Failure Policy Executor Server

This server can be deployed to extend the functionality of policies by handling failure cases. The Failure Policy Server is also a policy executor, specifically designed to execute fallback or failure policies when the original policy fails. It also supports logging for these failure events. Developers can deploy the Failure Policy Server and link it to a specific policy rule using a REST API, so that it gets triggered when a failure is reported.

The failure policy can be used to handle policy execution failures. For example, it can trigger a custom webhook to notify users or developers via Slack, Microsoft Teams, or other channels when a policy fails.

Failure policy executor server REST API:

API: `POST /executeFailurePolicy`

Description:
Executes a failure/fallback policy by evaluating the specified `failure_policy_id` with provided `inputs` and `parameters`. This API is typically called when another policy fails, and the failure policy is linked to handle such scenarios (e.g., notifications, logging, retries, etc.).

Environment Variable Required:
- `POLICY_EXECUTOR_API_URL`: Must be set to ensure the policy executor can function.

Request Body (JSON):

```
{
  "failure_policy_id": "string",    // Required: ID of the failure policy to execute
  "inputs": { ... },                // Required: Input data for the policy
  "parameters": { ... }             // Required: Additional parameters for policy execution
}
```

Response: - `200 OK`: Execution successful, result returned in `data`. - `400 Bad Request`: Missing required parameters. - `500 Internal Server Error`: Server error or misconfiguration.

Sample `curl` Request:

```
curl -X POST http://<your-server-host>/executeFailurePolicy \
  -H "Content-Type: application/json" \
  -d '{
        "failure_policy_id": "notify-slack",
        "inputs": {
          "error_message": "Policy X failed due to timeout"
        },
        "parameters": {
          "webhook_url": "https://hooks.slack.com/services/XXX/YYY/ZZZ"
        }
      }'
```

Integrating with the policy module using python

```python
import requests
import logging

def call_failure_policy(api_url: str, failure_policy_id: str, inputs: dict, parameters: dict
    """
    Calls the /executeFailurePolicy endpoint to execute a failure policy.

    Args:
        api_url (str): Base URL of the failure policy server (e.g., http://localhost:8000).
        failure_policy_id (str): ID of the failure policy to execute.
        inputs (dict): Input payload for the policy.
        parameters (dict): Parameters for the policy.

    Returns:
        dict: Response from the server.
    """
    try:
        url = f"{api_url}/executeFailurePolicy"
        payload = {
            "failure_policy_id": failure_policy_id,
            "inputs": inputs,
            "parameters": parameters
        }
        headers = {"Content-Type": "application/json"}

        response = requests.post(url, json=payload, headers=headers)
        response.raise_for_status()
        return response.json()
    except requests.RequestException as e:
        logging.error(f"Request failed: {e}")
        return {"success": False, "message": str(e)}
    except Exception as ex:
        logging.error(f"Unexpected error: {ex}")
        return {"success": False, "message": str(ex)}
```