AP050R.MBR

Path: NXCLOUD/rpgsrc/AP050R.MBR **Generated**: 2026-01-08 12:34:26 **Processing Time**: 12201ms

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, handles responses, and manages file operations related to the API.

The core logic for the Profile Manager API integration is contained within the main program logic of AP050R. The program initiates a request to the Profile Manager API, processes the response, and performs necessary file operations on the Integrated File System (IFS).

Request and Response Handling Rules

Profile Manager API Integration: AP050R

1. Constructing the Request

- **Logic:** The program constructs a request to the Profile Manager API by creating a request file and a headers file. The request file is initially empty, while the headers file contains the API token.
- **File:** IFS_Output1 (Request file for API)
- **Field:** w_reqf
- **Condition:** The request file is created only if the directory change to the specified IFS path is successful.

2. Writing the API Key

- **Logic:** After creating the headers file, the program writes the API key into the headers file.
- **File:** IFS_Output2 (Headers file for API)
- **Field:** w_tokeut
- **Condition:** The file must be successfully opened for writing; otherwise, the program will terminate.

3. Making the API Call

- **Logic:** The program constructs the URL for the API call and invokes a subprogram to perform the HTTP GET request to the Profile Manager API.
- **File:** API_Input (Response file for API)
- **Field:** w_url2
- **Condition:** The API call is made only if the request and response file paths are correctly constructed.

4. Handling the API Response

- **Logic:** The program checks if the response file has been created successfully and processes the data if it exists.
- **File:** API_Input (Response file for API)
- **Field:** API_status
- **Condition:** The response file must be opened successfully; 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 the cleanup section.

•**File:** N/A

•**Field:** p_stat

•**Condition:** The error handling routine is triggered when the API call does not succeed.

2. File Cleanup

•**Logic:** The program attempts to clean up the files created during the request process.

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

•**Condition:** Cleanup is performed regardless of the success of the API call, ensuring no residual files remain.

Initialization and Parameter Handling Rules

1. Parameter Initialization

•**Logic:** The program initializes parameters and retrieves necessary values such as the API URL and token from the database.

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

•**Fields:** w_url, w_token

•**Condition:** Parameters are fetched from the database based on the firm identifier provided as input.

2. Setting Up the Environment

•**Logic:** The program sets up the environment by changing the directory to the specified IFS path before file operations.

•**File:** N/A

•**Field:** IFS_path

•**Condition:** The directory change must be successful to proceed with file operations.

Subprogram Calls Affecting Logic

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

1. AW702C (HTTP Request Handler)

•**Trigger:** Called to perform the HTTP GET request to the Profile Manager API.

•**Logic:** This subprogram handles the HTTP request and response processing.

•**Impact:** This call acts as a **major logical gateway** for interacting with the Profile Manager API.

2. AP051R (Response Processing)

•**Trigger:** Called to process the response received from the API.

•**Logic:** This subprogram extracts relevant data from the API response file.

•**Impact:** This is a **critical step** in ensuring that the data received from the API is handled correctly.

3. AS100R (Number Generator)

•**Trigger:** Called to retrieve a unique number for the request.

•**Logic:** This subprogram generates a unique identifier used in file naming.
•**Impact:** This ensures that each request is uniquely identifiable, preventing file overwrites.
This documentation provides a comprehensive overview of the business logic encapsulated in the AP050R program, detailing how it interacts with the Profile Manager API and manages associated file operations.