Step 1: Secure Staging Directory for Meta Ads ETL

Purpose:
To securely store temporary Meta Ads API extraction files before loading them into SQL, ensuring data security and access control.

Actions Taken:
• Created sub-folder: C:\Secure_ETL_Staging\meta_ads_data.
• Confirmed the folder inherits locked-down permissions and EFS encryption from the parent directory.
• Verified that only SYSTEM, Administrators, and your specific user account have access permissions.

Step 2: Environment Variables Setup
Purpose:
To securely store Meta Ads API credentials without hardcoding.
Actions Taken:

- Updated .env with:
    - META_ACCESS_TOKEN
    - META_APP_ID
    - META_APP_SECRET
    - META_AD_ACCOUNT_ID
- Verified all credentials are injected via environment and not stored in code.

Step 3: Logging Setup
Purpose:
To track ETL execution, API responses, and errors for audit readiness.
Actions Taken:

- Log file auto-creation planned: C:\Secure_ETL_Staging\logs\meta_ads_etl.log
- Logging format and retention will follow existing logging_config setup.

Step 4: Data Validation & Structure Handling
Purpose:
To ensure clean, schema-consistent data from Meta Ads API before SQL load.
Actions Taken:

• Inline JSON structure validation in-memory for each API response.
• Used fallback defaults for missing fields.
• Typecasting of spend, conversions, revenue to ensure consistency.
• Applied final revenue calculation logic using purchase_roas * spend if needed.
• Included campaign-level debug logging for action_values count and revenue validation.

Step 5: Table Creation in Secure Database

Purpose:

To store cleaned Meta Ads campaign metrics for reporting and BI dashboards.

Actions Taken:

• Created meta_ads_campaigns_fact table inside shopify_etl schema.
• Defined primary key as campaign_id + start_date for unique weekly aggregation.
• Set etl_user permissions with SELECT, INSERT, and UPDATE only (delete restricted for security).

Step 6: Secure ETL Script Development

Purpose:

To automate weekly extraction, validation, transformation, and loading of campaign metrics.

Actions Taken:

• Developed meta_ads_etl_pipeline.py mirroring Google Ads ETL best practices.
• Used secure environment variable loading for API keys.
• Added fallback for missing fields and inline type validations.
• Introduced revenue calculation safeguards.
• Structured, file-based logging enabled.

Step 7: Automated Scheduling

Purpose:

To ensure timely weekly data pulls without manual intervention.

Actions Taken:

• Registered Windows scheduled task using schtasks.
• Task set to run every Wednesday at 3:00 PM.
• Execution verified via log generation and successful data load checks.

Step 8: Automated Logs Cleanup

Purpose:

To avoid storage bloat from weekly logs.

Actions Taken:

- Created metaads_log_cleanup.py in utils.
- Configured to purge log files older than 60 days.
- Scheduled automated cleanup monthly via schtasks.