

<p>Activity Header</p> <ul style="list-style-type: none"> • Learning Item: Apply Hands-on Learning - Configure and Test a Secure API Connection • Aligned LO: LO1: Apply secure authentication methods to integrate external AI services within a no-code platform. • High-level Description: Learners You will perform the core job task of securely connecting an external AI service to a no-code app. This involves obtaining an API (Application Programming Interface) key, storing it safely in Bubble's secrets manager, and running a test call to confirm the connection is authenticated and functional. • Estimated Time: 10 minutes (2 minutes Setup, 6 minutes Core Activity, 2 minutes Documentation) • Individual Work: Solo completion, no collaboration required • Prerequisites: A Bubble.io account (free plan is sufficient) and access to the course-provided mock "AI Idea Generator" service documentation (including the API key). <p>Detailed Activity Design</p> <p>Scenario</p> <p>: As a no-code developer on a fast-moving product team, you have been tasked with integrating a new "AI Idea Generator" into your Bubble application. Before you can build any user-facing features, your first</p>	10 mins
---	---------

critical step is to establish a secure and working connection. Your goal is to get the system authenticated and ready for development, ensuring that the API key is never exposed.

Learning Objectives:

- Analyze the components of a secure API connection within a no-code platform.
- Apply best practices for API key management by utilizing a platform's secrets manager.
- Validate a successful API integration by initializing a test call and verifying the response.

Your Tasks (10 Minutes):

Setup (2 minutes): Log in to your Bubble.io account. Open the provided Bubble app template for this assignment. Review the one-page documentation for the mock "AI Idea Generator" service to locate your unique API key.

Core Activity Steps (6 minutes):

- **Store the API Key Securely:** Navigate to the "Settings" tab in your Bubble editor, go to the "API" section, and add a new private key. Name it **AI_Idea_Generator_Key** and paste the key from the documentation.
- **Configure the API Connector:** Go to the "Plugins" tab and open the API Connector. Create a new API call named **Get Idea**.

<p>Configure the call with the correct Method (POST) and URL from the mock service documentation.</p> <ul style="list-style-type: none"> ● Add Authentication: Add the required Content-Type and Authorization headers. For the Authorization header, use the Bearer prefix followed by the dynamic private key you just saved in the secrets manager. ● Run a Test Call: Add the simple JSON body provided in the mock service documentation and click "Initialize call." <p>Documentation (2 minutes): Take a screenshot of the successful API call initialization window showing the returned data. In a brief reflection (2–3 sentences), describe why using the secrets manager is a critical step in this process.</p>	<p>Submission Output</p> <p>: Upload a screenshot of the successful "Initialize call" window in Bubble. Submit a brief reflection (2–3 sentences) answering: "Why is storing the API key in the secrets manager, instead of pasting it directly into the API Connector's header, essential for application security?"</p> <p>Self-Assessment Checklist:</p> <ul style="list-style-type: none"> ● An API key for the "AI Idea Generator" service has been added to Bubble's secrets manager.
---	--

- The API Connector has been configured with the correct URL, method, and headers.
- The "Authorization" header correctly uses the private key from the secrets manager.
- The test call was successfully initialized, and a valid response was received.
- A screenshot of the successful test call has been captured.
- A brief reflection on the importance of the secrets manager is complete.

Your Tasks (10 Minutes):

1. **Setup (2 minutes):** Log in to your [Bubble.io](#) account. Open the provided Bubble app template for this assignment. Review the one-page documentation for the mock "AI Idea Generator" service to locate your unique API key.

The screenshot shows the OpenAI API keys management interface. A red box highlights the URL in the browser's address bar: `platform.openai.com/settings/organization/api-keys`. Another red box highlights the 'API keys' tab in the left sidebar. A third red box highlights the 'Copy' button in a modal window titled 'Save your key'. A fourth red box highlights the 'Create new secret key' button in the top right corner. The modal window contains instructions about saving the key securely and a note about OpenAI's security measures. It also includes a 'Learn more about API key best practices' link and a 'Permissions' section indicating 'Read and write API resources'. The main table lists two API keys: 'My Bubble App' (Active) and 'Bubble' (Active). The table columns are: ID, PROJECT ACCESS, CREATED BY, and PERM.

ID	PROJECT ACCESS	CREATED BY	PERM
Default project	Default project	Sasi T M	All
Default project	Default project	Sasi T M	All

The screenshot shows the 'Installed Plugins' section of the Bubble.io interface. A red box labeled '1: Click on Plugins' highlights the main list area. Another red box labeled '2' highlights the 'Add another API' button. The page displays three installed plugins:

- API Name: Hubspot
- API Name: HubSpot OAuth
- API Name: OpenAI Service

Below the list are two buttons: 'Add another API' and 'Uninstall this plugin'.

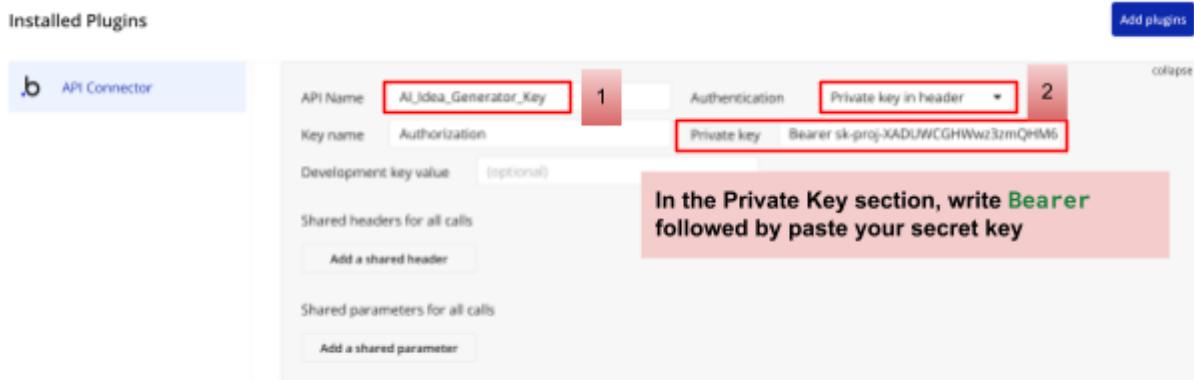
The screenshot shows the 'Install new plugins' search results. A red box labeled '3: Install the API connector' highlights the 'Uninstall' button for the API Connector plugin. The API Connector plugin card includes the following details:

- API Connector
- The API connector lets you define your own API calls directly in the Bubble Editor and use them in your app.
- By Bubble
- 1,395,403 apps
- Plugin page
- Uninstall

Other plugin cards visible include Google Material Icons and Ionic Elements, each with their respective details and 'Install' buttons.

Core Activity Steps (6 minutes):

- **Store the API Key Securely:** Navigate to the "Settings" tab in your Bubble editor, go to the "API" section, and add a new private key. Name it `AI_Idea_Generator_Key` and paste the key from the documentation.



- **Configure the API Connector:** Create a new API call named `Get Idea`. Configure the call with the correct Method (POST) and URL from the mock service documentation.

The screenshot shows the configuration for a new API call named `Get Idea`:

- Name:** `Get Idea`
- Method:** POST
- URL:** `https://api.openai.com/v1/responses`
- Headers:**
 - `Content-Type`: `application/json` (Private, checked, Optional, checked)
 - `Authorization`: `Bearer sk-proj-XADUWCG` (Private, checked, Optional, checked)
- Body type:** JSON
- Parameters:** Add parameter
- Body (JSON object):**

```

1 {
2   "model": "<model>",
3   "input": "<prompt>",
4   "stream": true
5 }
6
7 
```

[See reference](#)

Bubble API Connector Setup: (make sure to paste this to your API call setup)

API Call Name: Get Idea

Method: POST

API URL: <https://api.openai.com/v1/responses>

Headers:

Key: Content-Type Value: application/json

Key: Authorization Value: Bearer sk*****

Body Type: **JSON** (Paste this exactly):

```
{  
  "model": "<model>",  
  "input": "<prompt>",  
  "stream": true  
}
```

Body (JSON object, use <> for dynamic values)

```
1 {  
2   "model": "<model>",  
3   "input": "<prompt>",  
4   "stream": true  
5 }  
6  
7
```

Key	model	Value	gpt-5-nano	Private	<input checked="" type="checkbox"/>	Allow blank	<input type="checkbox"/>
Key	prompt	Value	"Tell me a joke"	Private	<input checked="" type="checkbox"/>	Allow blank	<input type="checkbox"/>

Key: model Value: gpt-5-nano

Key: prompt Value: "Tell me a joke"

Returned values - Get Idea

Returned Events

Final Step: Success

We detected 9 kinds of chunks returned in this response stream. You can choose how to handle them here.

- ▶ Event: response.created
- ▶ Event: response.in_progress
- ▶ Event: response.output_item.added
- ▶ Event: response.output_item.done
- ▶ Event: response.content_part.added
- ▶ Event: response.output_text.delta
- ▶ Event: response.output_text.done

SAVE

Cancel