MentorOS

Mentoros AI Assistant

Integration & Setup Manual

Version 1.0 • December 2024

Table of Contents

1. Overview 3

1.1 Features 3

2. Quick Start Guide 4

2.1 Prerequisites 4

2.2 SDK Installation 4

2.3 Initialization 5

3. Configuration Options 7

4. Registering Tasks (Actions) 9

4.1 Task System Overview 9

4.2 How to Register Tasks 10

4.3 LLM Instructions (Contact Us) 12

5. Merchant Dashboard 13

5.1 Dashboard Overview 13

5.2 Widget Customization 14

5.3 API Keys Management 15

5.4 Data Management 16

5.5 Analytics Integrations 17

6. Events & Callbacks 18

7. Support & Contact 19

1. Overview

Mentoros is a powerful AI assistant SDK designed for seamless integration into e-commerce websites. Built on advanced large language model technology fine-tuned specifically for e-commerce, Mentoros transforms the online shopping experience by providing intelligent product discovery, personalized recommendations, instant customer support, and guided checkout assistance. More than just a chatbot—it's your store's AI-powered sales and support team.

1.1 Features

🛡️ Security First

AES-256-GCM encryption, JWT authentication, rate limiting, XSS prevention

⚡ Lightweight

~12KB gzipped vanilla JavaScript implementation

🔌 Easy Integration

Simple script tag inclusion with minimal configuration

🤖 AI Powered

Advanced LLM specialized for your product catalog with context-aware responses

🎨 Customizable

Flexible styling via dashboard or CSS variables

🌐 Cross-Origin Safe

CORS protection and secure communication

2. Quick Start Guide

2.1 Prerequisites

Before integrating the Mentoros SDK, ensure you have:

- **Client Token:** An SDK API key from the Mentoros Merchant Dashboard
- **App ID:** Your unique tenant/application identifier
- **Base URL:** The Mentoros API endpoint (provided during onboarding)

ℹ️ **Getting Your Credentials**

Your `clientToken` and `appId` will be provided during the onboarding process. Contact the Mentoros team to receive your production credentials.

2.2 SDK Installation

Option 1: CDN (Recommended for Production)

Add the following tags to your HTML page, preferably just before the closing `</body>` tag:

```
<!-- Include the CSS in your <head> section -->
<link rel="stylesheet"
      href="https://sdk.mentoros.gr/latest/chatbot.min.css">

<!-- Include the JavaScript before </body> -->
<script src="https://sdk.mentoros.gr/latest/chatbot.min.js"></script>
```

Option 2: Development Build

For testing and development, you can use the development build:

```
<link rel="stylesheet"
      href="https://sdk.mentoros.gr/dev/chatbot.min.css">

<script src="https://sdk.mentoros.gr/dev/chatbot.min.js"></script>
```

> ⚠️ **Development Builds**
>
> Development builds have shorter cache times and may include experimental features. Use `/latest/` URLs for production.

2.3 Initialization

Initialize the chatbot after the SDK script has loaded:

```
// Create a container element for the chatbot
<div id="chatbot-container"></div>

<script>
  // Initialize the Mentoros Chatbot
  const chatbot = new SecureLightChatbot({
    // Required Configuration
    clientToken: 'your-sdk-api-key',
    appId: 'your-tenant-id',
    baseUrl: 'https://api.mentoros.gr',

    // Optional Configuration
    encrypted: true,
    debug: false,
    maxRequestsPerMinute: 10,
    placeholder: 'Ask me anything...',

    // Task handling configuration
    tasks: {
      autoExecute: true,
      confirmation: 'builtin'
    }
  });

  // Initialize with the container selector
  chatbot.init('#chatbot-container');
</script>
```

Complete Integration Example

Here's a complete HTML page with the Mentoros chatbot integrated:

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>My E-commerce Store</title>

    <!-- Mentoros Chatbot CSS -->
    <link rel="stylesheet"
          href="https://sdk.mentoros.gr/latest/chatbot.min.css">
</head>
<body>
    <!-- Your website content here -->

    <!-- Chatbot container (required) -->
    <div id="chatbot-container"></div>

    <!-- Mentoros Chatbot SDK -->
    <script src="https://sdk.mentoros.gr/latest/chatbot.min.js"></script>

    <script>
      // Initialize chatbot when page loads
      document.addEventListener('DOMContentLoaded', function() {
        const chatbot = new SecureLightChatbot({
          clientToken: 'mk_your_api_key_here',
          appId: 'your-store-name',
          baseUrl: 'https://api.mentoros.gr',
          encrypted: true
        });

        chatbot.init('#chatbot-container');
      });
    </script>
</body>
</html>
```

3. Configuration Options

The `SecureLightChatbot` constructor accepts a configuration object with the following options:

Required Parameters

Parameter	Type	Description
`clientToken`	string	Your SDK API key from the Merchant Dashboard
`appId`	string	Your unique tenant/application identifier
`baseUrl`	string	The Mentoros API endpoint URL

Optional Parameters

Parameter	Type	Default	Description
`encrypted`	boolean	true	Enable end-to-end encryption
`debug`	boolean	true	Enable debug logging in console
`maxRequestsPerMinute`	number	10	Rate limit for API requests
`maxMessageLength`	number	1000	Maximum characters per message
`maxHistoryLength`	number	20	Messages to keep in context
`placeholder`	string	"Type your message..."	Input field placeholder text
`trackCartInteractions`	boolean	true	Track add-to-cart events for analytics

Task Configuration

The `tasks` object controls how frontend tasks (actions) are handled:

Parameter	Type	Default	Description
`autoExecute`	boolean	true	Auto-execute tasks without confirmation
`confirmation`	string	"event"	Confirmation mode: "builtin", "event", or "none"
`stopOnError`	boolean	false	Stop processing tasks on first error

4. Registering Tasks (Actions)

4.1 Task System Overview

Tasks are actions that the AI can request to be executed on your website. This enables the chatbot to interact with your e-commerce functionality, such as:

- **cart.addItem** - Add products to the shopping cart
- **order.checkout** - Navigate to the checkout page
- **product.view** - Navigate to a product page
- **Custom actions** - Any action specific to your store

ℹ️ **Two-Part Task Setup**

Setting up tasks requires **two steps**:

1. **Frontend:** Register task handlers in your JavaScript code
2. **Backend:** Configure LLM instructions (contact Mentoros team)

4.2 How to Register Tasks

Use the `registerTask()` method to register handlers for tasks:

```
// Register a task to handle checkout navigation
chatbot.registerTask('order.checkout', async (params, ctx) => {
  // Navigate to your checkout page
  window.location.href = '/checkout';

  // Return success status
  return { success: true, message: 'Navigating to checkout' };
}, { aliases: ['checkout', 'sdk.checkout'] });
```

Add to Cart Example

```
// Register a task to add items to cart
chatbot.registerTask('cart.addItem', async ({ sku, qty = 1 }, ctx) => {
  if (!sku) {
    return { success: false, message: 'Missing product SKU' };
  }

  try {
    // Call your cart API or function
    await yourCartService.addItem(sku, qty);

    // Update UI (cart badge, etc.)
    updateCartUI();

    return {
      success: true,
      message: `Added ${qty} item(s) to cart`
    };
  } catch (error) {
    return {
      success: false,
      message: 'Failed to add item to cart'
    };
  }
});
```

Register Multiple Tasks

```
// Register multiple tasks at once
chatbot.registerTasks({
  'cart.addItem': async ({ sku, qty }) => {
    await addToCart(sku, qty);
    return { success: true };
  },

  'cart.removeItem': async ({ sku }) => {
    await removeFromCart(sku);
    return { success: true };
  },

  'order.checkout': async () => {
    navigateToCheckout();
    return { success: true };
  }
});
```

Task Handler Context

Each task handler receives two arguments:

Argument	Description
`params`	Object containing task parameters (e.g., `{ sku: "ABC123", qty: 2 }`)
`ctx`	Context object with `conversationId`, `messageId`, etc.

4.3 LLM Instructions (Contact Us)

For the AI to know *when* to trigger your registered tasks, the LLM instructions must be configured on the backend. This involves:

1. **Define Your Actions**

   List the actions you want the chatbot to perform (e.g., add to cart, checkout, apply discount codes)

2. **Contact Mentoros Team**

   Email **support@mentoros.gr** or your account manager with your list of actions and expected behavior

3. **We Configure the LLM**

   Our team will update the AI's instructions to recognize user intents and trigger the appropriate tasks

4. **Test & Iterate**

   Test the integration and provide feedback for fine-tuning the AI responses

💡 **Example Request Email**

Subject: Task Configuration for [Your Store Name]

Hi Mentoros Team,

Please configure the following tasks for our chatbot:

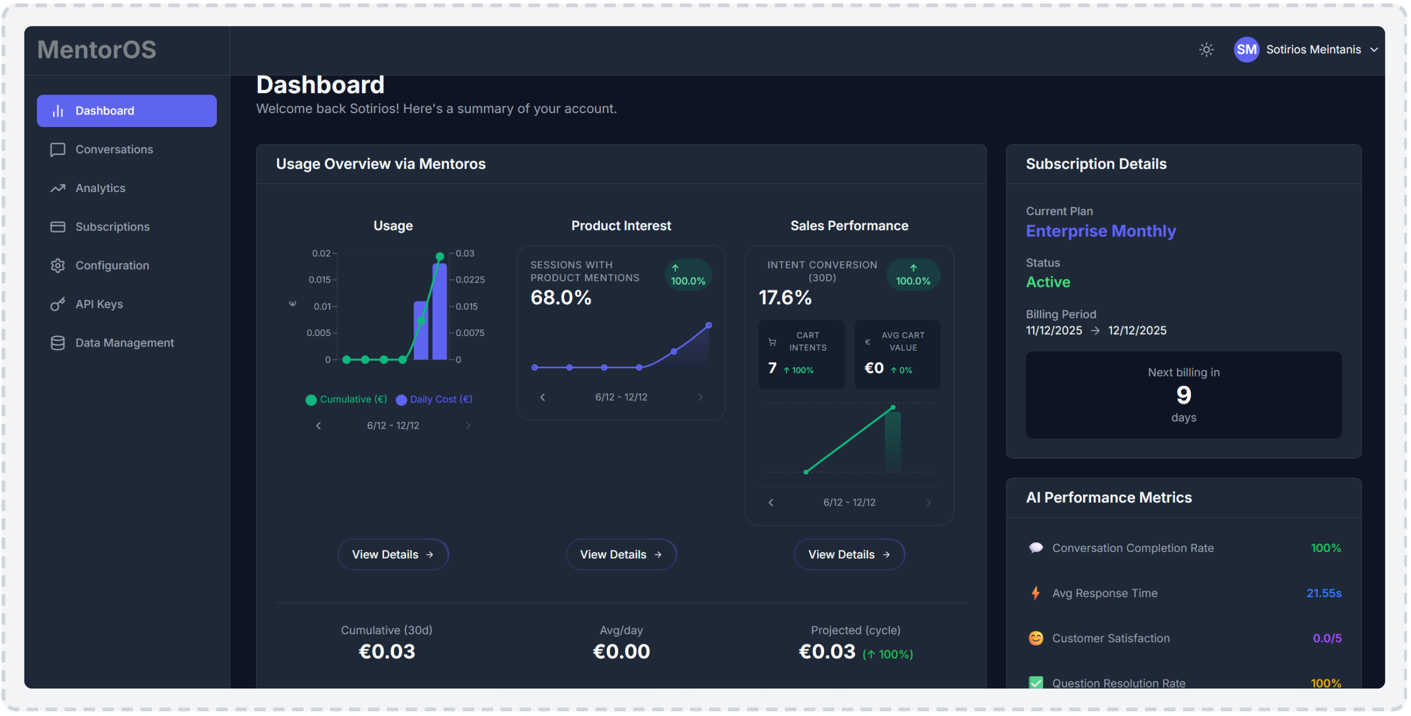
- **cart.addItem** - When user wants to add a product to cart
- **order.checkout** - When user says "checkout" or "buy now"

5. Merchant Dashboard

The Mentoros Merchant Dashboard provides a web interface for managing your chatbot configuration, monitoring performance, and viewing analytics.

5.1 Dashboard Overview

The main **Dashboard** provides an at-a-glance view of your chatbot's performance:



Dashboard

Welcome back Sotirios! Here's a summary of your account.

Usage Overview via Mentoros

Usage

Cumulative (€) Daily Cost (€)

6/12 - 12/12

Product Interest

SESSIONS WITH PRODUCT MENTIONS 68.0% ↑ 100.0%

6/12 - 12/12

Sales Performance

INTENT CONVERSION (30D) 17.6% ↑ 100.0%

CART INTENTS 7 ↑ 100%

AVG CART VALUE €0 ↑ 0%

6/12 - 12/12

View Details → View Details → View Details →

Cumulative (30d) €0.03

Avg/day €0.00

Projected (cycle) €0.03 (↑ 100%)

Subscription Details

Current Plan: Enterprise Monthly

Status: Active

Billing Period: 11/12/2025 → 12/12/2025

Next billing in 9 days

AI Performance Metrics

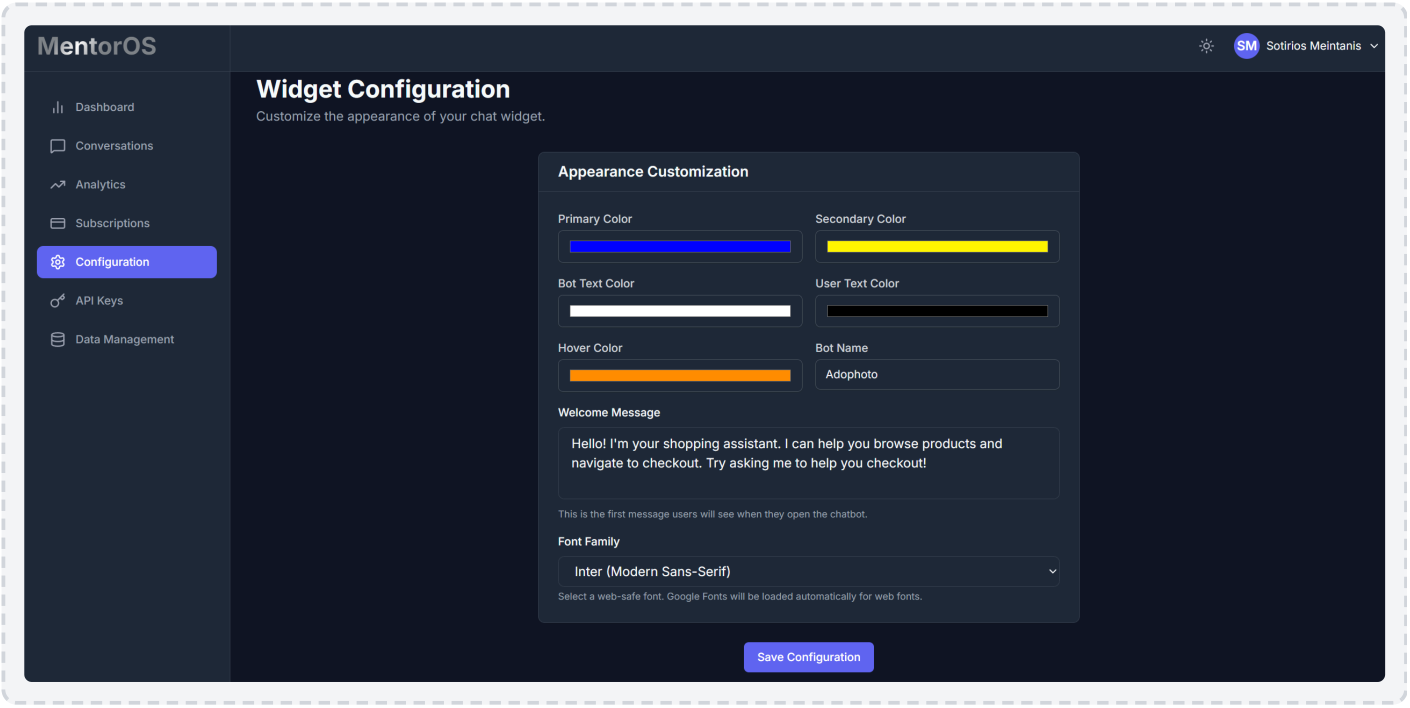
Conversation Completion Rate	100%
Avg Response Time	21.55s
Customer Satisfaction	0.0/5
Question Resolution Rate	100%

Key Metrics

- **Usage Summary:** Messages used vs. plan limits
- **Conversations:** Total chat sessions over time
- **Cart Value KPI:** Revenue influenced by chatbot interactions
- **Sales Performance:** Conversion metrics and trends
- **Response Times:** Average AI response latency

5.2 Widget Customization

Navigate to **Configuration** in the dashboard to customize your chatbot's appearance:



Widget Configuration

Customize the appearance of your chat widget.

Appearance Customization

Primary Color: 🟦

Secondary Color: 🟨

Bot Text Color: ⬜

User Text Color: ⬛

Hover Color: 🟧

Bot Name: Adophoto

Welcome Message: Hello! I'm your shopping assistant. I can help you browse products and navigate to checkout. Try asking me to help you checkout!

This is the first message users will see when they open the chatbot.

Font Family: Inter (Modern Sans-Serif)

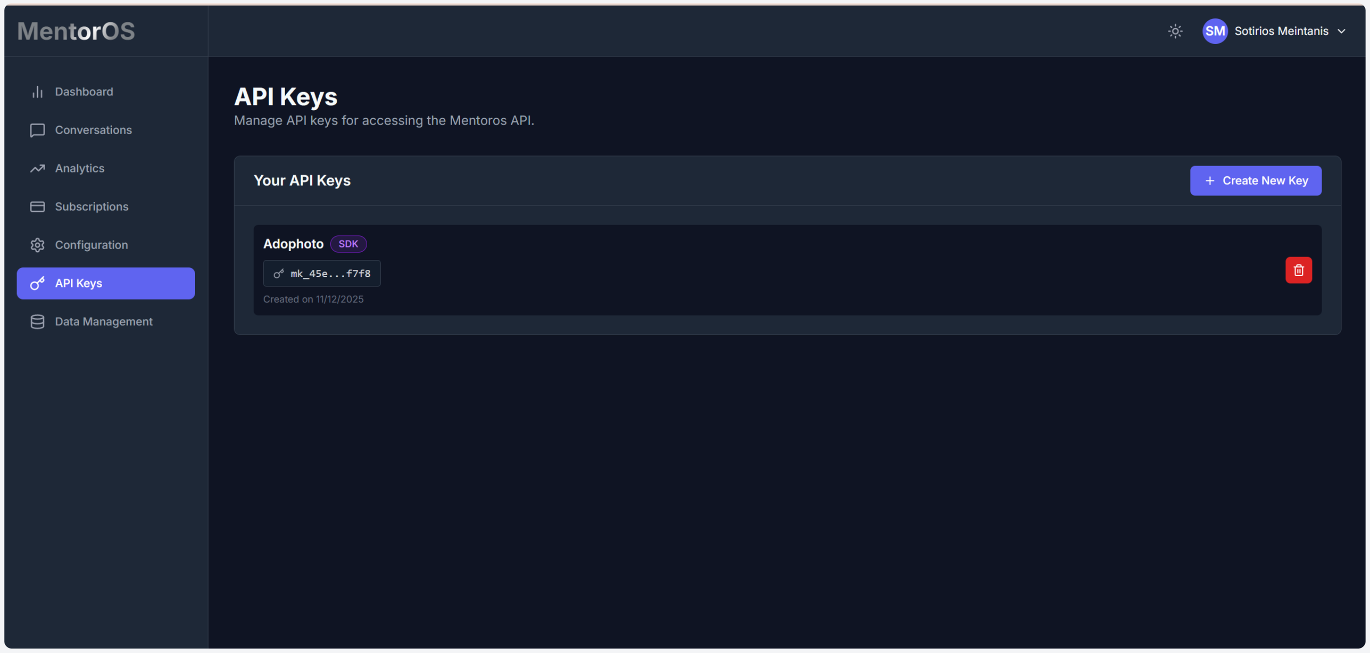
Select a web-safe font. Google Fonts will be loaded automatically for web fonts.

Customizable Properties

Property	Description	Default
Primary Color	Main brand color (header, AI messages, send button)	#6366F1
Secondary Color	User message bubbles	#1F2937
Bot Name	Display name in chat header	Mentoros
Welcome Message	Initial greeting message	Custom greeting
Font Family	Typography for the widget	Inter
Hover Color	Interactive element hover states	#ff8c00

5.3 API Keys Management

Navigate to **API Keys** to create and manage your SDK keys:



API Keys
Manage API keys for accessing the Mentoros API.

Your API Keys

Adophoto SDK
mk_45e...f7f8
Created on 11/12/2025

API Key Types

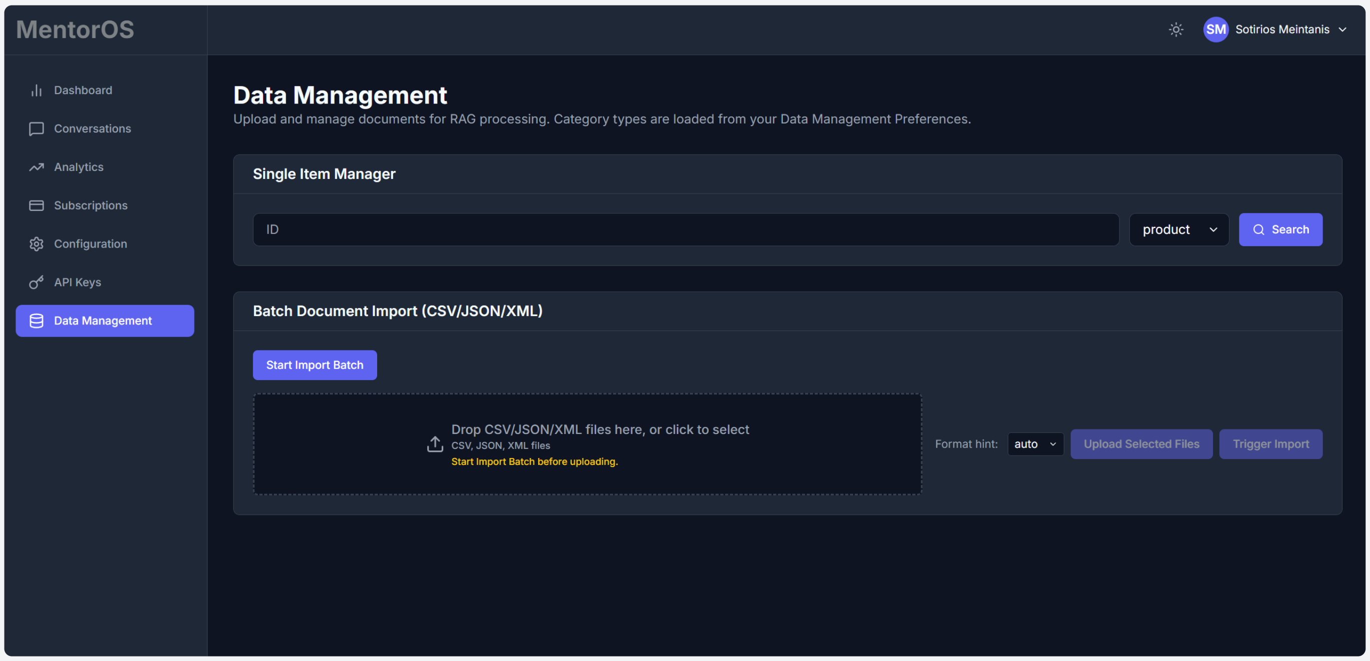
Type	Use Case
SDK Keys	For the hosted chatbot widget on your website
Integration Keys	For backend API integrations (binds to your user account)

⚠️ Security Note

API keys are shown only once when created. Store them securely. If you lose a key, you'll need to create a new one.

5.4 Data Management

The **Data Management** page allows you to manage your product catalog and other data that the AI uses:



Data Management

Upload and manage documents for RAG processing. Category types are loaded from your Data Management Preferences.

Single Item Manager

Batch Document Import (CSV/JSON/XML)

Drop CSV/JSON/XML files here, or click to select

CSV, JSON, XML files

Start Import Batch before uploading.

Format hint: auto

Features

- **Bulk Import:** Upload product data via CSV, JSON, or XML files
- **Search & Edit:** Find and modify individual items by ID
- **Category Types:** Manage Products, Categories, and custom types
- **JSON Editor:** Direct JSON editing for advanced users
- **Schema Management:** Define custom fields and data structure

File Format Specifications

When importing data, files must follow these formats. The `id` field is **required** for all records.

Common Fields (All Formats)

Field	Required	Description
`id`	Yes	Unique identifier (also accepts: `document_id`, `uuid`, `_id`)
`type`	No	Document type (e.g., "product", "category"). Default: "record"
`action`	No	Operation: `upsert` (default), `delete`, `create`, `update`
`text`	No	Main text content (also: `content`, `body`, `description`)

CSV Format

Use header prefixes to separate indexed (searchable) vs. regular data fields:

- `i:fieldname` → Indexed property (searchable by AI)
- `d:fieldname` → Data property (stored but not indexed)
- No prefix → Defaults to data property

```
id,type,action,i:title,i:description,d:sku,price
prod-1,product,upsert,Widget A,"Great item",A100,29.99
prod-2,product,delete,,,,
prod-3,product,upsert,Widget C,"Nice one",A102,39.99
```

JSON Format

Accepts array of objects, or object with `items` / `documents` key. Use `index` and `data` objects to separate searchable vs. stored fields:

```
{
  "documents": [
    {
      "id": "prod-1",
      "type": "product",
      "action": "upsert",
      "index": {
        "title": "Widget A",
        "description": "Great item"
      },
      "data": {
        "sku": "A100",
        "price": 29.99
      }
    },
    {
      "id": "prod-2",
      "type": "product",
      "action": "delete"
    }
  ]
}
```

XML Format

Use `<index>` and `<data>` child elements for field separation:

```
<documents>
  <doc>
    <id>prod-1</id>
    <type>product</type>
    <action>upsert</action>
    <index>
      <title>Widget A</title>
      <description>Great item</description>
    </index>
    <data>
      <sku>A100</sku>
      <price>29.99</price>
    </data>
  </doc>
  <doc>
    <id>prod-2</id>
    <type>product</type>
    <action>delete</action>
  </doc>
</documents>
```

💡 **Index vs Data Fields**

Index fields are searchable by the AI and used for product discovery. Include product names, descriptions, categories, and key attributes here.

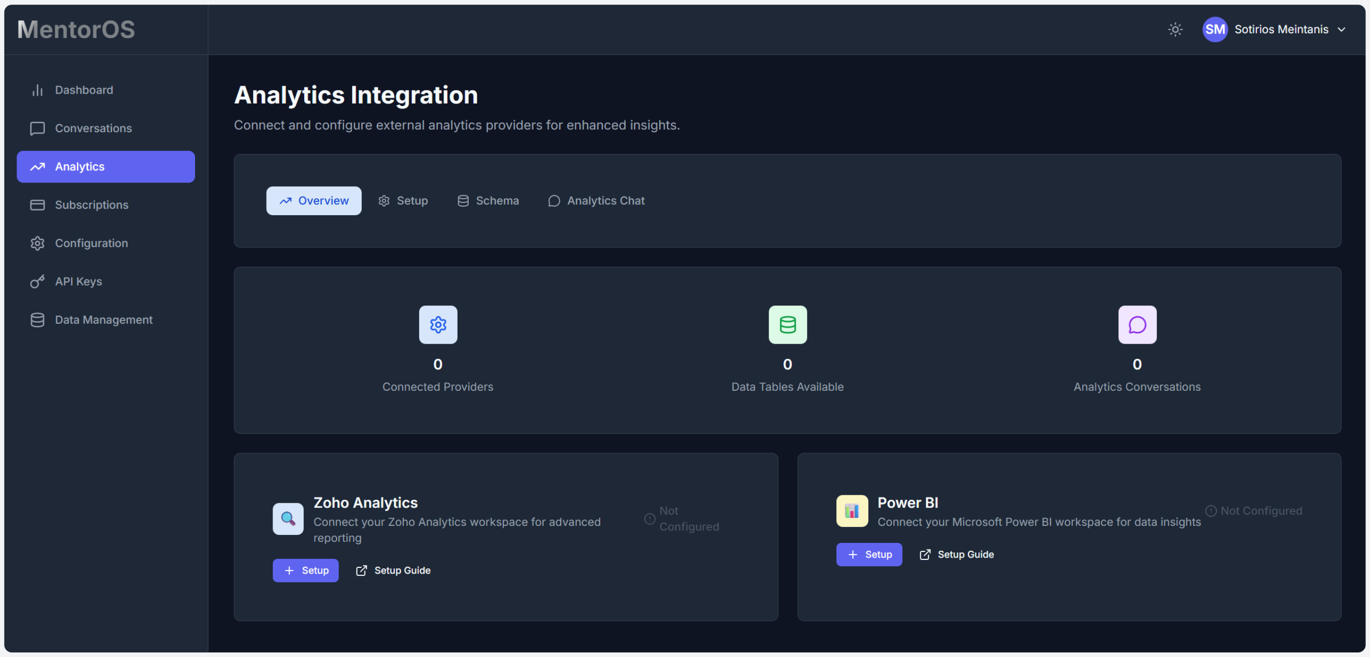
Data fields are stored but not directly searchable. Use for SKUs, internal IDs, prices, and metadata.

⚠️ **Important Notes**

- Records without an `id` field are skipped
- For `delete` action, only the `id` is required
- Actions are case-insensitive (`Upsert` , `DELETE` both work)
- Files can contain mixed operations (upsert and delete together)

5.5 Analytics Integrations

The **Analytics** page enables integration with external analytics providers for advanced reporting:



Analytics Integration

Connect and configure external analytics providers for enhanced insights.

0 Connected Providers

0 Data Tables Available

0 Analytics Conversations

Zoho Analytics — Connect your Zoho Analytics workspace for advanced reporting — Not Configured

Power BI — Connect your Microsoft Power BI workspace for data insights — Not Configured

Supported Providers

- **Zoho Analytics:** Connect your Zoho account for custom reports
- **Power BI:** Microsoft Power BI integration

Features

- **Schema Explorer:** Browse connected data tables
- **Analytics Chat:** Ask questions about your data in natural language
- **OAuth Setup:** Secure authorization with providers

6. Events & Callbacks

Subscribe to SDK events for custom integrations:

```
// Listen for when the chatbot is ready
chatbot.on('ready', ({ instance }) => {
  console.log('Chatbot initialized successfully');
});

// Listen for user messages
chatbot.on('message:sent', (message) => {
  console.log('User sent:', message.content);
});

// Listen for AI responses
chatbot.on('message:received', (message) => {
  console.log('AI responded:', message.content);
});

// Listen for errors
chatbot.on('error', (error) => {
  console.error('Chatbot error:', error);
});

// Listen for task execution
chatbot.on('task:completed', ({ task, result }) => {
  console.log('Task completed:', task.id, result);
});

// Listen for chat open/close
chatbot.on('open', () => console.log('Chat opened'));
chatbot.on('close', () => console.log('Chat closed'));
```

Available Events

Event	Description
`ready`	SDK initialized and ready
`open`	Chat window opened
`close`	Chat window closed
`message:sent`	User sent a message
`message:received`	AI response received
`error`	An error occurred
`task:received`	Task received from AI
`task:started`	Task execution started
`task:completed`	Task completed successfully
`task:failed`	Task execution failed
`chat:reset`	Chat history was cleared

7. Support & Contact

We're here to help you integrate Mentoros successfully!

Contact Information

📧 **Email**	support@mentoros.gr

What We Need for LLM Task Configuration

When contacting us to set up custom tasks, please provide:

1. **Task ID:** The identifier you use in `registerTask()`
2. **Description:** What the action should do
3. **Trigger Phrases:** Example user messages that should trigger this action
4. **Parameters:** What data the AI should extract (e.g., product SKU, quantity)
5. **Success Response:** What the AI should say after success

💡 **Quick Reference**

CDN URL: https://sdk.mentoros.gr/latest/chatbot.min.js

API Endpoint: https://api.mentoros.gr

Dashboard: https://www.mentoros.gr

Version 1.0 • December 2024