tydids

Turn documents into intelligence
Tydids - Because your documents have stories to tell

A complete Qdrant-compatible vector database with AI-powered document processing and search capabilities. Features a modern web interface, multi-user support, and comprehensive admin tools.

🎨 Branding

- **Primary color:** `#147a50`
- **Secondary color:** `#e6b41e`
- **Accent color 1:** `#28aa6e`
- **Accent color 2:** `#ee7f4b`

✅ Complete Implementation Status

Frontend + Backend: All use cases fully implemented and tested

🚀 Features

Core Functionality

- **Multi-User Support**: Secure authentication with role-based access control
- **Document Processing**: Upload PDF, Word, text, and image files with automatic content extraction
- **Vector Search**: Semantic similarity search powered by Qdrant vector database
- **AI Chat**: RAG-based question answering with source attribution
- **Collection Management**: Organize documents into collections with detailed statistics
- **Admin Tools**: Complete user management and system monitoring

User Interface

- **Modern Design**: Responsive Bootstrap-based interface
- **Real-time Updates**: Live progress tracking and notifications
- **Drag & Drop**: Intuitive file upload with progress indicators
- **Advanced Search**: Collection-specific and global search with similarity controls

- **AI Chat Interface**: Interactive Q&A with document source tracking

Technical Features

- **RESTful API**: Complete OpenAPI 3.0 specification
- **JWT Authentication**: Secure token-based authentication
- **Rate Limiting**: Configurable limits for different operation types
- **Usage Tracking**: Tier-based limits with detailed analytics
- **File Storage**: Secure UUID-based file management
- **Vector Operations**: Optimized similarity search and document clustering

📋 Prerequisites

- Node.js 18+ and npm
- Docker and Docker Compose
- Qdrant vector database
- OpenAI API key (for embeddings and chat)

🛠️ Installation

1. **Clone the repository**

```
git clone https://github.com/your-repo/tydids.git
cd tydids
```

2. **Install dependencies**

```
npm install
```

3. **Setup environment**

```
cp .env.example .env
# Edit .env with your configuration
```

4. **Start services**

```
# Start Qdrant and other services
docker-compose up -d

# Initialize database
npm run db:migrate

# Start the application
npm start
```

5. **Access the application**

- Frontend: http://localhost:3001 (http://localhost:3001)
- API Documentation: http://localhost:3001/api-docs (http://localhost:3001/api-docs)
- Health Check: http://localhost:3001/api/health (http://localhost:3001/api/health)

🎯 Quick Start

Default Admin Account

- **Username**: `stromdao` (from .env)
- **Password**: `Maus12Rad` (from .env)

First Steps

1. Login with the default admin account
2. Create your first collection
3. Upload documents or create text content
4. Start searching and asking questions about your documents

📖 API Documentation

Complete API documentation is available at `/api-docs` when the server is running, or view the OpenAPI specification in `public/openapi.json` .

Key Endpoints

Authentication

- `POST /api/auth/register` - Register new user
- `POST /api/auth/login` - Login and get JWT token

Collections

- `GET /api/collections` - List user collections
- `POST /api/collections` - Create new collection
- `GET /api/collections/{id}` - Get collection details
- `DELETE /api/collections/{id}` - Delete collection

Documents

- `POST /api/upload/{collection}` - Upload files
- `GET /api/collections/{id}/documents` - List documents
- `POST /api/collections/{id}/search` - Search documents
- `POST /api/collections/{id}/ask` - Ask AI questions

Admin

- `GET /api/admin/dashboard` - Admin statistics
- `GET /api/admin/users` - Manage users
- `GET /api/admin/system/health` - System health

🏗️ Architecture

Backend Components

- **Express.js API**: RESTful API with comprehensive error handling
- **SQLite Database**: User data, collections, and document metadata
- **Qdrant Vector DB**: Document embeddings and similarity search
- **OpenAI Integration**: Text embeddings and chat completions
- **File Processing**: Multi-format document parsing and chunking

Frontend Components

- **Vanilla JavaScript SPA**: Modern ES6+ with modular architecture
- **Bootstrap UI**: Responsive design with custom styling
- **Real-time Features**: Progress tracking and live updates
- **State Management**: Efficient client-side state handling

Security Features

- **JWT Authentication**: Secure token-based auth with refresh
- **Rate Limiting**: Per-user and per-endpoint limits
- **Input Validation**: Comprehensive request/response validation
- **User Isolation**: Data segregation between users

- **CORS Configuration**: Secure cross-origin resource sharing

🔧 Configuration

Environment Variables

```
# Server Configuration
PORT=3001
NODE_ENV=development

# Database
DATABASE_URL=./tydids.db

# Qdrant Vector Database
QDRANT_URL=http://localhost:6333

# OpenAI API
OPENAI_API_KEY=your_openai_api_key
OPENAI_MODEL=gpt-4
EMBEDDING_MODEL=text-embedding-3-small

# Authentication
JWT_SECRET=your_jwt_secret_key
JWT_EXPIRES_IN=24h

# Default Admin User
ADMIN_USERNAME=stromdao
ADMIN_PASSWORD=Maus12Rad

# Rate Limiting
RATE_LIMIT_AUTH=5
RATE_LIMIT_SEARCH=20
RATE_LIMIT_GENERAL=100

# File Upload
MAX_FILE_SIZE=50MB
UPLOAD_PATH=./uploads
```

Usage Tiers

- **Free**: 5 collections, 100 documents, 50 searches/hour
- **Pro**: 50 collections, 10,000 documents, 500 searches/hour
- **Unlimited**: No limits

🚀 Deployment

Docker Deployment

```
# Build and run with Docker Compose
docker-compose -f docker-compose.prod.yml up -d
```

Manual Deployment

```
# Build for production
npm run build

# Start production server
npm run start:prod
```

Environment Setup

- Ensure Qdrant is accessible
- Configure OpenAI API key
- Set up proper JWT secrets
- Configure file storage paths

🧪 Testing

```
# Run API tests
npm test

# Run integration tests
npm run test:integration

# Test specific endpoints
npm run test:auth
npm run test:collections
npm run test:search
```

🤝 Contributing

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests for new functionality
5. Submit a pull request

📄 License

🆘 Support

- **Documentation**: Complete API docs at `/api-docs`
- **Issues**: GitHub Issues for bug reports
- **Discussions**: GitHub Discussions for questions

🎉 Acknowledgments

- **Qdrant**: Vector database for semantic search
- **OpenAI**: Embeddings and language model API
- **Bootstrap**: UI framework and components
- **Express.js**: Web application framework