AP050R.MBR

Path: NXCLOUD/rpgsrc/AP050R.MBR **Generated**: 2026-01-08 13:32:00 **Processing Time**: 12446ms

Business Logic for Profile Manager API Integration

This document outlines the business rules that govern the integration 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 on the Integrated File System (IFS).

The core logic for the Profile Manager API integration is contained within the main program logic of AP050R. The program prepares a request to the Profile Manager API, processes the response, and performs cleanup operations on the IFS.

API Request and Response Handling Rules

Profile Manager API Integration: AP050R

1. Request Construction

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

2. Header File Creation

- **Logic:** A separate headers file is created to store the API key required for authentication.
- **File:** IFS_Output2 (Headers file for API)
- **Field:** w_tokeut
- **Condition:** The headers file is created if the directory change to the specified IFS path is successful.

3. API Call Execution

- **Logic:** The program calls an external web service to send the request and receive a response.
- **File:** IFS_Input (Response file from API)
- **Field:** w_url2
- **Condition:** The API call is made using the constructed URL and request parameters.

4. Response File Validation

- **Logic:** The program checks if the response file is successfully created after the API call.
- **File:** IFS_Input (Response file from API)
- **Condition:** If the response file cannot be opened, the API status is set to '9', indicating an error.

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
•**Condition:** Triggered when the API call does not return a valid response file.

2. File Cleanup

•**Logic:** The program attempts to delete temporary files created during the API request process.
•**File:** IFS_Output1, IFS_Output2, IFS_Input
•**Condition:** Cleanup is performed at the end of the program execution, regardless of success or failure.

Initialization and Parameter Handling Rules

1. Parameter Initialization

•**Logic:** The program initializes parameters required for the API call, including firm and member identifiers.
•**File:** N/A
•**Condition:** This occurs at the beginning of the program during the initialization subroutine.

2. Token Retrieval

•**Logic:** The program retrieves the API token from the aposextnst table based on the provided firm identifier.
•**File:** aposextnst (Database table for company settings)
•**Field:** w_token
•**Condition:** The token is fetched using an SQL query if the firm identifier is valid.

Subprogram Calls Affecting Logic

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

1. AW702C (Web Service Call)

•**Trigger:** Called to execute the API request after constructing the necessary parameters.
•**Logic:** This subprogram handles the actual HTTP request to the Profile Manager API.
•**Impact:** This call is critical as it represents the main interaction with the API, determining the success of the request.

2. AP051R (Response Processing)

•**Trigger:** Called to process the response received from the API.
•**Logic:** This subprogram extracts relevant data from the response file and updates the status.
•**Impact:** This call is essential for handling the data returned from the API, affecting further processing.

3. AS100R (Number Generation)

•**Trigger:** Called to generate a unique number for the API request.
•**Logic:** This subprogram initializes the request number and prepares it for use in the API call.
•**Impact:** This is important for ensuring that each API request is uniquely identified, which can be critical for tracking and logging purposes.