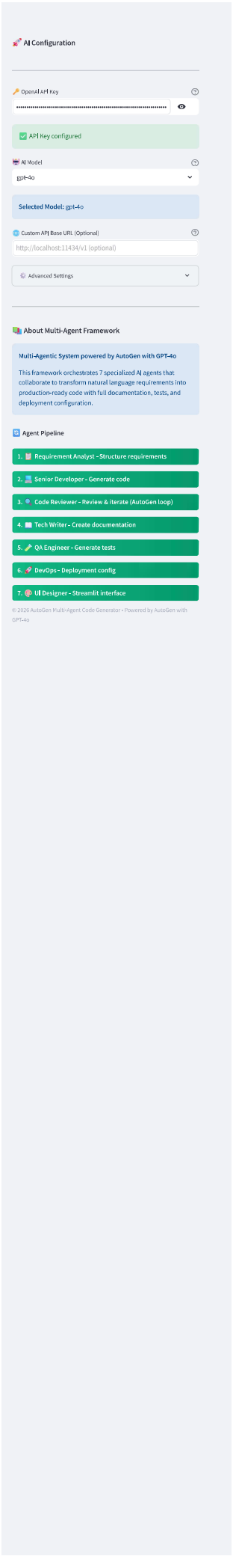
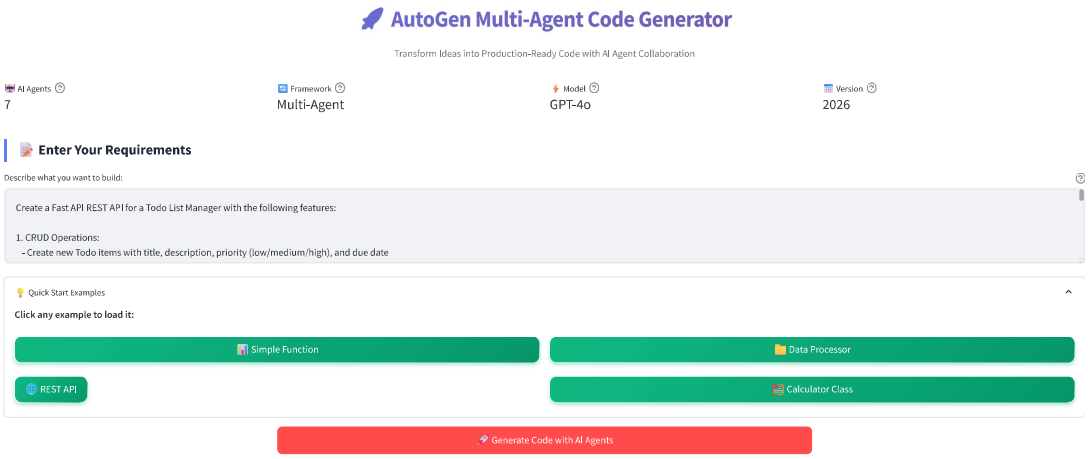
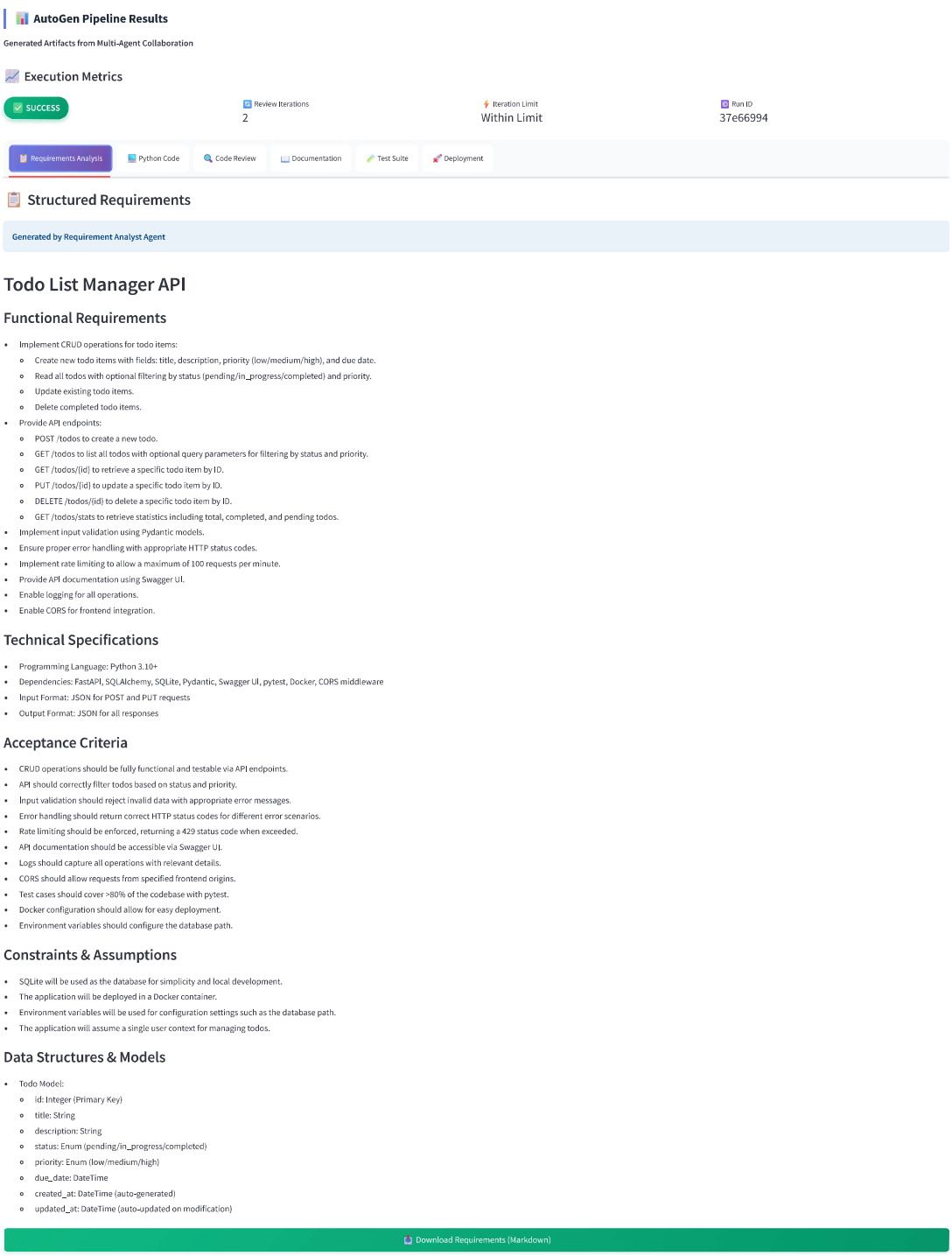
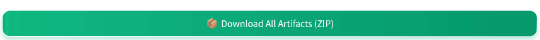

AI Configuration
OpenAI API Key
API Key configured
AI Model
gpt-4o
Selected Model: gpt-4o
Custom API Base URL (Optional)
http://localhost:11434/v1 (optional)
Advanced Settings
About Multi-Agent Framework
Multi-Agentic System powered by AutoGen with GPT-4o
This framework orchestrates 7 specialized AI agents that collaborate to transform natural language requirements into production-ready code with full documentation, tests, and deployment configuration.
Agent Pipeline
1. Requirement Analyst - Structure requirements
2. Senior Developer - Generate code
3. Code Reviewer - Review & iterate (AutoGen loop)
4. Tech Writer - Create documentation
5. QA Engineer - Generate tests
6. DevOps - Deployment config
7. UI Designer - Streamlit interface
© 2026 AutoGen Multi-Agent Code Generator • Powered by AutoGen with GPT-4o


AutoGen Multi-Agent Code Generator
Transform Ideas into Production-Ready Code with AI Agent Collaboration
AI Agents
7
Framework
Multi-Agent
Model
GPT-4o
Version
2026
Enter Your Requirements
Describe what you want to build:
Create a Fast API REST API for a Todo List Manager with the following features:
1. CRUD Operations:
- Create new Todo items with title, description, priority (low/medium/high), and due date
Quick Start Examples
Click any example to load it:
Simple Function
Data Processor
REST API
Calculator Class
Generate Code with AI Agents


AutoGen Pipeline Results
Generated Artifacts from Multi-Agent Collaboration
Execution Metrics
SUCCESS
Review Iterations
2
Iteration Limit
Within Limit
Run ID
37e66994
Requirements Analysis
Python Code
Code Review
Documentation
Test Suite
Deployment
Structured Requirements
Generated by Requirement Analyst Agent
Todo List Manager API
Functional Requirements
Implement CRUD operations for todo items:
Create new todo items with fields: title, description, priority (low/medium/high), and due date.
Read all todos with optional filtering by status (pending/in_progress/completed) and priority.
Update existing todo items.
Delete completed todo items.
Provide API endpoints:
POST /todos to create a new todo.
GET /todos to list all todos with optional query parameters for filtering by status and priority.
GET /todos/{id} to retrieve a specific todo item by ID.
PUT /todos/{id} to update a specific todo item by ID.
DELETE /todos/{id} to delete a specific todo item by ID.
GET /todos/stats to retrieve statistics including total, completed, and pending todos.
Implement input validation using Pydantic models.
Ensure proper error handling with appropriate HTTP status codes.
Implement rate limiting to allow a maximum of 100 requests per minute.
Provide API documentation using Swagger UI.
Enable logging for all operations.
Enable CORS for frontend integration.
Technical Specifications
Programming Language: Python 3.10+
Dependencies: FastAPI, SQLAlchemy, SQLite, Pydantic, Swagger UI, pytest, Docker, CORS middleware
Input Format: JSON for POST and PUT requests
Output Format: JSON for all responses
Acceptance Criteria
CRUD operations should be fully functional and testable via API endpoints.
API should correctly filter todos based on status and priority.
Input validation should reject invalid data with appropriate error messages.
Error handling should return correct HTTP status codes for different error scenarios.
Rate limiting should be enforced, returning a 429 status code when exceeded.
API documentation should be accessible via Swagger UI.
Logs should capture all operations with relevant details.
CORS should allow requests from specified frontend origins.
Test cases should cover >80% of the codebase with pytest.
Docker configuration should allow for easy deployment.
Environment variables should configure the database path.
Constraints & Assumptions
SQLite will be used as the database for simplicity and local development.
The application will be deployed in a Docker container.
Environment variables will be used for configuration settings such as the database path.
The application will assume a single user context for managing todos.
Data Structures & Models
Todo Model:
id: Integer (Primary Key)
title: String
description: String
status: Enum (pending/in_progress/completed)
priority: Enum (low/medium/high)
due_date: DateTime
created_at: DateTime (auto-generated)
updated_at: DateTime (auto-updated on modification)
Download Requirements (Markdown)


Download All Artifacts (ZIP)