📚 UPSC AI - News & Notes Platform

A comprehensive AI-powered platform for UPSC aspirants to scrape current affairs news, generate study notes, create MCQs, and build daily digests using Google Gemini API.

🎯 Overview

UPSC AI is a full-stack application designed to help Civil Services examination aspirants efficiently manage current affairs for their preparation. The platform leverages Google Gemini AI to automatically analyze news articles and generate comprehensive study materials including notes, multiple-choice questions, and main answer frameworks.

Key Features

- 🔍 **Live News Scraping** - Fetch latest news from 9+ major sources
- 🤖 **AI-Powered Analysis** - Automatic note generation using Google Gemini
- 📝 **Study Materials** - MCQs, mains questions, and detailed notes
- 💾 **Secure Database** - MongoDB integration for persistent storage
- 🔐 **User Authentication** - Secure registration and login system
- 📊 **Daily Digest** - AI-generated comprehensive current affairs digest
- 🏷️ **Smart Organization** - Categorize notes by GS papers (GS1-GS4, Prelims, Essay)
- 🔖 **Bookmark System** - Save articles for later analysis
- 🔄 **Fallback Storage** - localStorage backup when database unavailable

---

🚀 Quick Start

Prerequisites

- Node.js v18+ and npm
- MongoDB Atlas account (for cloud database)
- Google Gemini API key
- Git

Installation

1. **Clone the repository**

   ```
   git clone https://github.com/supernova0311/upsc-digest.git
   cd civil-services-ai---upsc-news-&-notes
   ```

2. **Install dependencies**

   ```
   npm install
   ```

3. **Configure environment variables**

   Create a `.env.local` file in the root directory:

   ```
   GEMINI_API_KEY=your_gemini_api_key_here
   MONGODB_URI=mongodb+srv://username:password@cluster.mongodb.net/upsc-ai
   JWT_SECRET=your_secure_jwt_secret_key_here
   ```

   How to get these credentials:

   - **GEMINI_API_KEY**: Visit Google AI Studio (https://aistudio.google.com/apikey)
   - **MONGODB_URI**: Create account at MongoDB Atlas (https://www.mongodb.com/cloud/atlas)
   - **JWT_SECRET**: Any strong random string (min 32 characters recommended)

4. **Start the development servers**

   Open two terminal windows:

   Terminal 1 - Backend Server (port 5000):

   ```
   node server.js
   ```

   Expected output:

   ```
   🚀 Server running on port 5000
   ✅ Connected to MongoDB
   ```

   Terminal 2 - Frontend Dev Server (port 3000):

   ```
   npm run dev
   ```

   Expected output:

   ```
   ➜  Local:   http://localhost:3000/
   ```

5. **Access the application**

   Open your browser and navigate to: `http://localhost:3000`

📖 User Manual

1. Authentication

Registration

1. Open the application at `http://localhost:3000`
2. Click **"Create a new account"**
3. Enter your details:
    - **Full Name**: Your name
    - **Email**: Valid email address
    - **Password**: Strong password (min 8 characters recommended)
4. Click **Register**
5. You'll be logged in automatically and redirected to the Dashboard

Login

1. If already registered, click **"Sign in instead"**
2. Enter your **Email** and **Password**
3. Click **Sign in**
4. Access the dashboard with your credentials

Mock Login (Development Mode)

If the backend is unavailable, the app creates a mock account automatically for testing purposes.

---

2. Dashboard

The main hub displaying your UPSC preparation statistics and daily digest.

Components:

📊 Statistics Panel

- Total notes saved
- Notes by GS paper (GS1, GS2, GS3, GS4)
- Quick overview of your study progress

📰 Recent Articles

- Latest 10 scraped articles
- Source and publication date
- Quick view of trending current affairs topics

📋 Daily Digest

- AI-generated comprehensive summary
- Organized by sections:
    - Top Stories
    - Economic Updates
    - Environment
    - Prelims Pointers
- Click **Generate Digest** to refresh

Navigation

- **Live Scraper**: Fetch news and generate notes
- **Saved Notes**: View and manage all bookmarked notes
- **Logout**: Sign out from the application

---

3. Live News Scraper

The core feature for discovering and analyzing current affairs.

Workflow:

Step 1: Select News Source

Choose from 9 major news sources:

- **The Hindu** - National news, policy, and supreme court judgments
- **The Hindu (Editorials)** - Editorials, opinion pieces, and analysis
- **Indian Express** - Explained series, economy, and politics
- **PIB** - Government schemes, cabinet decisions, press releases
- **Livemint** - Indian economy, banking, and finance
- **Economic Times** - Macroeconomy, RBI, and fiscal policy
- **Business Standard** - Business, infrastructure, and policy
- **Sansad TV** - Parliamentary debates, bills passed, committees
- **Down To Earth** - Environment, ecology, climate change, agriculture

Step 2: Fetch News

1. Click **"Fetch [Source] News"** button
2. Status updates show:
    - 🔍 Searching... (news retrieval in progress)
    - ✅ Success (articles fetched)
    - ❌ Error (if fetch fails)

3. Articles appear in the left panel

Step 3: Analyze Article

1. From the article list, click **"Generate Notes"**
2. The AI analyzes the article and displays:
    - **Title**: Auto-generated note title
    - **GS Paper**: Classification (GS1-GS4, Prelims, Essay)
    - **Tags**: Relevant keywords
    - **Summary**: Quick overview
    - **Detailed Notes**: Comprehensive markdown formatted content
    - **MCQs**: 2-3 practice questions with explanations
    - **Mains Question**: Model answer framework

Step 4: Save Note

1. Review the generated note
2. Click **"Save to Database"**
3. Confirmation message appears
4. Note is saved to MongoDB (or localStorage as fallback)

Note: Articles are automatically saved to your history. Click the **Refresh** button to update the article feed.

---

4. Saved Notes

View, search, and manage all your bookmarked notes.

Features:

Search & Filter

- Search by title or summary keywords
- Filter by GS paper (All, GS1, GS2, GS3, GS4, Prelims, Essay)
- See matching notes count

Note Management

- **View**: Click any note to see full details on the right
- **Delete**: Click trash icon to remove unwanted notes
- **Sort**: Notes automatically sorted by date (newest first)

Note Viewer

- Full markdown rendering
- MCQs with explanations
- Mains question with model answer points
- Mobile-friendly responsive design

---

5. AI Features

Note Generation

The AI automatically analyzes article content and produces:

1. **Title** - Concise, searchable title
2. **GS Classification** - Categorizes into appropriate GS paper
3. **Tags** - 3-5 relevant keywords for quick identification
4. **Summary** - 100-150 word quick reference
5. **Detailed Notes** - Structured markdown with:
    - Key concepts explained
    - Real-world applications
    - Important facts and figures
    - Connections to other topics
6. **MCQs** - 2-3 competitive exam style questions with:
    - 4 multiple choice options
    - Correct answer indicated
    - Detailed explanations
7. **Mains Question** - Main exam answer framework with:
    - Relevant question format
    - Model answer key points (bullet format)

Daily Digest Generation

1. Collects 15 most recent articles
2. Analyzes across all sources
3. Generates organized digest with:
    - **Top Stories**: Major news affecting governance
    - **Economic Updates**: Banking, fiscal, monetary policy
    - **Environment**: Climate, ecology, conservation
    - **Prelims Pointers**: Quick facts and definitions

🏗️ Project Architecture

Technology Stack

Frontend

- React 19.2 - UI framework
- TypeScript - Type-safe development
- Vite 6.2 - Lightning-fast build tool
- React Router 7.10 - Client-side routing
- Lucide React - Icon library
- React Markdown - Markdown rendering
- Tailwind CSS - Utility-first styling

Backend

- Node.js - Runtime environment
- Express 5.2 - Web framework
- MongoDB 9.0 - NoSQL database
- Mongoose - ODM for MongoDB
- JWT - Authentication tokens
- bcryptjs - Password hashing
- CORS - Cross-origin requests

AI & External APIs

- Google Gemini 2.5 Flash - AI content generation
- Google Search Integration - News sourcing

Development

- TypeScript 5.8 - Type checking
- Vite - Build tooling

Application Flow

```
User Registration/Login
    ↓
Dashboard (View Stats & Recent Articles)
    ├─→ Live Scraper
    │   ├─→ Select Source
    │   ├─→ Fetch Articles (Google Search + Gemini)
    │   ├─→ Generate Notes (AI Analysis)
    │   └─→ Save to Database
    ├─→ Saved Notes
    │   ├─→ Search & Filter
    │   ├─→ View Details
    │   └─→ Delete
    └─→ Daily Digest (Generate from Recent Articles)
```

🗄️ Database Schema

User Collection

```
{
  _id: ObjectId,
  name: String,
  email: String (unique),
  password: String (hashed),
  createdAt: Date
}
```

Note Collection

```
{
  _id: ObjectId,
  userId: ObjectId (ref: User),
  articleId: String,
  title: String,
  source: String,
  gsPaper: Enum['GS1', 'GS2', 'GS3', 'GS4', 'Prelims', 'Essay'],
  tags: [String],
  summary: String,
  content: String (Markdown),
  mcqs: [{
    question: String,
    options: [String],
    correctOption: Number,
    explanation: String
  }],
  mainsQuestion: {
    question: String,
    modelAnswerPoints: [String]
  },
  createdAt: Date
}
```

Article Collection

```
{
  _id: ObjectId,
  userId: ObjectId (ref: User),
  id: String,
  title: String,
  source: String,
  url: String,
  summary: String,
  publishedDate: String,
  scrapedAt: Date
}
```

🔌 API Endpoints

Authentication

```
POST /api/auth/register
POST /api/auth/login
```

Notes

```
GET  /api/notes          - Get all user notes
POST /api/notes          - Create new note
DELETE /api/notes/:id    - Delete note by ID
```

Articles

```
GET  /api/articles       - Get article history
POST /api/articles       - Save article
```

Health Check

```
GET  /api/health         - Check server & DB status
```

⚙️ Configuration

Environment Variables

Variable	Description	Example
`GEMINI_API_KEY`	Google Gemini API key	`AIza...`
`MONGODB_URI`	MongoDB connection string	`mongodb+srv://...`
`JWT_SECRET`	JWT signing secret	`your_secret_key`
`PORT`	Backend server port (optional)	`5000`

Fallback Behavior

The application automatically handles backend failures:

- **When MongoDB unavailable**: Notes save to browser localStorage
- **When Backend offline**: Mock tokens created for login
- **When API fails**: Previous cached data used

🔒 Security Features

1. **Password Hashing** - bcryptjs with salt rounds
2. **JWT Authentication** - Secure token-based auth
3. **CORS Protection** - Cross-origin requests validated
4. **HTTP-Only Considerations** - Tokens stored securely
5. **Input Validation** - Server-side validation of all inputs
6. **Environment Secrets** - Sensitive data in .env.local

📱 Browser Compatibility

- Chrome/Chromium 90+
- Firefox 88+
- Safari 14+
- Edge 90+

🐛 Troubleshooting

Backend Connection Issues

Error: "Failed to fetch"

- Ensure backend server is running: `node server.js`
- Check MongoDB connection string in `.env.local`
- Verify GEMINI_API_KEY is valid
- Check firewall isn't blocking port 5000

Error: "Cannot find module 'dotenv'"

```
npm install dotenv
```

Frontend Issues

Error: "Module not found 'vite'"

```
npm install
```

Port 3000 already in use

```
# Kill process on port 3000
# On Windows: netstat -ano | findstr :3000
# Get PID and: taskkill /PID <PID> /F
```

Database Issues

Error: "Authentication failed"

- Verify MongoDB Atlas credentials
- Check IP whitelist in MongoDB Atlas settings
- Ensure database name matches in MONGODB_URI

No data persisting

- Check MongoDB connection is established (look for ✅ message in console)
- Verify user is authenticated
- Check browser localStorage isn't disabled

AI Generation Fails

Error: "API Key missing"

- Add `GEMINI_API_KEY` to `.env.local`
- Restart backend server for changes to take effect

No notes generated

- Verify Gemini API quota isn't exceeded
- Check network requests in browser DevTools
- Try with a different news source

📊 Performance Tips

1. **Database Indexing** - Consider indexing userId and gsPaper fields
2. **Caching** - Recent articles cached in browser
3. **Pagination** - Add pagination for 1000+ notes
4. **Image Optimization** - Compress article images
5. **API Rate Limiting** - Implement to prevent abuse

🤝 Contributing

To contribute to UPSC AI:

1. Fork the repository
2. Create feature branch: `git checkout -b feature/amazing-feature`
3. Commit changes: `git commit -m 'Add amazing feature'`
4. Push to branch: `git push origin feature/amazing-feature`
5. Open a Pull Request

📝 License

This project is licensed under the MIT License - see LICENSE file for details.

📞 Support & Contact

- **Issues**: Report bugs in GitHub Issues
- **Email**: ayushkumar.inspire@gmail.com
- **Documentation**: See PROJECT_REPORT.md for technical details

---

🎓 Disclaimer

This tool is designed to supplement UPSC preparation. Users should:

- Verify information from official sources
- Not rely solely on AI-generated content
- Cross-check with NCERT and standard resources
- Consult coaching materials when needed

🙏 Acknowledgments

- Google Gemini API for AI capabilities
- MongoDB for reliable database
- React community for excellent libraries
- UPSC aspirants for valuable feedback

Last Updated: December 11, 2025
Version: 1.0.0

Happy Studying! 📚 ✨