Canvas API Academic Data Collector

Project Overview

In this project, we will use Python to call Canvas API to get the student's academic performance data.

We will mainly focus on 2 datasets:

1. ALL student's currently enrolled courses' scores.
2. ALL student's currently enrolled courses' assignment status (submitted, graded, late, etc.).

Then we will use these data to update the database in CSV format for later use and visualization. The data is also prepared in a Notion-friendly format for integration with Notion databases.

Dataset 1: Scores

The dataset should include the following information:

- Student Name: Directly from Canvas API
- Student Chinese Name: Mapping from Student Name
- Student Preferred English Name: Mapping from Student Name
- Course Name: Directly from Canvas API
- Course Name in Chinese: Mapping from Course Name
- Score: Directly from Canvas API
- Grade: Simple mapping: 90-100 A, 80-89 B, 70-79 C, 60-69 D, 0-59 F
- Fetch Time: When the data is fetched. YYYY-MM-DD HH:MM:SS

We ALWAYS append the new data to the existing dataset.

Dataset 2: Assignment Status

The dataset should include the following information:

- Student Name: Directly from Canvas API
- Student Chinese Name: Mapping from Student Name
- Student Preferred English Name: Mapping from Student Name
- Course Name: Directly from Canvas API

- Course Name in Chinese: Mapping from Course Name
- Course Academic Support: Mapping from Academic Support Mapping
- Assignment Name: Directly from Canvas API
- Assignment ID: Directly from Canvas API
- Assignment Due Date: Directly from Canvas API
- Assignment Submission Time: Directly from Canvas API, May be date data or Not submitted
- Assignment Full Mark: Directly from Canvas API
- Assignment Student Score: Directly from Canvas API
- Assignment Status: Directly from Canvas API
- Fetch Time: When the data is fetched. YYYY-MM-DD HH:MM:SS

We DO NOT ALWAYS append the new data to the existing dataset.
We apply a specific rule for data update:

1. If the assignment is new, we append the data to the existing dataset.
2. If the assignment is not new, we only update the assignment data as in: fetch time, assignment status, assignment student score.
   This makes sure there is only one record for each assignment and avoids duplicate data.

Installation & Setup

1. Ensure Python 3.8+ is installed on your system
2. Clone this repository
3. Install required dependencies:

   ```
   pip install -r requirements.txt
   ```

4. Configure your Canvas API credentials in `credentials.json` (see Security Considerations section)

Project Structure

```
canvas_api/
├── main.py                    # Main entry point
├── config.py                  # Configuration settings and mappings
├── update_user_ids.py         # Utility for updating user IDs in credentials
├── data_collectors/
│   ├── __init__.py
│   ├── base_collector.py      # Base class for collectors
│   └── grades.py              # Collects course grades
├── notion_processor/
│   ├── __init__.py
│   ├── notion_main.py         # Main entry point for Notion processing
│   ├── data/                  # Notion-specific data files
│   │   └── notion_grades.csv # Notion-friendly formatted grades
│   └── utils/
│       ├── __init__.py
│       └── notion_formatter.py # Formats data for Notion compatibility
├── utils/
│   ├── __init__.py
│   ├── credential_manager.py # Manages student credentials
│   ├── csv_handler.py         # CSV file operations
│   └── error_handler.py       # Error handling utilities
└── data/                      # Directory for storing data files
    ├── grades.csv             # Raw grades data
    └── assignments.csv        # Raw assignments data
```

Configuration & Mappings

All configurations and mappings are stored in `config.py` :

1. File paths:
   - GRADES_CSV_PATH
   - ASSIGNMENTS_CSV_PATH
   - NOTION_GRADES_CSV_PATH
2. Data Mappings:
   - Student Name to Chinese Name
   - Student Name to Preferred English Name
   - Course Name to Chinese Name
   - Course Name to Academic Support

- Score to Grade mapping

Data Storage

Data is stored in CSV format for easy access and development:

1. `data/grades.csv` : Contains all collected course grades
   - Format: Student Name, Chinese Name, English Name, Course Name, Chinese Course Name, Score, Grade, Fetch Time
2. `data/assignments.csv` : Contains all collected assignment data
   - Format: Student Name, Chinese Name, English Name, Course Name, Chinese Course Name, Academic Support, Assignment Name, Assignment ID, Due Date, Submission Time, Full Mark, Student Score, Status, Fetch Time
3. `notion_processor/data/notion_grades.csv` : Contains grades data formatted for Notion
   - Format: Student Name, Chinese Name, English Name, [All Course Columns], Updated Time

Notion Integration

The project includes a dedicated Notion processor module that:

1. Reads data from the standard grades CSV
2. Transforms it into a Notion-friendly format:
   - Student information as rows
   - Course names as columns
   - Latest grades as values
   - All possible courses included, even if a student isn't enrolled
3. Automatically updates the Notion CSV each time the main script runs

This formatted data can be easily imported into a Notion database or used with the Notion API.

Error Handling

The application implements comprehensive error handling:

1. API Connection Errors: Display connection failure messages with detailed error information
2. Authentication Errors: Provide clear guidance for API token issues
3. Data Processing Errors: Log specific data processing failures
4. File I/O Errors: Handle CSV file read/write exceptions

All errors are raised with descriptive messages to help troubleshoot issues and logged to `canvas_api.log`.

Security Considerations

For security in this project:

1. Student API credentials are stored in a separate `credentials.json` file outside of version control
2. The credentials file is excluded from version control via .gitignore
3. An example `credentials.json.example` file is provided as a template
4. Environment variables can still be used for global configuration
5. Ensure API tokens have minimal required permissions

To set up your credentials:

1. Copy credentials.json.example to credentials.json:
   `cp credentials.json.example credentials.json`
2. Edit credentials.json with each student's actual Canvas API values
3. Never commit the credentials.json file to version control

The credentials.json file uses a format where each student has their own section:

```
{
  "student1": {
    "api_key": "student1_canvas_api_key",
    "domain": "canvas_domain",
    "user_id": 123456,
    "student_name": "Canvas Display Name",
    "student_chinese_name": "中文名",
    "student_english_name": "Preferred English Name"
  },
  "student2": {
    "api_key": "student2_canvas_api_key",
    "domain": "canvas_domain",
    "user_id": 789012,
    "student_name": "Canvas Display Name",
    "student_chinese_name": "中文名",
    "student_english_name": "Preferred English Name"
  }
}
```

Scheduling & Automation

Data collection can be automated using:

1. Manual execution via `main.py`
2. Scheduled execution using system schedulers:
    - cron (Linux/Mac): `0 */6 * * * cd /path/to/project && python main.py`
    - Task Scheduler (Windows)
3. Future improvement: Implement automatic scheduling within the application

Dependencies

Required Python packages:

- requests==2.28.1
- python-dateutil==2.8.2
- pandas==1.5.0

Future Enhancements

- Direct integration with Notion API to update Notion databases
- Assignment data collection and processing
- Integration with Lark workspace via API
- Email notification system for grade updates