross-document analysis with Gemini
3. Build theme extraction across documents
4. Create connection visualization
5. Implement connection strength evaluation
6. Build cross-document report generator

7.2 Consensus & Conflict Finder

1. Create claim extraction across documents
2. Implement consensus detection algorithm
3. Build conflict identification system
4. Create consensus/conflict visualization
5. Implement evidence strength evaluation
6. Build consensus report generator

7.3 Knowledge Gap Identifier

1. Create research question input interface

2. Implement gap analysis algorithm
3. Build gap visualization component
4. Create research recommendation generator
5. Implement literature search suggestions
6. Build gap analysis report

7.4 Comparative Analyzer

1. Create document comparison interface
2. Implement side-by-side comparison view
3. Build difference highlighting system
4. Create methodology comparison tool
5. Implement results comparison visualization
6. Build comparison report generator

8. Application & Foresight Engine Module

8.1 Practical Application Brainstormer

1. Create application domain selection interface
2. Implement application idea generation with Gemini
3. Build idea evaluation and ranking system
4. Create implementation roadmap generator
5. Implement resource requirement estimator
6. Build application report generator

8.2 SWOT Analysis Generator

1. Create SWOT analysis interface
2. Implement SWOT extraction from documents
3. Build SWOT visualization component
4. Create SWOT comparison across documents
5. Implement SWOT report generator
6. Build SWOT template library

8.3 Ethical & Societal Impact Scanner

1. Create ethical consideration extraction
2. Implement impact evaluation framework
3. Build stakeholder identification system
4. Create risk assessment generator

5. Implement mitigation strategy suggestions
6. Build ethics report generator

8.4 Innovation & IP Assistant

1. Create novelty assessment algorithm
2. Implement patent search integration
3. Build innovation scoring system
4. Create IP protection recommendation generator
5. Implement competitive analysis tool
6. Build innovation report generator

8.5 Future Trajectory Spotter

1. Create trend identification algorithm
2. Implement future scenario generator
3. Build trajectory visualization component
4. Create research direction recommender
5. Implement funding opportunity matcher
6. Build future outlook report generator

9. Collaboration & Sharing

9.1 Project Management

1. Create project creation interface
2. Implement project settings and configuration
3. Build document organization within projects
4. Create project dashboard with metrics
5. Implement project archiving and deletion
6. Build project template system

9.2 User Collaboration

1. Create user invitation system
2. Implement role-based permissions
3. Build real-time collaboration with WebSockets
4. Create activity tracking and history
5. Implement comment and feedback system
6. Build notification system for collaboration events

9.3 Export & Publishing

1. Create export format selection interface
2. Implement PDF export with custom styling
3. Build Word document export functionality
4. Create presentation slide export
5. Implement web publishing with sharing links
6. Build citation export in multiple formats

9.4 Notification System

1. Create notification center interface
2. Implement in-app notification storage
3. Build email notification system
4. Create notification preferences management
5. Implement real-time notification delivery
6. Build notification read/unread tracking

10. Subscription & Billing

10.1 Subscription Management

1. Create subscription plans in database
2. Implement plan selection interface
3. Build subscription management dashboard
4. Create usage tracking system
5. Implement plan upgrade/downgrade functionality
6. Build subscription expiration handling

10.2 Payment Processing

1. Set up Stripe integration `bash npm install stripe`
2. Create secure payment form
3. Implement subscription creation and management
4. Build invoice generation and delivery
5. Create payment history tracking
6. Implement failed payment handling

10.3 Usage Limits & Quotas

1. Create usage tracking for API calls

2. Implement document storage quota system
3. Build feature access control based on plan
4. Create usage dashboard for users
5. Implement quota warning notifications
6. Build grace period handling for overages

11. Testing & Quality Assurance

11.1 Unit Testing

1. Set up Jest testing framework `bash npm install -D jest @testing-library/react @testing-library/jest-dom`
2. Create test utilities and mocks
3. Implement component tests
4. Build service and utility function tests
5. Create API route tests
6. Implement database model tests

11.2 Integration Testing

1. Set up Cypress for end-to-end testing `bash npm install -D cypress`
2. Create test fixtures and data
3. Implement user flow tests
4. Build API integration tests
5. Create authentication flow tests
6. Implement subscription and payment tests

11.3 Performance Testing

1. Set up performance monitoring tools
2. Create load testing scripts
3. Implement response time benchmarks
4. Build memory usage monitoring
5. Create database query optimization tests
6. Implement API response time tests

12. Deployment & DevOps

12.1 Production Environment

1. Set up production database
2. Configure production environment variables
3. Implement database backup system
4. Create logging and monitoring
5. Set up error tracking with Sentry
6. Implement performance monitoring

12.2 CI/CD Pipeline

1. Create GitHub Actions workflow
2. Implement automated testing in pipeline
3. Build deployment automation
4. Create staging environment deployment
5. Implement production deployment with approval
6. Build rollback mechanism

12.3 Security

1. Implement HTTPS and security headers
2. Create rate limiting for API endpoints
3. Build input validation and sanitization
4. Create security audit logging
5. Implement data encryption at rest
6. Build regular security scanning

13. Documentation

13.1 Code Documentation

1. Create JSDoc comments for all functions
2. Implement TypeScript type definitions
3. Build API documentation with Swagger
4. Create architecture documentation
5. Implement changelog maintenance
6. Build developer onboarding guide

13.2 User Documentation

1. Create user guide with screenshots
2. Implement in-app help system
3. Build video tutorials
4. Create FAQ section
5. Implement feature documentation
6. Build troubleshooting guide

14. Maintenance & Support

14.1 Monitoring

1. Set up application health monitoring
2. Implement database performance monitoring
3. Build API usage tracking
4. Create error rate monitoring
5. Implement user activity analytics
6. Build automated alerts for issues

14.2 Support System

1. Create support ticket system
2. Implement user feedback collection
3. Build knowledge base for common issues
4. Create admin dashboard for support
5. Implement user communication tools
6. Build feature request tracking