Copy page

AI Gateway

AI Gateway is available in Beta on all plans and your use is subject to Vercel's Public Beta Agreement and AI Product Terms.

The AI Gateway provides a unified API to access 100+ models through a single endpoint. It gives you the ability to set budgets, monitor usage, load-balance requests, and manage fallbacks.

The design allows it to work seamlessly with AI SDK 5, OpenAI SDK, or your preferred framework.

Some of the key features of the AI Gateway include:

- Unified API: helps you switch between providers and models with minimal code changes
- High Reliability: automatically retries requests to other providers if one fails
- Embeddings Support: generate vector embeddings for search, retrieval, and other tasks
- Spend Monitoring: monitor your spending across different providers

Getting started

Create an API key

First, let's create an API key in the AI Gateway tab of the Vercel Dashboard:

1. From the Vercel dashboard, click the AI Gateway tab
2. Click API keys on the left side bar
3. Click Create key and proceed with Create key from the Dialog

Once you have the API key, you'll need to provide it to your application, which we will do in the next section.

This guide uses API keys. You can also use OIDC tokens to authenticate your requests.

Using AI SDK

The AI SDK is a Typescript library that helps developers build AI-powered applications. In the below quickstart, you'll build a simple AI-chatbot with a streaming user interface.

Prerequisites

To follow this quickstart, you'll need:

- Node.js 18+ and `pnpm` installed on your local development machine.
- An AI Gateway API key (created in the previous step).

Setup your application

Start by creating a new directory using the mkdir command. Change into your new directory and then run the pnpm init command. This will create a package.json in your new directory.

_terminal

```
mkdir demo
cd demo
pnpm init
```

Install Dependencies

Install `ai` , along with other necessary dependencies.

The AI SDK is designed to be a unified interface to interact with any large language model. This means that you can change model and providers with just one line of code! Learn more about available providers and building custom providers in the providers section.

_terminal

```
pnpm add ai zod dotenv
pnpm add -D @types/node tsx typescript
```

The `ai` package contains the AI SDK. You will use `zod` to define type-safe schemas that you will pass to the large language model (LLM). You will use `dotenv` to access environment variables (your AI Gateway API key) within your application. There are also three development dependencies, installed with the `-D` flag, that are necessary to run your Typescript code.

Configure AI Gateway API key

Create a `.env` file in your project's root directory and add your AI Gateway API Key. This key is used to authenticate your application with the AI Gateway service.

_terminal

```
touch .env
```

Edit the .env file:

.env

```
AI_GATEWAY_API_KEY=your_ai_gateway_api_key
```

Replace `your_ai_gateway_api_key` with your actual AI Gateway API key from the previous step.

The AI Gateway provider will default to using the `AI_GATEWAY_API_KEY` environment variable.

On this page

Create your application

Create an `index.ts` file in the root of your project and add the following code:

Anthropic OpenAI Google Grok

index.ts

```ts
import { ModelMessage, streamText } from 'ai';
import 'dotenv/config';
import * as readline from 'node:readline/promises';

const terminal = readline.createInterface({
input: process.stdin,
output: process.stdout,
});

const messages: ModelMessage[] = [];

async function main() {
while (true) {
const userInput = await terminal.question('You: ');

  messages.push({ role: 'user', content: userInput });

  const result = streamText({
    model: 'anthropic/claude-opus-4-20250514',
    messages,
  });

  let fullResponse = '';
  process.stdout.write('\nAssistant: ');
  for await (const delta of result.textStream) {
    fullResponse += delta;
    process.stdout.write(delta);
  }
  process.stdout.write('\n\n');

  messages.push({ role: 'assistant', content: fullResponse });

}
}

main().catch(console.error);
```

Run your application

Now you can run your application:

_terminal

```
pnpm tsx index.ts
```

You should see a prompt where you can start chatting with the AI. The responses will be streamed in real-time!

Next steps

Continue with the AI SDK documentation to learn advanced configuration, set up provider routing and fallbacks, and explore more integration examples.

Using OpenAI SDK

The AI Gateway provides OpenAI-compatible API endpoints that allow you to use existing OpenAI client libraries and tools with the AI Gateway.

The OpenAI-compatible API includes:

- Model Management: List and retrieve the available models
- Chat Completions: Create chat completions that support streaming, images, and file attachments
- Tool Calls: Call functions with automatic or explicit tool selection
- Existing Tool Integration: Use your existing OpenAI client libraries and tools without needing modifications

Learn more about using the OpenAI SDK with the AI Gateway in the OpenAI-Compatible API page.

Using other community frameworks

The AI Gateway is designed to work with any framework that supports the OpenAI API or AI SDK 5.

Read more about using the AI Gateway with other community frameworks in the AI Gateway with community frameworks section.

Last updated on August 6, 2025