Deep Research Agent - Quick Start Guide

🚀 Get Started in 10 Minutes

Follow this guide to set up and run the Deep Research Agent locally for development.

📋 Prerequisites

Before starting, ensure you have:

- **Node.js 18+** and npm/yarn installed
- **PostgreSQL 14+** installed and running
- **Redis 6+** installed and running
- **OpenAI API key** (required for AI features)

⚡ Quick Setup

Step 1: Environment Setup

```
# Navigate to project directory
cd deep_research_agent_project

# Copy environment configuration
cp .env.example .env.local

# Edit environment file with your settings
nano .env.local
```

Step 2: Database Setup

```
# Connect to PostgreSQL
sudo -u postgres psql

# Create database and user
CREATE DATABASE deep_research_agent;
CREATE USER research_user WITH PASSWORD 'research_password';
GRANT ALL PRIVILEGES ON DATABASE deep_research_agent TO research_user;
\q
```

Step 3: Essential API Keys

Add these to your `.env.local` file:

```
# Required - Get from OpenAI
OPENAI_API_KEY=sk-your-openai-api-key-here

# Database connection
DATABASE_URL=postgresql://research_user:research_password@localhost:5432/
deep_research_agent

# JWT secret (generate a secure one)
JWT_SECRET=your-super-secure-jwt-secret-here
```

Step 4: Start Services

```
# Start PostgreSQL (if not running)
sudo systemctl start postgresql

# Start Redis (if not running)
sudo systemctl start redis-server

# Verify services are running
redis-cli ping  # Should return PONG
psql -d deep_research_agent -U research_user -c "SELECT 1;"  # Should return 1
```

🎯 Development Workflow

Phase 1: Foundation (Week 1-2)

Start by building the core infrastructure:

1. **Project Setup**
   - Initialize NextJS with TypeScript
   - Configure database connections
   - Set up authentication system

2. **Basic UI Components**
   - User registration and login
   - Dashboard layout
   - Project management interface

Phase 2: Patent Search (Week 3-4)

Add patent search capabilities:

1. **API Integration**
   - Google Patents API connection
   - USPTO API integration
   - Search query optimization

2. **Search Interface**
   - Research scope definition form
   - Search results display
   - Basic filtering options

🔧 Configuration Options

Development Mode

For local development with relaxed settings:

```
NODE_ENV=development
LOG_LEVEL=debug
ENABLE_CORS=true
RATE_LIMIT_MAX_REQUESTS=1000
```

API Configuration

Essential API settings:

```
# Patent APIs
GOOGLE_PATENTS_API_URL=https://patents.googleapis.com/v1/patents
USPTO_API_KEY=DEMO_KEY  # Use DEMO_KEY for testing

# AI Processing
OPENAI_MODEL=gpt-4
OPENAI_MAX_TOKENS=2000
```

Feature Flags

Control which features are enabled:

```
ENABLE_PATENT_UPLOAD=true
ENABLE_BATCH_PROCESSING=true
ENABLE_REAL_TIME_UPDATES=true
ENABLE_EMAIL_NOTIFICATIONS=false  # Disable for development
```

📊 Development Tools

Database Tools

```
# View database schema
psql -d deep_research_agent -U research_user -c "\dt"

# Check database connections
psql -d deep_research_agent -U research_user -c "SELECT count(*) FROM pg_stat_activity;"
```

Redis Tools

```
# Check Redis status
redis-cli info server

# Monitor Redis activity
redis-cli monitor
```

API Testing

```bash
# Test Google Patents API
curl "https://patents.googleapis.com/v1/patents:search" \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"q": "machine learning"}'

# Test USPTO API
curl "https://developer.uspto.gov/api/v1/patent/application?searchText=artificial+intelligence"
```

🎯 Key Features to Implement First

MVP Features (Priority 1)

1. **User Authentication**
   - Registration and login
   - JWT token management
   - Basic user profiles

2. **Project Management**
   - Create research projects
   - Define research scopes
   - Save project settings

3. **Basic Patent Search**
   - Simple keyword search
   - Display search results
   - Basic patent information

Enhanced Features (Priority 2)

1. **Advanced Search**
   - Date range filtering
   - Company-specific searches
   - Technology classification

2. **Batch Processing**
   - Background job queues
   - Progress tracking
   - Email notifications

3. **Data Visualization**
   - Patent timeline charts
   - Assignee distribution
   - Basic analytics

🚨 Common Issues & Solutions

Database Connection Issues

```
# Check if PostgreSQL is running
sudo systemctl status postgresql

# Restart if needed
sudo systemctl restart postgresql

# Test connection
psql -h localhost -U research_user -d deep_research_agent
```

Redis Connection Issues

```
# Check Redis status
sudo systemctl status redis-server

# Test Redis connection
redis-cli ping
```

API Rate Limiting

- Google Patents: 100 requests per 100 seconds
- USPTO: 1000 requests per hour
- OpenAI: Based on your plan (3 RPM for free tier)

Monitor your usage to avoid hitting limits during development.

OpenAI API Issues

```
# Test OpenAI API connection
curl https://api.openai.com/v1/models \
  -H "Authorization: Bearer $OPENAI_API_KEY"
```

📈 Success Metrics

Track these metrics during development:

Technical Metrics

- API response times under 2 seconds
- Database queries under 100ms
- Background jobs complete within 5 minutes
- 99%+ uptime during development

Feature Metrics

- User can create account in under 1 minute
- Patent search returns results in under 5 seconds
- Research reports generate in under 10 minutes
- UI is responsive on mobile devices

🔄 Development Iterations

Week 1: Foundation

- [ ] NextJS project setup
- [ ] Database schema creation
- [ ] Basic authentication
- [ ] Simple UI layout

Week 2: Core Features

- [ ] Project management
- [ ] Basic patent search
- [ ] Search results display
- [ ] User dashboard

Week 3-4: Enhanced Search

- [ ] Google Patents API integration
- [ ] USPTO API integration
- [ ] Advanced search filters
- [ ] Search optimization

🎯 Next Steps

After completing the quick start:

1. **Review the full documentation**
2. **Set up your development environment**
3. **Start with Phase 1 implementation**
4. **Test each feature as you build**
5. **Deploy to staging environment**

📞 Getting Help

If you encounter issues:

1. **Check the logs** in `./logs/development.log`
2. **Review API documentation** for external services
3. **Test individual components** using provided test endpoints
4. **Monitor database and Redis** for connection issues

This quick start guide gets you up and running with the Deep Research Agent development environment in minutes, allowing you to focus on building the core patent research features.