# Almina extract data

## Almina extract data

### Git repo: GitLab

### Overview

This module provides **backward compatibility** for legacy tests and systems, while internally leveraging a new modular architecture for data processing. It acts as a bridge, maintaining the old API while using improved, maintainable, and testable components.

**Key Features:**

- Processes Excel files from Google Cloud Storage (GCS)
- Validates and cleans tabular data
- Uploads processed data to Google BigQuery
- Structured logging and robust error handling
- Designed for both batch and event-driven (Cloud Function) execution

---

### Architecture

- **Wrappers:** Functions maintain the legacy API but delegate logic to new modular components.
- **Modular Imports:** Uses new architecture from `src.config`, `src.utils`, `src.models`, `src.services`, and `src.processors`.
- **Cloud Integration:** Handles authentication, storage, and BigQuery operations using Google Cloud libraries.
- **Logging:** Structured logging for both local and cloud environments.
- **Data Validation:** Ensures data quality before upload.

---

### Configuration

The module reads configuration from environment variables for backward compatibility:

| Variable Name | Description | Default Value |
|---|---|---|
| `GCP_PROJECT_ID` | Google Cloud Project ID | `'your-project-id'` |
| `GCS_BUCKET_NAME` | GCS bucket containing Excel files | `'your-bucket-name'` |
| `BQ_DATASET_ID` | BigQuery dataset ID | `'your-dataset-id'` |
| `BQ_TABLE_ID` | BigQuery table ID | `'almina_data'` |
| `CUSTOMER_ID` | Customer identifier | `'almina_1'` |
| `GOOGLE_APPLICATION_CREDENTIALS` | Path to GCP credentials file | `None` |

---

### Main Functions

#### 1. setup_logging()

Configures structured logging for both local and GCP environments.

#### 2. retry_with_backoff(max_retries=3, backoff_factor=2.0)

Decorator for retrying functions with exponential backoff on failure.

#### 3. sanitize_column_name(original_name)

Converts column headers to snake_case and ensures BigQuery compatibility.

#### 4. get_bq_schema()

Returns the explicit BigQuery schema as a list of `SchemaField` objects.

### 5. map_columns_to_output_schema(df)

Maps incoming DataFrame columns to canonical schema names, tolerating minor variations.

### 6. validate_data_quality(df, file_name)

Validates the DataFrame for structure, required columns, data types, and value ranges. Returns a validation report.

### 7. process_xlsx_file(file_content, file_name, upload_time_to_bucket=None)

Processes an Excel file:

- Reads and cleans data from the 'Resumo' sheet
- Maps and sanitizes columns
- Validates data quality
- Adds metadata columns
- Returns a cleaned DataFrame ready for upload

### 8. authenticate_gcp()

Authenticates with GCP using service account credentials and returns storage and BigQuery clients.

### 9. ensure_table_exists(bq_client)

Ensures the BigQuery table exists with the correct schema, partitioning, and clustering.

### 10. upload_to_bigquery(df_processed, bigquery_client)

Uploads the processed DataFrame to BigQuery, ensuring schema compliance.

### 11. process_bucket_files()

Processes all `.xlsx` files in the configured GCS bucket and uploads them to BigQuery.

### 12. process_xlsx_cloud_function(event, context)

Entry point for GCP Cloud Function, triggered by file uploads to the bucket.

### 13. main()

Main entry point for batch processing. Validates environment, processes all files, and handles errors.

---

## Usage

- **Batch Mode:** Run as a script to process all files in the bucket.
- **Cloud Function:** Deploy `process_xlsx_cloud_function` for real-time processing on file upload.

---

## Error Handling & Logging

- All critical operations are wrapped with structured logging.
- Errors are logged with context and re-raised for visibility.
- Data validation errors prevent upload to BigQuery.

---

## Security & Compliance

- Uses Google Cloud IAM and service accounts for authentication.
- Handles credentials securely via environment variables.
- **Note:** Always verify compliance with your organization's data handling and cloud security policies.

---

## Example: Cloud Function

```
Deploy almina-extract-data as a GCP Cloud Function with appropriate triggers and permissions.
```

---

## Encouragement for Critical Thinking

This documentation is AI-generated and should be reviewed for accuracy and completeness. Always validate the module's behavior in your environment and consult your team for best practices.