AXIO - AI Personal Assistant

by Perfionix AI

An intelligent, feature-rich AI-powered personal assistant with advanced chat capabilities, document analysis, data visualization, task management, and productivity tools.

VERSION 2.0

PYTHON 3.8+

FLASK 3.0.0

LICENSE MIT

PERFIONIX AI

Table of Contents

Overview

Axio is a comprehensive AI-powered personal assistant developed by **Perfionix AI**. It combines the power of Large Language Models (LLMs) with modern web technologies to provide users with an intelligent, intuitive, and feature-rich productivity platform.

Axio goes beyond simple chatbots by offering specialized modules for document analysis (DocIQ), data visualization (VizIQ), task management, note-taking, and reminders - all wrapped in a beautiful, responsive dark-themed interface.

Problem Statement

In today's digital world, professionals and individuals face several challenges:

Problem	Impact
Information Overload	Difficulty in extracting insights from large documents and datasets
Fragmented Tools	Using multiple apps for chat, tasks, notes, and data analysis
Complex Data Analysis	Non-technical users struggle to visualize and understand data
Document Comprehension	Time-consuming process to read and analyze lengthy documents
Productivity Gaps	Lack of unified platform for personal productivity management
Accessibility	Enterprise AI solutions are expensive and complex to deploy

Solution

Axio provides a comprehensive solution to modern productivity challenges:

🎯 Core Solutions

#	Solution	Description	Benefit
1	**Unified AI Platform**	Single application combining chat, documents, data, and tasks	Eliminates need for multiple tools
2	**Intelligent Chat Assistant**	LLM-powered conversational AI with web search	Instant answers and code help
3	**Document Intelligence (DocIQ)**	RAG-based document Q&A system	Extract insights from any document
4	**Auto Data Visualization (VizIQ)**	Upload data, get instant dashboards	No coding required for analytics
5	**Smart Task Management**	Priority-based task tracking	Stay organized and productive
6	**Local & Private Processing**	All data processed on your machine	Complete data privacy

💡 How Axio Solves Each Problem

❌ **Problem: Information Overload** • Too many documents to read • Complex data to analyze • Time-consuming research	✅ **Axio Solution:** • **DocIQ** extracts key information instantly • **VizIQ** auto-generates visual insights • **AI Chat** summarizes and explains
❌ **Problem: Fragmented Tools** • Multiple apps for different tasks • Context switching overhead • Data scattered everywhere	✅ **Axio Solution:** • **All-in-one platform** with 6 integrated modules • **Seamless navigation** between features • **Centralized workspace** for everything
❌ **Problem: Complex Data Analysis** • Need technical skills for visualization • Manual chart creation is tedious	✅ **Axio Solution:** • **Auto-generated KPIs** from your data

- Hard to identify patterns	- **Smart chart selection** (Column, Pie, Line, etc.) - **AI-powered insights** highlight key findings
❌ **Problem: Document Comprehension** - Long documents take hours to read - Finding specific information is hard - No easy way to query documents	✅ **Axio Solution:** - **Natural language Q&A** with documents - **RAG technology** finds relevant sections - **Instant answers** from uploaded files

⭐ Key Value Propositions

Value	Description
🔗 **Unified Platform**	All productivity tools in one place - no more app switching
🤖 **AI-Powered Intelligence**	Leveraging Large Language Models for smart, contextual responses
📊 **No-Code Data Analysis**	Upload CSV/Excel, get instant KPIs, charts, and insights
📄 **Document Q&A**	Chat with your documents using RAG (Retrieval Augmented Generation)
🎨 **Beautiful UX**	Modern, responsive, dark-themed interface with smooth animations
🔒 **Local & Private**	All data stays on your machine - complete privacy control
⚡ **Real-time Processing**	Instant responses with streaming AI and live updates
🌐 **Web Search Integration**	Get up-to-date information from the internet

Key Features

1. AI Chat Assistant

- Real-time conversational AI powered by LLM
- Syntax highlighting for 10+ programming languages
- Markdown rendering with code blocks
- Web search integration for real-time information
- Message editing and regeneration
- Typing indicators and streaming responses

2. DocIQ - Document Intelligence

- Upload and analyze PDF, DOCX, TXT, and MD files
- RAG (Retrieval Augmented Generation) for accurate answers
- Chat with your documents naturally
- Multi-document support
- Context-aware responses

3. VizIQ - Data Visualization

- Upload CSV, Excel, JSON data files
- Automatic data preprocessing and cleaning
- AI-generated dashboard names and descriptions
- Auto-generated KPIs (Key Performance Indicators)
- Multiple chart types:
    - Column Charts
    - Distribution Charts (Doughnut)
    - Comparison Charts
    - Trend Analysis (Line Charts)
- AI-powered insights generation
- Interactive data preview table

4. Task Management

- Create, edit, delete tasks
- Priority levels (High, Medium, Low)
- Status tracking (Pending, Completed)

- Filter and sort capabilities
- Visual progress indicators

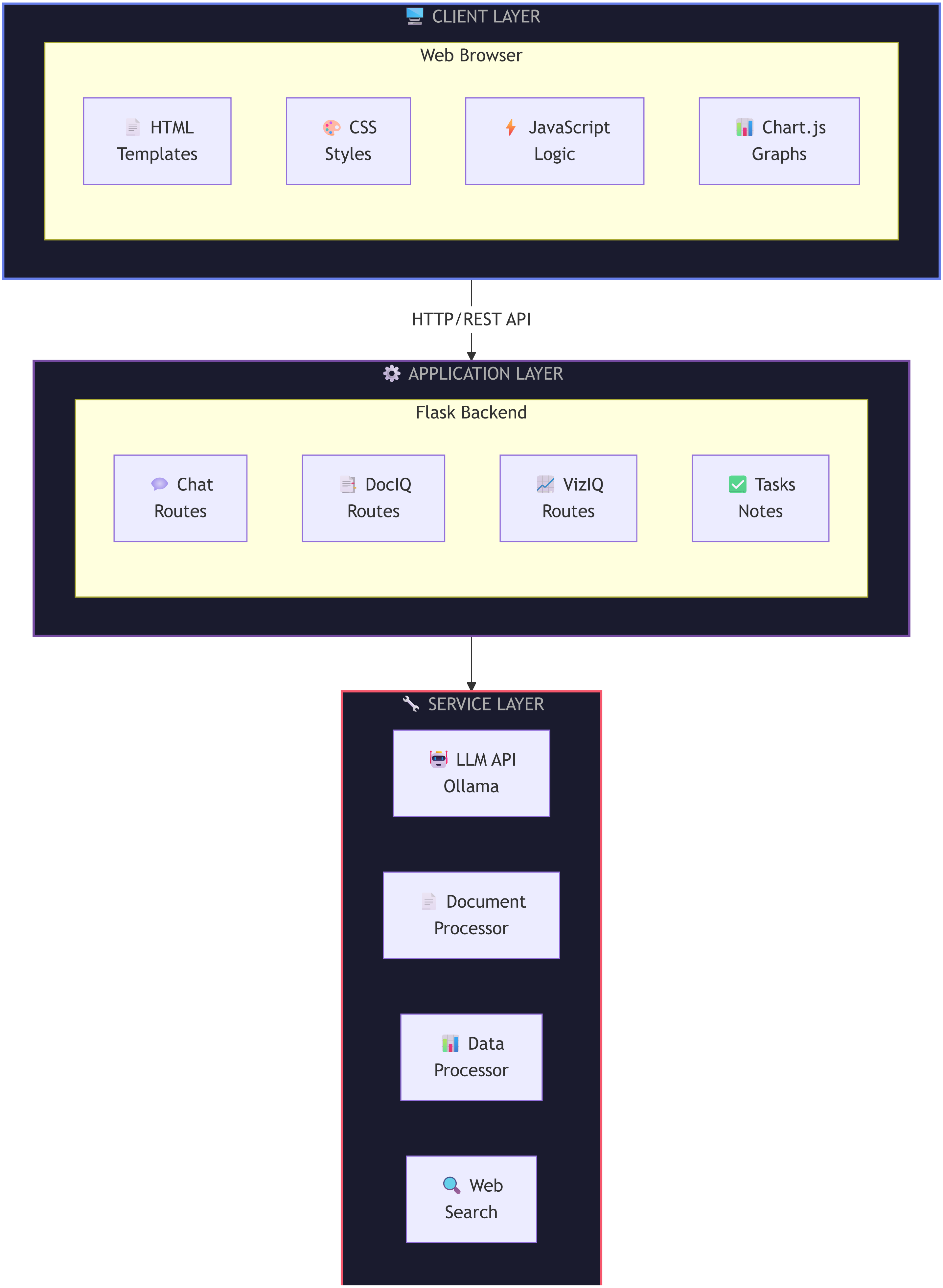
5. Notes System

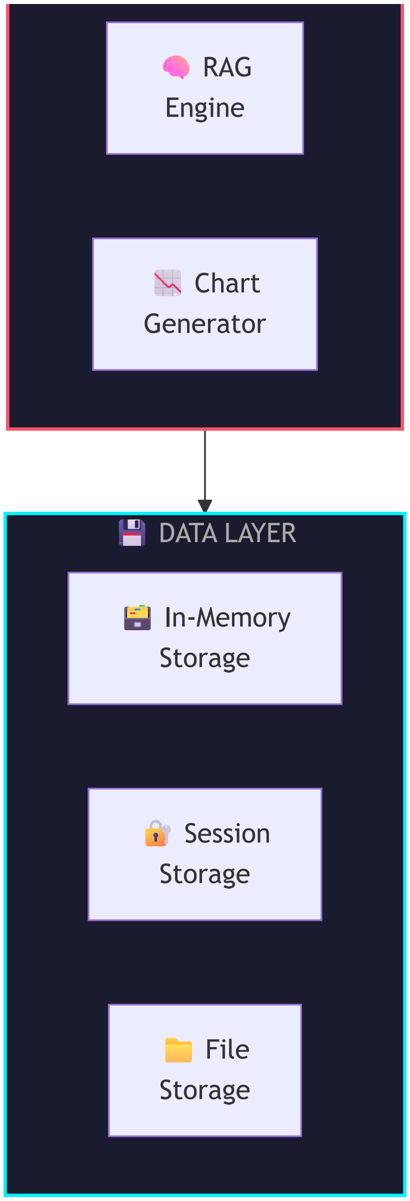
- Rich text note creation
- Card-based layout
- Quick access and search
- Timestamp tracking

6. Reminders

- Time-based reminder alerts
- Date and time picker
- Reminder management interface

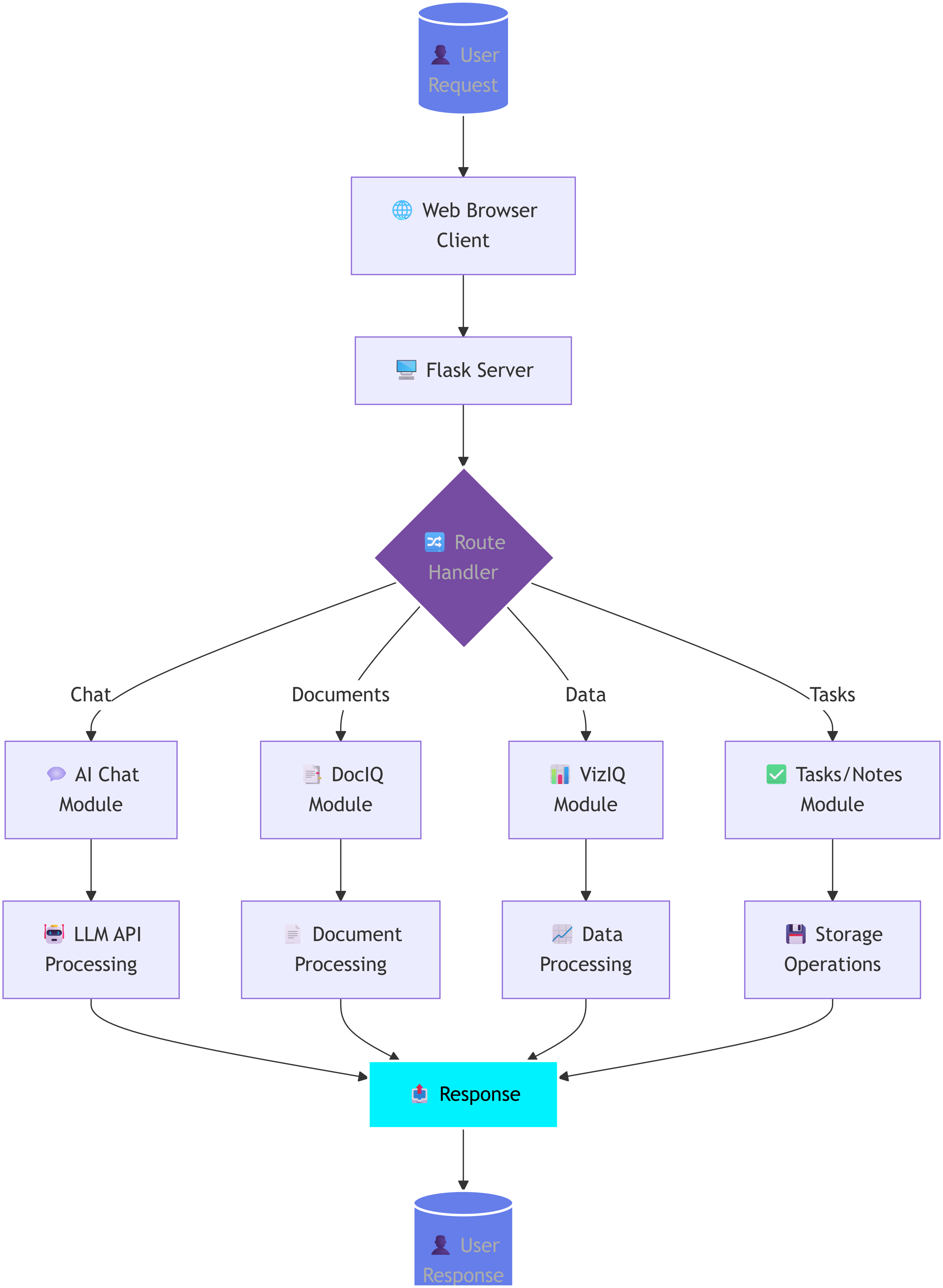
System Architecture


🖥️ CLIENT LAYER
Web Browser
📄 HTML Templates
🎨 CSS Styles
⚡ JavaScript Logic
📊 Chart.js Graphs
HTTP/REST API
⚙️ APPLICATION LAYER
Flask Backend
💬 Chat Routes
📑 DocIQ Routes
📈 VizIQ Routes
✅ Tasks Notes
🔧 SERVICE LAYER
🤖 LLM API Ollama
📄 Document Processor
📊 Data Processor
🔍 Web Search

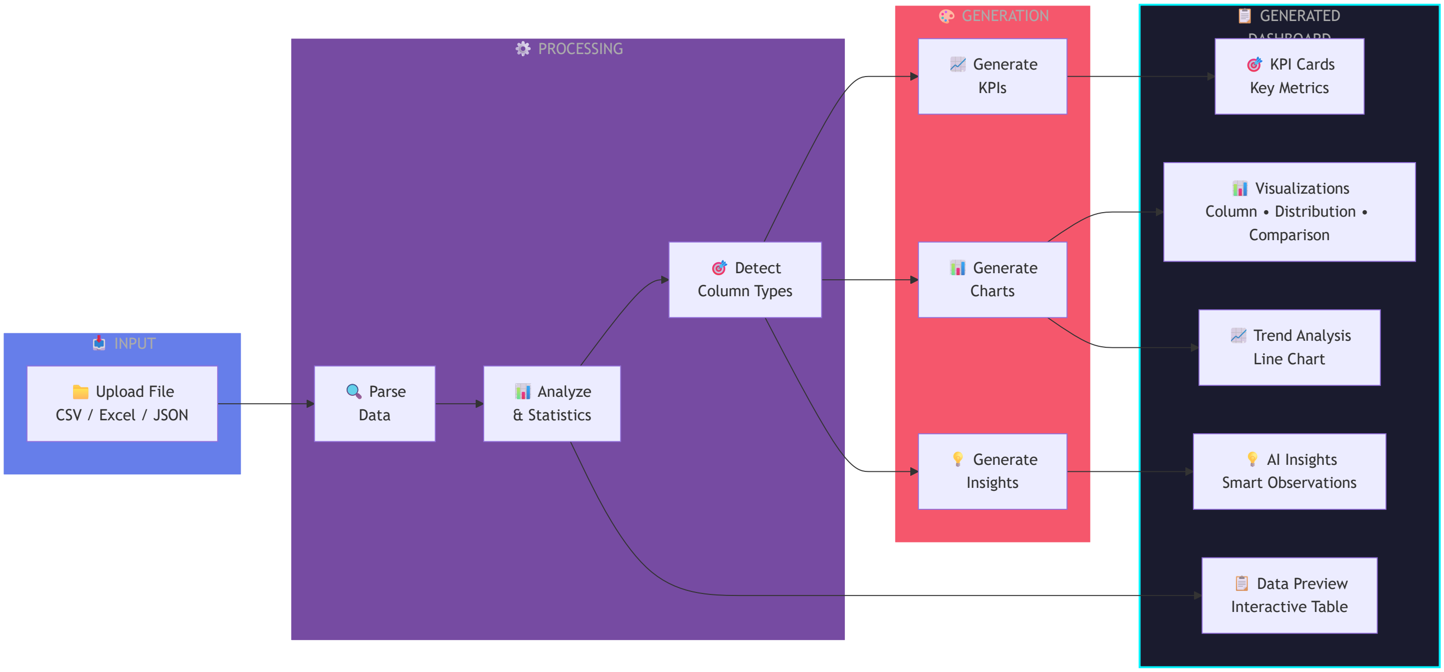

🧠 RAG Engine
📉 Chart Generator
💾 DATA LAYER
🗃️ In-Memory Storage
🔐 Session Storage
📁 File Storage

Workflow Diagrams

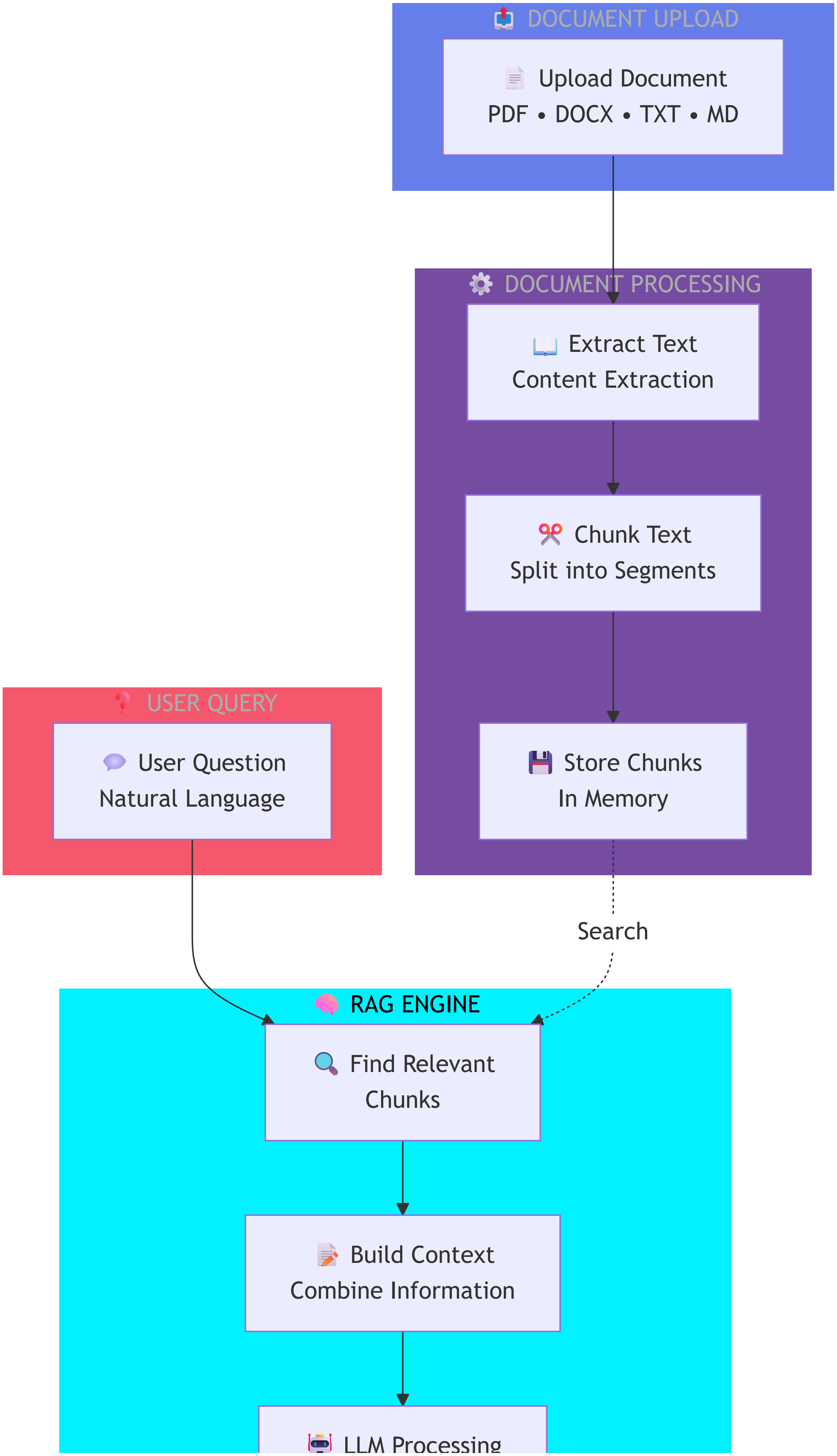
Main Application Flow

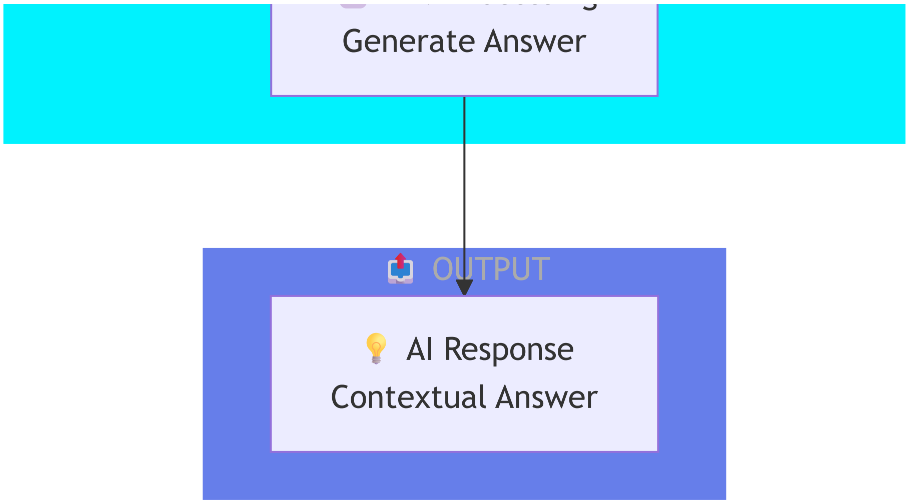

User Request
Web Browser Client
Flask Server
Route Handler
Chat
Documents
Data
Tasks
AI Chat Module
DocIQ Module
VizIQ Module
Tasks/Notes Module
LLM API Processing
Document Processing
Data Processing
Storage Operations
Response
User Response

VizIQ Data Flow

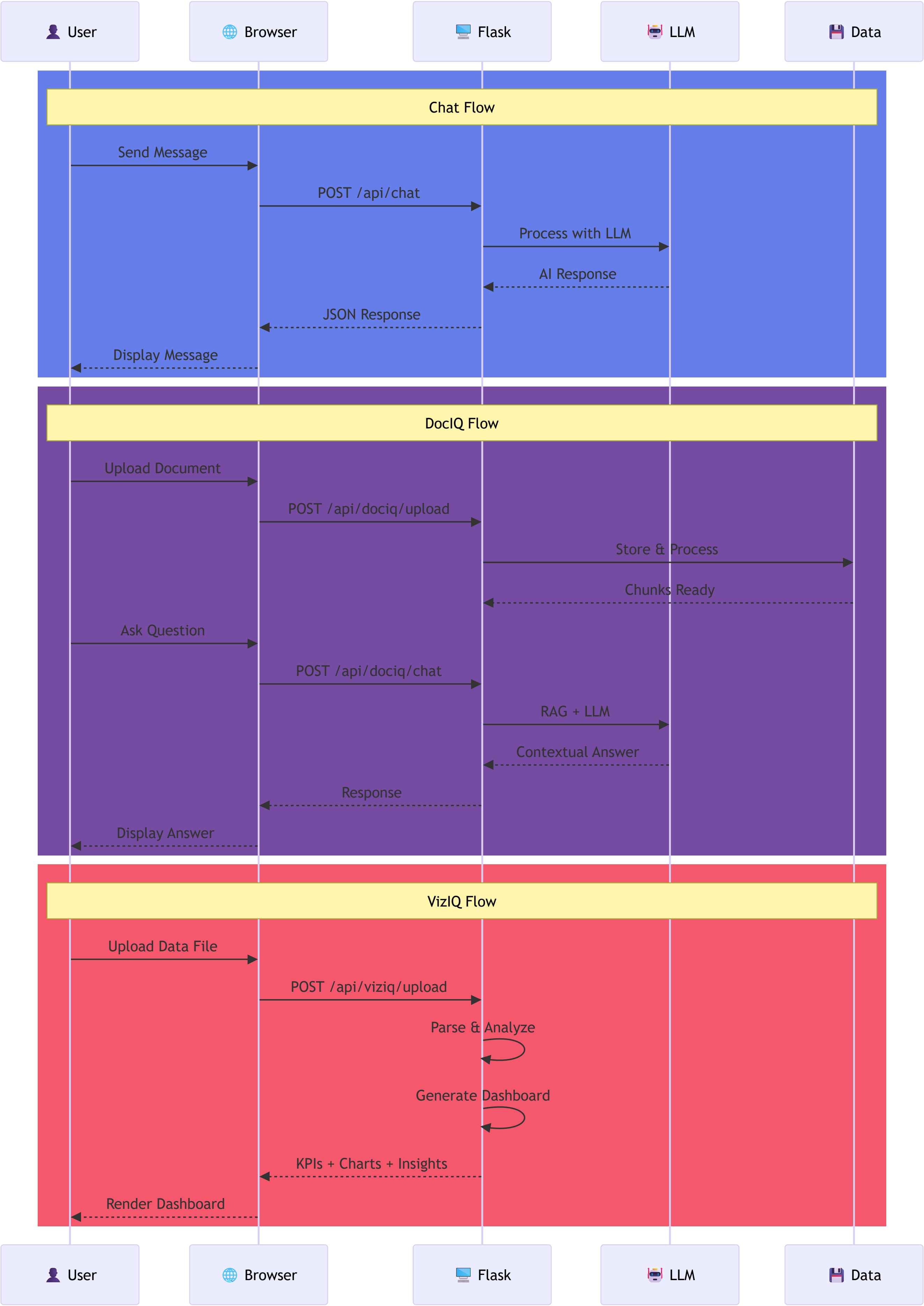


DocIQ RAG Flow


📤 DOCUMENT UPLOAD
📄 Upload Document
PDF • DOCX • TXT • MD
⚙️ DOCUMENT PROCESSING
📖 Extract Text
Content Extraction
✂️ Chunk Text
Split into Segments
💾 Store Chunks
In Memory
❓ USER QUERY
💬 User Question
Natural Language
Search
🧠 RAG ENGINE
🔍 Find Relevant Chunks
📝 Build Context
Combine Information
🤖 LLM Processing



Complete System Interaction


👤 User
🌐 Browser
🖥️ Flask
🤖 LLM
💾 Data
Chat Flow
Send Message
POST /api/chat
Process with LLM
AI Response
JSON Response
Display Message
DocIQ Flow
Upload Document
POST /api/dociq/upload
Store & Process
Chunks Ready
Ask Question
POST /api/dociq/chat
RAG + LLM
Contextual Answer
Response
Display Answer
VizIQ Flow
Upload Data File
POST /api/viziq/upload
Parse & Analyze
Generate Dashboard
KPIs + Charts + Insights
Render Dashboard
👤 User
🌐 Browser
🖥️ Flask
🤖 LLM
💾 Data

Tech Stack

Backend

Technology	Purpose	Version
Python	Core programming language	3.8+
Flask	Web framework	3.0.0
Werkzeug	WSGI utilities	3.0.1
Requests	HTTP client	2.31.0
python-dotenv	Environment management	1.0.0

Document Processing

Technology	Purpose	Version
PyPDF2	PDF parsing	3.0.1
python-docx	DOCX parsing	1.1.0
BeautifulSoup4	HTML/XML parsing	4.12.2

Data Processing

Technology	Purpose	Version
openpyxl	Excel file processing	3.1.2
CSV (stdlib)	CSV file processing	Built-in
JSON (stdlib)	JSON processing	Built-in

Audio Processing

Technology	Purpose	Version
SpeechRecognition	Voice input	3.10.0

Technology	Purpose	Version
PyAudio	Audio I/O	0.2.14
pydub	Audio manipulation	0.25.1

Frontend

Technology	Purpose
HTML5	Structure
CSS3	Styling (Custom dark theme)
JavaScript (ES6+)	Interactivity
Chart.js	Data visualization
Highlight.js	Code syntax highlighting
Marked.js	Markdown parsing

External Services

Service	Purpose
Ollama	Local LLM inference
Google Search	Web search integration
ElevenLabs	Text-to-speech (optional)

Installation

Prerequisites

- Python 3.8 or higher
- pip (Python package manager)
- Ollama (for local LLM) or compatible LLM API
- Modern web browser

Step-by-Step Installation

1. **Clone the Repository**

   ```
   git clone https://github.com/perfionix-ai/axio.git
   cd axio
   ```

2. **Create Virtual Environment (Recommended)**

   ```
   python -m venv venv

   # Windows
   venv\Scripts\activate

   # Linux/Mac
   source venv/bin/activate
   ```

3. **Install Dependencies**

   ```
   pip install -r requirements.txt
   ```

4. **Configure Environment**

   ```
   # Create .env file
   cp .env.example .env

   # Edit .env with your settings
   GPT_SERVER_URL=http://localhost:11434/api/chat
   GPT_MODEL=llama3.2
   ELEVENLABS_API_KEY=your_key_here  # Optional
   ```

5. **Start Ollama (if using local LLM)**

   ```
   ollama serve
   ollama pull llama3.2
   ```

6. **Run the Application**

   ```
   python app.py
   ```

7. **Access Axio**

   ```
   Open browser: http://localhost:5000
   ```

Project Structure

```
axio/
├── app.py                      # Main Flask application
├── requirements.txt            # Python dependencies
├── .env                        # Environment configuration
├── README.md                   # Documentation
│
├── templates/
│   └── index.html              # Main HTML template
│
├── static/
│   ├── style.css               # Custom CSS styles
│   ├── script.js               # JavaScript logic
│   └── logo gen .png           # Application logo
│
└── uploads/                    # Temporary file uploads (auto-created)
```

Module Details

AI Chat Module

```
# Key Functions
- chat(): Handle chat messages
- format_response(): Format AI responses with markdown
- Web search integration with fallback
- Streaming response support
```

Features:

- Multi-turn conversations
- Context management (15 messages)
- Code syntax highlighting
- Web search for real-time data
- Message editing capability

DocIQ Module

```
# Key Functions
- dociq_upload(): Handle document uploads
- dociq_chat(): Process document queries
- extract_text_from_pdf(): PDF text extraction
- extract_text_from_docx(): DOCX text extraction
- chunk_text(): Split text for RAG
- find_relevant_chunks(): Retrieve relevant context
```

Supported Formats:

- PDF (.pdf)
- Word Documents (.docx)
- Text Files (.txt)
- Markdown Files (.md)

VizIQ Module

```
# Key Functions
- viziq_upload(): Handle data file uploads
- parse_csv_data(): CSV parsing
- parse_excel_data(): Excel parsing
- detect_column_types(): Auto-detect data types
- calculate_statistics(): Compute statistics
- generate_kpis(): Generate KPI cards
- generate_chart_configs(): Create chart configurations
- generate_insights(): AI-powered insights
```

Chart Types Generated:

1. Column Chart - Categorical data visualization
2. Distribution Chart - Data distribution (Doughnut)
3. Comparison Chart - Metric comparisons
4. Trend Chart - Time series analysis

API Endpoints

Chat API

Endpoint	Method	Description
`/api/chat`	POST	Send chat message
`/api/chat/clear`	POST	Clear chat history

DocIQ API

Endpoint	Method	Description
`/api/dociq/upload`	POST	Upload document
`/api/dociq/chat`	POST	Query documents
`/api/dociq/clear`	POST	Clear documents

VizIQ API

Endpoint	Method	Description
`/api/viziq/upload`	POST	Upload data file
`/api/viziq/clear`	POST	Clear data

Task Management API

Endpoint	Method	Description
`/api/tasks`	GET	Get all tasks
`/api/tasks`	POST	Create task
`/api/tasks/<id>`	PUT	Update task
`/api/tasks/<id>`	DELETE	Delete task

Notes API

Endpoint	Method	Description
`/api/notes`	GET	Get all notes
`/api/notes`	POST	Create note
`/api/notes/<id>`	DELETE	Delete note

Reminders API

Endpoint	Method	Description
`/api/reminders`	GET	Get all reminders
`/api/reminders`	POST	Create reminder
`/api/reminders/<id>`	DELETE	Delete reminder

Screenshots

Dashboard Layout

````
+------------------------------------------------------------------+
|  [Logo]  AXIO                    [Chat] [Tasks] [Notes] [DocIQ]  |
+------------------------------------------------------------------+
|                                                                  |
|  Welcome to Axio! How can I help you today?                      |
|                                                                  |
|  +------------------------------------------------------------+  |
|  |  User: How do I create a REST API in Python?               |  |
|  +------------------------------------------------------------+  |
|  |  Axio: Here's how to create a REST API using Flask:        |  |
|  |                                                            |  |
|  |  ```python                                                 |  |
|  |  from flask import Flask, jsonify                          |  |
|  |  app = Flask(__name__)                                     |  |
|  |                                                            |  |
|  |  @app.route('/api/data')                                   |  |
|  |  def get_data():                                           |  |
|  |      return jsonify({"message": "Hello!"})                 |  |
|  |  ```                                                       |  |
|  +------------------------------------------------------------+  |
|                                                                  |
|  [____________________Type your message..._____________________]  |
+------------------------------------------------------------------+
````

VizIQ Dashboard

```
  +-------------------------------------------------------------------+
  |  Sales Analytics Dashboard                                         |
  |  AI-generated insights from sales_data.csv                        |
  |  [1,234 Rows] [8 Columns] [Updated: Now]                          |
  +-------------------------------------------------------------------+
  |  [Total Revenue]  [Avg Order]  [Total Orders]  [Growth Rate]      |
  |  [$1.2M]          [$156]       [7,689]         [+12.5%]           |
  +-------------------------------------------------------------------+
  |  [Column Chart]  [Distribution]  [Comparison]                      |
  |  Revenue by      Product Mix     Metrics                           |
  |  Category                        Comparison                        |
  +-------------------------------------------------------------------+
  |  [              Trend Analysis: Revenue Over Time                ] |
  +-------------------------------------------------------------------+
  |  [Insight 1]  [Insight 2]  [Insight 3]  [Insight 4]               |
  +-------------------------------------------------------------------+
```

Future Roadmap

Version 2.1

- ☐ Database integration (SQLite/PostgreSQL)
- ☐ User authentication system
- ☐ Export dashboards as PDF/PNG

Version 2.2

- ☐ Voice input (speech-to-text)
- ☐ Multi-language support
- ☐ Calendar integration

Version 3.0

- ☐ Mobile responsive PWA
- ☐ Real-time collaboration
- ☐ Plugin architecture

- □ Custom AI model fine-tuning

Contributing

We welcome contributions! Please follow these steps:

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

Code Style

- Follow PEP 8 for Python code
- Use meaningful variable names
- Add comments for complex logic
- Write unit tests for new features

License

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

Contact & Support

Perfionix AI

- Website: perfionix.ai
- Email: support@perfionix.ai
- GitHub: github.com/perfionix-ai

Built with by Perfionix AI

Empowering productivity through intelligent automation

Version: 2.0.0
Last Updated: December 2024
Status: Active Development