API Usage Examples

Authentication

Register a new user

```
curl -X POST http://localhost:5000/api/auth/register \
  -H "Content-Type: application/json" \
  -d '{
    "email": "inventor@example.com",
    "password": "securepassword123",
    "firstName": "John",
    "lastName": "Inventor",
    "personaType": "inventor",
    "organization": "Innovation Labs"
  }'
```

Login

```
curl -X POST http://localhost:5000/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{
    "email": "inventor@example.com",
    "password": "securepassword123"
  }'
```

Get user profile

```
curl -X GET http://localhost:5000/api/auth/profile \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Projects

Create a new project

```
curl -X POST http://localhost:5000/api/projects \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "name": "AI Patent Research",
    "description": "Research project for AI-related patents",
    "projectType": "research",
    "tags": ["AI", "machine learning", "patents"]
  }'
```

Get all projects

```
curl -X GET http://localhost:5000/api/projects \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Get project details

```
curl -X GET http://localhost:5000/api/projects/PROJECT_ID \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Patent Search

Basic patent search

```
curl -X POST http://localhost:5000/api/search/patents \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "query": "machine learning patent analysis",
    "filters": {
      "dateRange": {
        "start": "2020-01-01",
        "end": "2023-12-31"
      },
      "patentStatus": "granted",
      "country": "US"
    },
    "limit": 20
  }'
```

Semantic search

```
curl -X POST http://localhost:5000/api/search/semantic \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "query": "automated system for finding similar patents using AI",
    "threshold": 0.7,
    "limit": 10
  }'
```

Advanced search with Boolean operators

```
curl -X POST http://localhost:5000/api/search/advanced \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "query": "(artificial intelligence OR machine learning) AND (patent OR intellectual
property)",
    "filters": {
      "classifications": ["G06N", "G06F"],
      "assignees": ["Google", "Microsoft", "IBM"],
      "inventors": ["John Smith"]
    },
    "sortBy": "relevance",
    "limit": 50
  }'
```

Patent Analysis

Create a new analysis

```
curl -X POST http://localhost:5000/api/analyses \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "projectId": "PROJECT_ID",
    "patentNumber": "US10123456B2",
    "analysisType": "comprehensive",
    "inputData": {
      "title": "Machine Learning System for Patent Analysis",
      "abstract": "A system and method for analyzing patent documents...",
      "claims": ["Claim 1 text", "Claim 2 text"]
    }
  }'
```

Get analysis results

```
curl -X GET http://localhost:5000/api/analyses/ANALYSIS_ID \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Run specific agent analysis

```
curl -X POST http://localhost:5000/api/agents/analyze \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "agentType": "prior_art",
    "patentData": {
      "title": "Machine Learning System for Patent Analysis",
      "abstract": "A system and method for analyzing patent documents...",
      "claims": ["Claim 1 text", "Claim 2 text"]
    },
    "configuration": {
      "searchDepth": "comprehensive",
      "includeNonPatentLiterature": true
    }
  }'
```

Batch Processing

Submit a batch job

```
curl -X POST http://localhost:5000/api/batch \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN" \
  -d '{
    "jobName": "Q4 Patent Portfolio Analysis",
    "jobType": "portfolio_analysis",
    "analysisType": "comprehensive",
    "inputData": [
      {
        "patentNumber": "US10123456B2",
        "title": "Patent 1 Title"
      },
      {
        "patentNumber": "US10234567B1",
        "title": "Patent 2 Title"
      }
    ],
    "configuration": {
      "priority": 1,
      "notifyOnCompletion": true
    }
  }'
```

Get batch job status

```
curl -X GET http://localhost:5000/api/batch/BATCH_JOB_ID \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Get all batch jobs

```
curl -X GET http://localhost:5000/api/batch \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Dashboard Data

Get inventor dashboard data

```
curl -X GET http://localhost:5000/api/dashboard/inventor \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Get corporate R&D dashboard data

```
curl -X GET http://localhost:5000/api/dashboard/corporate-rd \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

WebSocket Events

Connect to real-time updates

```javascript
import io from 'socket.io-client';

const socket = io('http://localhost:5000');

// Join a batch job room for real-time updates
socket.emit('join-batch-room', 'BATCH_JOB_ID');

// Listen for batch progress updates
socket.on('batch-progress', (data) => {
  console.log('Batch progress:', data);
  // { jobId: 'BATCH_JOB_ID', progress: 45, status: 'processing' }
});

// Listen for analysis completion
socket.on('analysis-complete', (data) => {
  console.log('Analysis completed:', data);
  // { analysisId: 'ANALYSIS_ID', status: 'completed', results: {...} }
});
```

Error Handling

All API endpoints return consistent error responses:

```json
{
  "error": "Error message description",
  "code": "ERROR_CODE",
  "details": {
    "field": "Additional error details"
  }
}
```

Common HTTP status codes:
- `200` - Success
- `201` - Created
- `400` - Bad Request (validation errors)
- `401` - Unauthorized (missing or invalid token)
- `403` - Forbidden (insufficient permissions)
- `404` - Not Found
- `429` - Too Many Requests (rate limited)
- `500` - Internal Server Error

Rate Limiting

The API implements rate limiting:
- 100 requests per 15-minute window per IP address
- Higher limits for authenticated users based on subscription tier
- Batch operations have separate limits

Rate limit headers are included in responses:

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 95
X-RateLimit-Reset: 1634567890
```

Pagination

List endpoints support pagination:

```
curl -X GET "http://localhost:5000/api/projects?
page=2&limit=10&sortBy=createdAt&sortOrder=desc" \
  -H "Authorization: Bearer YOUR_JWT_TOKEN"
```

Response includes pagination metadata:

```
{
  "data": [...],
  "pagination": {
    "page": 2,
    "limit": 10,
    "total": 150,
    "totalPages": 15,
    "hasNext": true,
    "hasPrev": true
  }
}
```