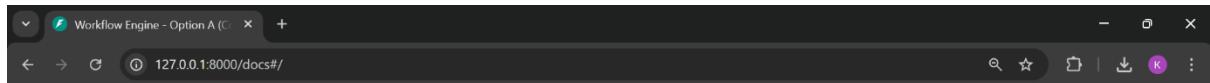


Workflow Engine – API Endpoints Test Results



Workflow Engine - Option A (Code Review)

openapi.json

default

GET /health Health Check

POST /graph/create Create Graph

POST /graph/run Run Graph

GET /graph/state/{run_id} Get Run State

Schemas

CreateGraphRequest > Expand all object

CreateGraphResponse > Expand all object

HTTPValidationError > Expand all object

NodeConfig > Expand all object

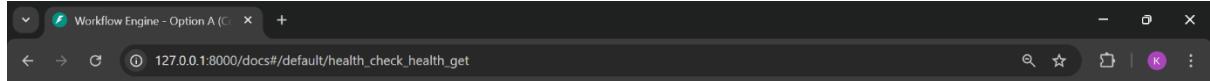
RunGraphRequest > Expand all object

RunGraphResponse > Expand all object

RunLogEntry > Expand all object

RunStateResponse > Expand all object

ValidationError > Expand all object



Workflow Engine - Option A (Code Review)

openapi.json

default

GET /health Health Check

Parameters

Cancel

No parameters

Execute

Clear

Responses

Curl

```
curl -X "GET" \
  "http://127.0.0.1:8000/health" \
  -H "accept: application/json"
```



Request URL

<http://127.0.0.1:8000/health>

Server response

Code Details

200

Response body

```
{  
  "status": "ok"  
}
```

Download

1) Dynamic Workflow Graph Creation using /graph/create Endpoint

The screenshot shows a browser window with the URL `127.0.0.1:8000/docs#/`. The main content area is a Swagger UI interface for the `/graph/create` endpoint.

POST /graph/create Create Graph

Create a new graph dynamically.

Parameters

No parameters

Request body required

application/json

```
Edit Value : Schema
{
    "www_key": {
        "next": {}
    },
    "analyze": {
        "name": "analyze",
        "tool": "check_complexity",
        "root_key": "",
        "next": {},
        "default_next": "end"
    },
    "end": {
        "name": "end",
        "tool": "finalize_review",
        "root_key": "",
        "next": {},
        "default_next": null
    }
}
```

Responses

Curl

```
curl -X "POST" \
  "http://127.0.0.1:8000/graph/create" \
  -H "Content-Type: application/json" \
  -d '{ "www_key": { "next": {} }, "analyze": { "name": "analyze", "tool": "check_complexity", "root_key": "", "next": {}, "default_next": "end" }, "end": { "name": "end", "tool": "finalize_review", "root_key": "", "next": {}, "default_next": null } }'
```

Responses

Curl

```
curl -X "POST" \
  "http://127.0.0.1:8000/graph/create" \
  -H "Content-Type: application/json" \
  -d '{ "www_key": { "next": {} }, "analyze": { "name": "analyze", "tool": "check_complexity", "root_key": "", "next": {}, "default_next": "end" }, "end": { "name": "end", "tool": "finalize_review", "root_key": "", "next": {}, "default_next": null } }'
```

Request URL

`http://127.0.0.1:8000/graph/create`

Server response

Code Details

200 Response body

```
{ "graph_id": "8dc24a32-3467-49e4-bfc7-3de93638c86d" }
```

Download

Response headers

```
content-length: 51
content-type: application/json
date: Wed, 06 Dec 2023 20:14:55 GMT
server: uvicorn
```

Responses

2) Workflow Execution and State Propagation using /graph/run Endpoint

The screenshot shows two consecutive views of a web application interface for executing a graph.

Top View (Request):

- URL:** 127.0.0.1:8000/docs#/
- Method:** POST /graph/run Run Graph
- Description:** Run an existing graph identified by graph_id with an initial state.
- Parameters:** No parameters
- Request body (required):** application/json
- Content:** A JSON object representing a graph configuration:

```
{
  "graph_id": "code_review_v1",
  "initial_state": {},
  "additionalProp1": {}
},
  "max_steps": 50
}
```
- Buttons:** Execute, Clear

Bottom View (Response):

- URL:** 127.0.0.1:8000/graph/run
- Responses:**
 - Curl:** curl -X POST -H "Accept: application/json" -H "Content-Type: application/json" -d '{"graph_id": "code_review_v1", "initial_state": {}, "additionalProp1": {}}, {"max_steps": 50}'
 - Request URL:** http://127.0.0.1:8000/graph/run
- Server response:**
 - Code:** 200
 - Details:** Response body
 - Content:** A large JSON object representing the execution results, including a graph ID, final state, function count, complexity score, and various logs and metrics. It includes sections for 'log', 'state', and 'function' counts.

3) Run Tracking and Execution State Monitoring via /graph/state/{run_id} Endpoint

The screenshot shows two consecutive views of a browser window displaying the API documentation for the `/graph/state/{run_id}` endpoint.

Top View:

- URL:** `127.0.0.1:8000/docs#/default/get_run_state_graph_state_run_id_get`
- Method:** `GET /graph/state/{run_id} Get Run State`
- Description:** Fetch the current/terminal state of a particular run.
- Parameters:**

Name	Description
<code>run_id</code> * required	string (path)
- Buttons:** `Execute` (highlighted in blue), `Cancel`, `Clear`
- Responses:**
 - Curl:**

```
curl -X 'GET' \
  'http://127.0.0.1:8000/graph/state/e38c6837-cddc-44f1-8b85-f1c8b46c9423' \
  -H 'accept: application/json'
```
 - Request URL:** `http://127.0.0.1:8000/graph/state/e38c6837-cddc-44f1-8b85-f1c8b46c9423`
 - Server response:**

Code	Details
200	Response body
- Response Body (Partial):**

```
{
  "id": "e38c6837-cddc-44f1-8b85-f1c8b46c9423",
  "graph_id": "code_review_v1",
  "current_node": null,
  "state": "initial",
  "additionalProp": {},
  "function_count": 0,
```

The screenshot shows the same browser view as the top one, but with the response body expanded to show more details.

Responses:

Response Body (Full):

```
{
  "id": "e38c6837-cddc-44f1-8b85-f1c8b46c9423",
  "graph_id": "code_review_v1",
  "current_node": null,
  "state": "initial",
  "additionalProp": {},
  "function_count": 0,
  "complexity_score": 0,
  "issues": 0,
  "snapshots": [],
  "quality_score": 100,
  "route": "finis",
  "running_status": "completed"
},
{
  "node": "true",
  "log": [
    {
      "mode": "extract",
      "state_snapshot": {
        "AdditionalProp": {},
        "FunctionCount": 0
      }
    },
    {
      "mode": "complexity",
      "state_snapshot": {
        "AdditionalProp": {},
        "FunctionCount": 0
      }
    }
  ]
}
```

Response Headers:

```
content-length: 1064
content-type: application/json
date: Wed, 16 Dec 2023 20:17:11 GMT
server: uvicorn
```