AP050R.MBR

Path: NXCLOUD/rpgsrc/AP050R.MBR **Generated**: 2026-01-08 12:04:36 **Processing Time**: 10827ms

Business Logic for Profile Manager API Interaction

This document outlines the business rules that govern the interaction with the Profile Manager API, based on an analysis of the RPG program AP050R. The primary focus is on how the program constructs requests to the API, handles responses, and manages file operations.

The core logic for API interaction is contained within the main program logic of AP050R. The program prepares a request to the Profile Manager API, processes the response, and cleans up temporary files on the Integrated File System (IFS).

API Request and Response Handling Rules

Profile Manager API Interaction: AP050R

1. Request Construction

- **Logic:** The program constructs a request to the Profile Manager API by creating a request file and a headers file containing the necessary authentication token.
- **File:** IFS_Output1 (Request file for API)
- **Field:** w_token
- **Condition:** The request file is created only if the directory change to the specified IFS path is successful.

2. Header File Creation

- **Logic:** A header file is created to store the API key required for authentication when making the API call.
- **File:** IFS_Output2 (Header file for API)
- **Field:** w_tokeut
- **Condition:** The header file is created only if the directory change to the specified IFS path is successful.

3. API Call Execution

- **Logic:** The program calls an external service program (AW702C) to execute the API request, passing in the necessary parameters such as URL, request type, and file paths.
- **File:** N/A (External service call)
- **Field:** N/A
- **Condition:** The API call is executed after constructing the request and header files.

4. Response File Handling

- **Logic:** After the API call, the program checks for the existence of the response file and processes it if it is successfully retrieved.
- **File:** IFS_Input (Response file from API)
- **Field:** API_status
- **Condition:** The response file must exist for further processing; if not, an error status is set.

Error Handling and Cleanup Rules

1. Error Handling

•**Logic:** If the API call fails, the program sets an error status and proceeds to cleanup.

•**File:** N/A (Error handling)

•**Field:** p_stat

•**Condition:** Triggered by a failure in the API call, leading to setting p_stat to '8'.

2. File Cleanup

•**Logic:** Temporary files created during the process are deleted to maintain a clean environment.

•**File:** IFS_Output1, IFS_Output2, IFS_Input (Temporary files)

•**Field:** N/A

•**Condition:** Cleanup is performed regardless of the success of the API call, although the specific cleanup code is currently commented out.

Initialization and Parameter Handling Rules

1. Parameter Initialization

•**Logic:** The program initializes parameters and retrieves necessary values from the database to prepare for the API interaction.

•**File:** afpspf, aposextnst (Database tables)

•**Fields:**

•w_url (API base URL)

•w_token (API authentication token)

•**Condition:** Parameters are initialized at the start of the program execution.

2. Session Management

•**Logic:** The program manages session identifiers and user credentials for the API request.

•**File:** N/A (Session management)

•**Field:** w_ident, w_usrn, w_pass

•**Condition:** These fields are populated before making the API call.

Subprogram Calls Affecting Logic

Beyond direct file checks, several external subprograms are called that play a significant role in the workflow.

1. AW702C (API Call Handler)

•**Trigger:** Called after constructing the request and header files.

•**Logic:** This program handles the actual HTTP request to the Profile Manager API.

•**Impact:** This call is essential for executing the API interaction and retrieving the response.

2. AP051R (Response Processing)

•**Trigger:** Called if the API response file is successfully retrieved.

•**Logic:** This program processes the data contained in the API response.

•**Impact:** This step is crucial for extracting and utilizing the data returned from the API.

3. AS100R (Parameter Initialization)

•**Trigger:** Called during the initialization phase to set up parameters.

- **Logic:** This program retrieves necessary parameters from the database.
- **Impact:** It ensures that the program has the required configuration to interact with the API correctly.