

Task Tracker Documentation

Table of Contents

1. [Overview](#)
2. [Features](#)
3. [Technology Stack](#)
4. [Installation & Setup](#)
5. [Project Structure](#)
6. [Backend API](#)
7. [Frontend Components](#)
8. [AI Chatbot Integration](#)
9. [Usage Guide](#)
10. [Environment Configuration](#)
11. [Deployment](#)
12. [Contributing](#)
13. [Troubleshooting](#)

Overview

Project Description

Task Tracker is a modern, full-stack task management application that helps users organize and manage their daily tasks efficiently. Built with Node.js and React, it provides a clean, intuitive interface for task management with AI-powered assistance.

Key Objectives

- Provide a seamless task management experience
- Integrate AI-powered chatbot for task guidance
- Ensure responsive design across all devices
- Maintain secure user authentication and data protection

Demo

- **Repository:** <https://github.com/ShivamNishad0/Task-Tracker.git>

- **Frontend:** <http://localhost:5173>
- **Backend API:** <http://localhost:4000>

Features

Core Features

- **Task CRUD Operations** - Create, read, update, and delete tasks
- **User Authentication** - Secure login and registration system
- **Real-time Updates** - Instant task updates across the interface
- **Responsive Design** - Works seamlessly on desktop and mobile devices
- **Task Status Management** - Mark tasks as pending, in-progress, or completed

AI-Powered Features

- **AI Chatbot** - Get intelligent assistance for task management
- **Task Query System** - Search and analyze tasks using natural language
- **Step-by-step Guidance** - Receive AI-generated guidance for task completion
- **Smart Suggestions** - Get AI-powered suggestions for task optimization
- **Task Analysis** - AI analysis of task descriptions and requirements

Additional Features

- **Clean UI/UX** - Modern interface built with TailwindCSS
- **Toast Notifications** - Real-time feedback using React-Toastify
- **Icon Integration** - Beautiful icons from Lucide React
- **Client-side Routing** - Smooth navigation with React Router DOM

Technology Stack

Backend

- **Node.js** - Runtime environment
- **Express.js** - Web application framework
- **MongoDB** - NoSQL database for data storage
- **Mongoose** - MongoDB object modeling library
- **JWT** - JSON Web Tokens for authentication

- **Google Generative AI** - AI integration for chatbot functionality

Frontend

- **React** - User interface library
- **Vite** - Modern build tool and development server
- **TailwindCSS** - Utility-first CSS framework
- **React Router DOM** - Client-side routing
- **Lucide React** - Icon library
- **React-Toastify** - Toast notification system

Development Tools

- **Nodemon** - Automatic server restart during development
- **ESLint** - Code linting and formatting
- **Git** - Version control system

Installation & Setup

Prerequisites

- Node.js (version 14.0.0 or higher)
- npm or yarn package manager
- MongoDB (local installation or MongoDB Atlas account)
- Google Gemini API key (for AI features)

Installation Steps

1. Clone the repository

```
bash

git clone https://github.com/ShivamNishad0/Task-Tracker.git
cd Task-Tracker
```

2. Backend Setup

```
bash
```

```
# Navigate to backend directory
```

```
cd backend
```

```
# Install dependencies
```

```
npm install
```

```
# Install AI dependency
```

```
npm install @google/generative-ai
```

3. Backend Environment Configuration Create a `.env` file in the `backend` folder:

```
bash
```

```
MONGO_URI=your_mongodb_connection_string
```

```
PORT=4000
```

```
JWT_SECRET=your_jwt_secret_key
```

```
GEMINI_API_KEY=your_gemini_api_key
```

4. Start Backend Server

```
bash
```

```
npm start
```

5. Frontend Setup

```
bash
```

```
# Navigate to frontend directory (from root)
```

```
cd frontend
```

```
# Install dependencies
```

```
npm install
```

6. Start Frontend Development Server

```
bash
```

```
npm run dev
```

7. Access the Application

- Frontend: <http://localhost:5173>
- Backend API: <http://localhost:4000>

Project Structure

```
task-tracker/
├── backend/
│   ├── config/
│   │   └── database.js      # Database connection configuration
│   ├── controllers/
│   │   ├── authController.js  # Authentication logic
│   │   └── taskController.js  # Task CRUD and AI operations
│   ├── middleware/
│   │   └── authMiddleware.js  # JWT authentication middleware
│   ├── models/
│   │   ├── User.js           # User schema
│   │   └── Task.js           # Task schema
│   ├── routes/
│   │   ├── authRoute.js      # Authentication routes
│   │   └── taskRoute.js      # Task management routes
│   ├── utils/
│   │   └── geminiAI.js       # AI integration utility
│   ├── server.js             # Main server file
│   ├── package.json
│   └── .env                  # Environment variables
├── frontend/
│   ├── src/
│   │   ├── assets/           # Static assets
│   │   ├── components/
│   │   │   ├── Chatbot.jsx   # AI chatbot component
│   │   │   ├── TaskCard.jsx  # Individual task display
│   │   │   ├── TaskForm.jsx  # Task creation/editing form
│   │   │   └── Navbar.jsx    # Navigation component
│   │   ├── pages/
│   │   │   ├── Dashboard.jsx # Main dashboard page
│   │   │   ├── Login.jsx     # Login page
│   │   │   ├── Register.jsx  # Registration page
│   │   │   └── Profile.jsx    # User profile page
│   │   ├── App.jsx           # Main app component
│   │   ├── index.css         # Global styles
│   │   └── main.jsx          # App entry point
│   └── package.json
```

| vite.config.js # Vite configuration
| tailwind.config.js # TailwindCSS configuration
| README.md

Backend API

Authentication Endpoints

- `POST /api/auth/register` - User registration
- `POST /api/auth/login` - User login
- `GET /api/auth/profile` - Get user profile (requires authentication)

Task Management Endpoints

- `GET /api/tasks` - Get all user tasks
- `POST /api/tasks` - Create a new task
- `GET /api/tasks/:id` - Get specific task
- `PUT /api/tasks/:id` - Update task
- `DELETE /api/tasks/:id` - Delete task

AI Chatbot Endpoints

- `POST /api/tasks/chatbot/query` - Process AI queries about tasks
- `GET /api/tasks/chatbot/suggestions` - Get AI-generated task suggestions

Request/Response Examples

Create Task

```
javascript

// POST /api/tasks
{
  "title": "Complete project documentation",
  "description": "Write comprehensive documentation for the task tracker app",
  "priority": "high",
  "dueDate": "2024-12-31",
  "status": "pending"
}
```

AI Chatbot Query

```
javascript

// POST /api/tasks/chatbot/query
{
  "query": "Help me with my project planning task"
}
```

Frontend Components

Core Components

Dashboard.jsx

- **Purpose:** Main application dashboard displaying all tasks
- **Features:**
 - Task grid display
 - AI chatbot integration
 - Task filtering and sorting
 - Quick task creation
- **State Management:** Uses React hooks for task data and UI state

Chatbot.jsx

- **Purpose:** AI-powered chatbot interface
- **Features:**
 - Natural language task queries
 - Real-time AI responses
 - Task analysis and guidance
 - Chat history management
- **Props:**
 - `isOpen` (boolean) - Controls chatbot visibility
 - `onClose` (function) - Callback for closing chatbot

TaskCard.jsx

- **Purpose:** Individual task display component

- **Features:**
 - Task information display
 - Status indicators
 - Quick action buttons
 - Priority visualization
- **Props:**
 - `task` (object) - Task data
 - `onUpdate` (function) - Update callback
 - `onDelete` (function) - Delete callback

TaskForm.jsx

- **Purpose:** Task creation and editing form
- **Features:**
 - Form validation
 - Date picker integration
 - Priority selection
 - Rich text description
- **Props:**
 - `task` (object, optional) - Task data for editing
 - `onSubmit` (function) - Form submission callback

AI Chatbot Integration

Setup Requirements

1. **Google Gemini API Key:** Obtain from [Google AI Studio](#)
2. **Environment Variable:** Add `GEMINI_API_KEY` to backend `.env` file
3. **Dependencies:** Install `@google/generative-ai` package

Features

- **Task Query Processing:** Natural language task searches
- **Intelligent Analysis:** AI-powered task description analysis
- **Step-by-step Guidance:** Detailed task completion guidance

- **Smart Suggestions:** Context-aware task management tips
- **User-friendly Interface:** Clean chat UI with loading states

Usage Examples

```
javascript
```

```
// Query examples that work well with the AI chatbot:
```

```
"Help me with my project planning task"
```

```
"How to complete the marketing campaign"
```

```
"What should I do for the code review task"
```

```
"Give me suggestions for task management"
```

Security Features

- Authentication required for all chatbot requests
- API keys stored securely in environment variables
- User data filtered by ownership
- Error handling for invalid queries and network issues

Usage Guide

Getting Started

1. **Register/Login:** Create an account or login to existing account
2. **Create Tasks:** Use the task creation form to add new tasks
3. **Manage Tasks:** Update task status, edit details, or delete completed tasks
4. **Use AI Assistant:** Click the "Ask AI" button for intelligent task assistance

Task Management Workflow

1. **Create:** Add new tasks with title, description, priority, and due date
2. **Organize:** Set appropriate priority levels and due dates
3. **Track:** Monitor progress using status updates (pending, in-progress, completed)
4. **Analyze:** Use AI chatbot for task analysis and guidance
5. **Complete:** Mark tasks as completed when finished

AI Chatbot Usage

1. **Access:** Click the green "Ask AI" button on the dashboard

2. **Query:** Type natural language questions about your tasks
3. **Analyze:** Review AI-generated analysis and suggestions
4. **Follow:** Implement step-by-step guidance provided by the AI
5. **Close:** Close the chatbot when finished

Environment Configuration

Backend Environment Variables

```
bash

# Database Configuration
MONGO_URI=mongodb://localhost:27017/tasktracker
# or for MongoDB Atlas:
MONGO_URI=mongodb+srv://username:password@cluster.mongodb.net/tasktracker

# Server Configuration
PORT=4000

# Authentication
JWT_SECRET=your_super_secure_jwt_secret_key

# AI Integration
GEMINI_API_KEY=your_gemini_api_key_here
```

Frontend Configuration

The frontend uses Vite's built-in environment variable system. Create `.env` file in frontend directory if needed:

```
bash

VITE_API_URL=http://localhost:4000
```

Deployment

Backend Deployment

1. **Build Process:** No build step required for Node.js backend
2. **Environment Setup:** Configure production environment variables
3. **Database:** Set up production MongoDB database

4. **Process Manager:** Use PM2 or similar for production deployment

```
bash

# Using PM2
npm install -g pm2
pm2 start server.js --name "task-tracker-backend"
```

Frontend Deployment

1. Build for Production

```
bash

npm run build
```

2. Preview Build Locally

```
bash

npm run preview
```

3. **Deploy:** Upload `dist` folder to hosting service

Deployment Platforms

- **Frontend:** Netlify, Vercel, GitHub Pages
- **Backend:** Heroku, Railway, DigitalOcean, AWS
- **Database:** MongoDB Atlas (recommended for production)

Contributing

Development Workflow

1. Fork the repository
2. Create a feature branch (`git checkout -b feature/AmazingFeature`)
3. Make your changes
4. Test thoroughly (both backend and frontend)
5. Commit your changes (`git commit -m 'Add some AmazingFeature'`)
6. Push to the branch (`git push origin feature/AmazingFeature`)

7. Open a Pull Request

Code Style Guidelines

- Use consistent indentation (2 spaces)
- Follow JavaScript ES6+ conventions
- Use meaningful variable and function names
- Add comments for complex logic
- Maintain consistent file and folder structure

Testing Guidelines

- Test all API endpoints using Postman or similar tools
- Verify frontend functionality across different browsers
- Test AI chatbot with various query types
- Ensure responsive design works on different screen sizes

Troubleshooting

Common Backend Issues

Database Connection Error

Problem: Cannot connect to MongoDB **Solution:**

- Verify MongoDB is running locally or MongoDB Atlas connection string is correct
- Check network connectivity
- Ensure IP whitelist includes your IP (for Atlas)

JWT Authentication Error

Problem: Authentication fails or token expired **Solution:**

- Verify JWT_SECRET is set in environment variables
- Check token expiration settings
- Clear browser storage and re-login

AI Chatbot Not Working

Problem: Gemini API errors or no responses **Solution:**

- Verify GEMINI_API_KEY is correctly set
- Check API quota and billing status
- Ensure internet connectivity for API calls

Common Frontend Issues

Build Errors

Problem: Vite build fails **Solution:**

- Clear node_modules and reinstall dependencies
- Check for syntax errors in React components
- Verify all imports are correct

API Connection Issues

Problem: Frontend cannot reach backend API **Solution:**

- Ensure backend server is running on correct port
- Check CORS configuration in backend
- Verify API endpoint URLs in frontend code

Performance Optimization Tips

- Implement lazy loading for large task lists
- Use React.memo for frequently re-rendering components
- Optimize database queries with proper indexing
- Implement caching for AI responses
- Compress images and assets

Browser Compatibility

- **Supported Browsers:** Chrome 80+, Firefox 75+, Safari 13+, Edge 80+
- **Mobile Support:** iOS Safari 13+, Android Chrome 80+
- **Known Issues:** Some older browsers may not support all ES6+ features

Scripts Reference

Backend Scripts

```
bash
```

```
npm start      # Start server with nodemon  
npm run dev    # Development mode (if configured)
```

Frontend Scripts

```
bash
```

```
npm run dev    # Start development server  
npm run build  # Build for production  
npm run preview # Preview production build
```

License

This project is licensed under the MIT License - see the [LICENSE](#) file for details.

Contact & Support

- **Developer:** Shivam Nishad
- **Repository:** [Task-Tracker GitHub](#)
- **Issues:** Open an issue on GitHub for bug reports or feature requests

Made with ❤️ for developers who love clean code

★ Star this repo if you found it helpful! ★

Last updated: September 2025