hyder.ai API Documentation

Service Name: hyder.ai API
Base URL: https://backend.hyder.ai
Audience: Developers, Organizations, and Community Platforms
Version: 1.2
Status: Public Release
Date: September 2025

---

1) hyder.ai API — Terms of Use

The following Terms of Use govern all integrations with the hyder.ai API. By accessing or using the API, you agree to comply fully with these terms.

1.1. Free Service:
hyder.ai is provided as a **free religious service** for the Shia Ithna Asheri community.

1.2. Prohibition on Monetization:
Under no circumstances may the API or its responses be monetized, sold, licensed, or used as part of a paid product or service.

1.3. No Donations:
You are not permitted to collect or solicit donations on behalf of hyder.ai or in connection with the use of this API.

1.4. Attribution Requirement:
Any user interface or application that integrates the hyder.ai API must clearly and visibly display the phrase **"Powered by hyder.ai."** This attribution must appear in a reasonable location (e.g., footer, header, or adjacent to the response).

1.5. Integrity of Content:
The responses generated by Hyder.ai **must not be altered, censored, or repackaged** in a manner that misrepresents the Shia Ithna Asheri perspective. If responses are truncated, this must be clearly indicated to the end user.

1.6. Permitted Use:
The API may only be used for **religious, educational, or community benefit purposes.** Use in contexts that promote hate, mockery, or non-religious commercial exploitation is strictly prohibited.

1.7. Data Privacy:
You must not misuse, sell, or share user data (including emails, IDs, and session information). If you store logs or interaction data, you are responsible for ensuring they are secure and confidential.

1.8. Branding Consistency:
All integrations must maintain consistent branding, including proper attribution, and must not misrepresent hyder.ai as another service or product.

1.9. Conditional Access:
Access to the API is non-exclusive and may be revoked at any time, without notice, if these Terms of Use are violated.

Important: Failure to comply with any of the above clauses may result in **immediate suspension of access** to the hyder.ai API.

---

2) Integration Options

There are two ways to integrate **hyder.ai** into your app, website, or any other platform:

2.1. Simplest Option — Redirect Button

- Add a button labeled something like **"Ask a Question"** or **"Ask hyder.ai."**
- When clicked, the button redirects users to the hyder.ai platform.
- Users can then interact directly with the service.
- **Quickest and easiest** to set up.

2.2. Full Integration — Built-in Chat

For a seamless experience where everything stays within your own app or platform:

- You can integrate via the **hyder.ai API**.
- In this case, you'll need to **build your own user interface** (similar to a chat window, like ChatGPT or hyder.ai).
- This approach provides full control but requires **more time and development effort** on your side.

Note: Both integration methods require compliance with the Terms of Use.

---

3) hyder.ai — API Documentation

This doc shows how to integrate your web/app frontend with hyder.ai. It covers two simple HTTPS endpoints:

- POST /query — send a user's question and get back the AI answer
- POST /feedback *(optional)* — send thumbs up/down on a prior answer

Base URL: https://backend.hyder.ai

3.1. Ask a Question

Endpoint

POST https://backend.hyder.ai/query

Description

Submits a user's question and returns the AI-generated answer. You control sessions, user identity, and the text of the question.

Required Headers

Content-Type: application/json

Accept: application/json

Request Body

```
{
  "mode": "c",
  "query": "first imam name",
  "session_id": "fc4165af-713f-47b9-b021-d32001879993",
  "source_id": 112,
  "user_email": "user@example.com",
  "user_id": "12345",
  "user_name": "John Doe"
}
```

3.1.1. Field reference

Field	Type	Required	Notes
mode	string	Yes	Always "c" (conversation mode).
query	string	Yes	The end-user's question/prompt. UTF-8 text.
session_id	string	Yes	A client-generated stable ID (UUID recommended) that groups a user's turns into one session.
source_id	number	Yes	Always 112.
user_email	string	Yes	End-user's email.
user_id	string	Yes	Your internal user identifier.
user_name	string	Yes	End-user's display name.

3.1.2. Successful Response (200)


```
{
 "response": "Imam Ali ibn Abi Talib (a) is the first Imam.",
 "interaction_id": 8611,
 "question": "first imam name"
}
```


3.1.3. Response fields

Field	Type	Notes
response	string	The AI's answer text.
interaction_id	number	Unique ID for this Q&A interaction. Save this if you intend to send feedback later.
question	string	Echo of your question.

3.1.4. Error Responses

- **400 Bad Request** – malformed JSON or missing/invalid fields
- { "error": "Invalid request body: session_id is required" }
- **401/403** – if authentication is later enabled on your tenant (not required by default)
- **429 Too Many Requests** – if you exceed agreed limits
- **5xx** – transient server errors

3.1.5. cURL example

```
curl -X POST "https://backend.hyder.ai/query" \
  -H "Content-Type: application/json" \
  -d '{
    "mode": "c",
    "query": "first imam name",
    "session_id": "fc4165af-713f-47b9-b021-d32001879993",
    "source_id": 111,
    "user_email": "user@example.com",
    "user_id": "12345",
    "user_name": "John Doe"
  }'
```

3.1.6. JavaScript (fetch)

```
const res = await fetch("https://backend.hyder.ai/query", {
  method: "POST",
  headers: { "Content-Type": "application/json", "Accept": "application/json" },
  body: JSON.stringify({
    mode: "c",
    query: "first imam name",
    session_id: crypto.randomUUID(),
    source_id: 111,
    user_email: "user@example.com",
    user_id: "12345",
    user_name: "John Doe"
  })
});
const data = await res.json();
// data.response (answer), data.interaction_id (save for feedback)
```

3.1.7. Python (requests)

```
import requests, uuid
payload = {
  "mode": "c",
  "query": "first imam name",
  "session_id": str(uuid.uuid4()),
  "source_id": 111,
  "user_email": "user@example.com",
  "user_id": "12345",
  "user_name": "John Doe"
}
r = requests.post("https://backend.hyder.ai/query", json=payload, timeout=60)
r.raise_for_status()
data = r.json()
print(data["response"], data.get("interaction_id"))
```

3.2. Send Feedback (Optional)

Endpoint

POST https://backend.hyder.ai/feedback

Description

Attaches a thumbs up/down to a prior interaction.

Required Headers

Content-Type: application/json

Accept: application/json

Request Body

```
{
  "interaction_id": 8611,
  "feedback": -1
}
```

3.2.1. Field reference

Field	Type	Required	Notes
interaction_id	number	Yes	The ID returned by /query.
feedback	number	Yes	1 for thumbs up, -1 for thumbs down.

3.2.2. Successful Response (200)

{ "ok": true }

3.2.3. Error Responses

- **400 Bad Request** – missing/invalid fields
- { "error": "feedback must be 1 or -1" }
- **404 Not Found** – unknown interaction_id
- **5xx** – transient server errors

3.2.4. cURL example

```
curl -X POST "https://backend.hyder.ai/feedback" \
  -H "Content-Type: application/json" \
  -d '{ "interaction_id": 8611, "feedback": 1 }'
```

---

4) Implementation Notes & Best Practices

4.1. Sessions:

Use a stable session_id (UUID recommended) per conversation thread. New tabs or separate chats can use different session IDs.

4.2. Field constraints:

- mode must be "c".
- source_id must be 111.
- Provide real user_email, user_id, user_name if available for analytics/audit.

4.3. Timeouts & retries:

- Client timeout: 60–120s is usually sufficient.
- For 5xx or network timeouts, retry with exponential backoff.
- Do **not** retry feedback blindly if you're unsure it failed—feedback should be idempotent on your side (e.g., only send once per interaction per user action).

4.4. Error handling:

Always parse JSON responses. If the status is not 2xx, read the returned { "error": "..." } message to decide what to show the user.

4.5. Internationalization:

Send query in the user's language; responses are returned as plain text in the response field.

4.6. Telemetry:

Save interaction_id with your chat transcript so you can submit feedback later.
