Analysis Data Cleaning Tool

Blue Group

Panchali Debnath
Cinthia Kone
Meltem Caliskan
Cel Du Bois
Aboukar Assidick Abdoulaye

1. Textual description

Data Cleaning Tool: Workflow and Description

The **Data Cleaning Tool** is a Python-based solution created to process and clean student datasets at the start of every academic year. Educational institutions often face the challenge of managing large datasets that include personal information like **name**, **date of birth**, **course**, and **nationality**. These datasets can be riddled with errors, duplicates, or inconsistencies, affecting the overall quality and reliability of the data.

This tool simplifies the process of cleaning the data by automating key tasks:

- **Removing duplicate entries** to ensure there is only one record per student.
- **Handling missing values** by identifying and filling or flagging incomplete data.
- **Correcting formatting issues**, such as inconsistent date formats or incorrect capitalization.
- **Validating and standardizing entries**, ensuring consistency across data points (e.g., ensuring consistent spelling of course names and nationalities).

The workflow follows these steps:

a. **Form Submission by Students**: At the start of the academic year, students receive a **data entry form** to complete. The form gathers essential information such as name, date of birth, course, nationality, and other relevant details.
b. **Form Submission to Administration :** Once completed, students submit their forms to the **administration** of the institution.
c. **Data Cleaning by Administration**: The administration then uploads the received data into the **Data Cleaning Tool**. The tool automatically processes the dataset by removing duplicates, filling in missing values, correcting any formatting errors, and ensuring that all entries are accurate and standardized.
d. **High-Quality Data for Analysis**: After the tool has processed the data, the institution has access to a clean and reliable dataset. This high-quality data can then be used for **administrative tasks** and **analytical purposes**, such as generating reports, assessing student demographics, and improving decision-making based on accurate information.

By automating the cleaning process, the **Data Cleaning Tool** helps institutions save time, reduce manual errors, and maintain **high-quality, consistent data** for their records.

2. Use cases/ user stories

a. Student user stories

As a student I want to easily access the data cleaning tool through a secure portal so I can review the information the university has on file for me.
As a student, I want to see a clear and concise summary of my personal information, including my name, contact details, student ID, and program of study, so I can quickly verify its accuracy.
As a student, I want to be able to correct any errors in my personal information, such as typos in my name or an incorrect phone number, directly through the tool, so I don't have to contact the university.

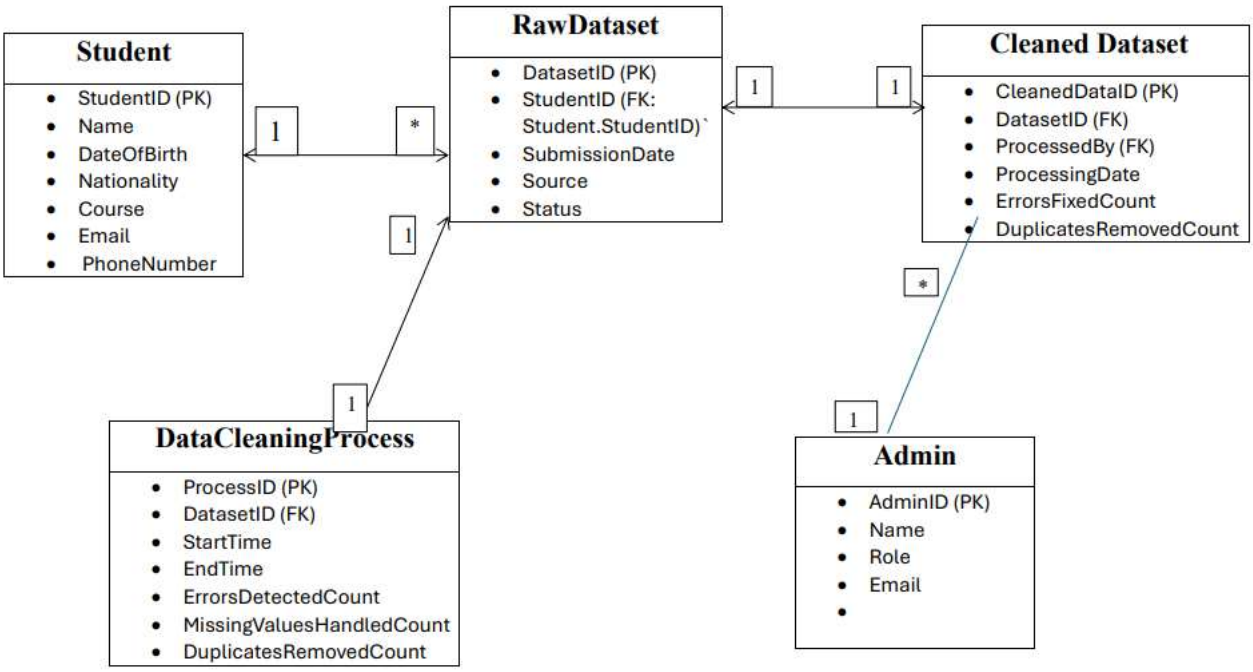
As a student, I want to receive confirmation that my corrections have been submitted and are under review by the university, so I know my changes are being processed.
As a student, I want to be notified when my corrections have been approved or if further information is needed, so I can stay informed about the status of my data.

As a student, I want to be able to securely upload supporting documents, like a copy of my passport or official transcript, if needed to verify my information, so I can provide necessary proof.
As a student, I want the tool to be accessible on different devices (desktop, mobile) so I can review and update my information conveniently.
As a student, I want the tool to be available in multiple languages so I can use it in my preferred language.

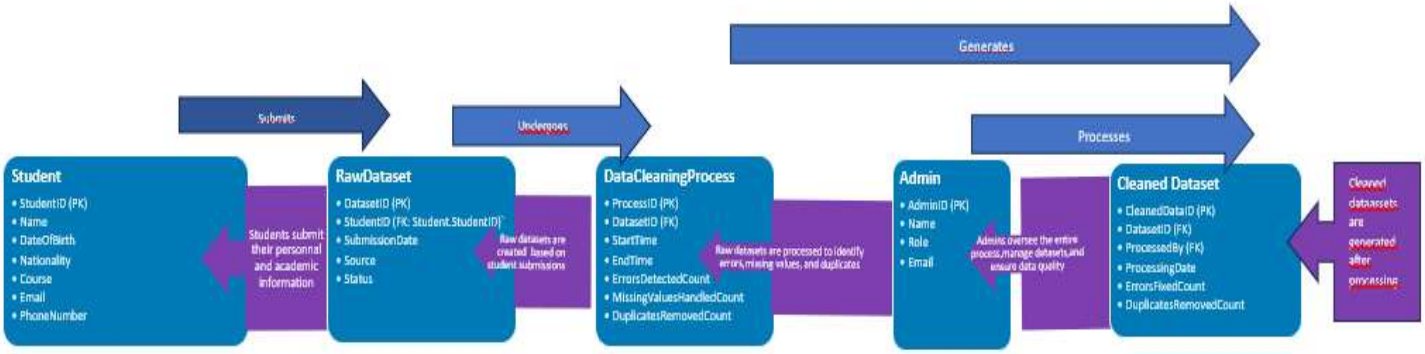
b. Data Entry Operator (administration) User Stories

As a data entry operator, I want to efficiently input student data from various sources (application forms, transcripts, etc.) into the data cleaning tool.
As a data entry operator, I want the tool to provide real-time validation checks as I enter data, so I can immediately correct any errors and ensure data accuracy.
As a data entry operator, I want the tool to flag potential duplicate student records based on name, ID, or other criteria, so I can investigate and merge them appropriately.
As a data entry operator, I want the tool to standardize student data (e.g., name formatting, address formats) automatically, so I don't have to manually correct inconsistencies.
As a data entry operator, I want to be able to search for student records quickly and easily using various criteria (name, ID, etc.), so I can efficiently access and update student information.
As a data entry operator, I want the tool to provide clear instructions and help documentation, so I can easily understand how to use all its features.
As a data entry operator, I want the tool to track all data entry and cleaning activities, so there's an audit trail of changes made to student records.
As a data entry operator, I want to be able to generate reports on data quality metrics (e.g., number of errors, duplicates), so I can monitor and improve data accuracy.
As a data entry operator, I want the tool to be secure and protect student data from unauthorized access or modification, so I can maintain data privacy and integrity.
As a data entry operator, I want the tool to integrate with other university systems (e.g., student information system), so I can seamlessly transfer data between systems.
As a data entry operator, I want to be able to manage student requests for data correction, reviewing their submissions and approving or requesting further information as needed.

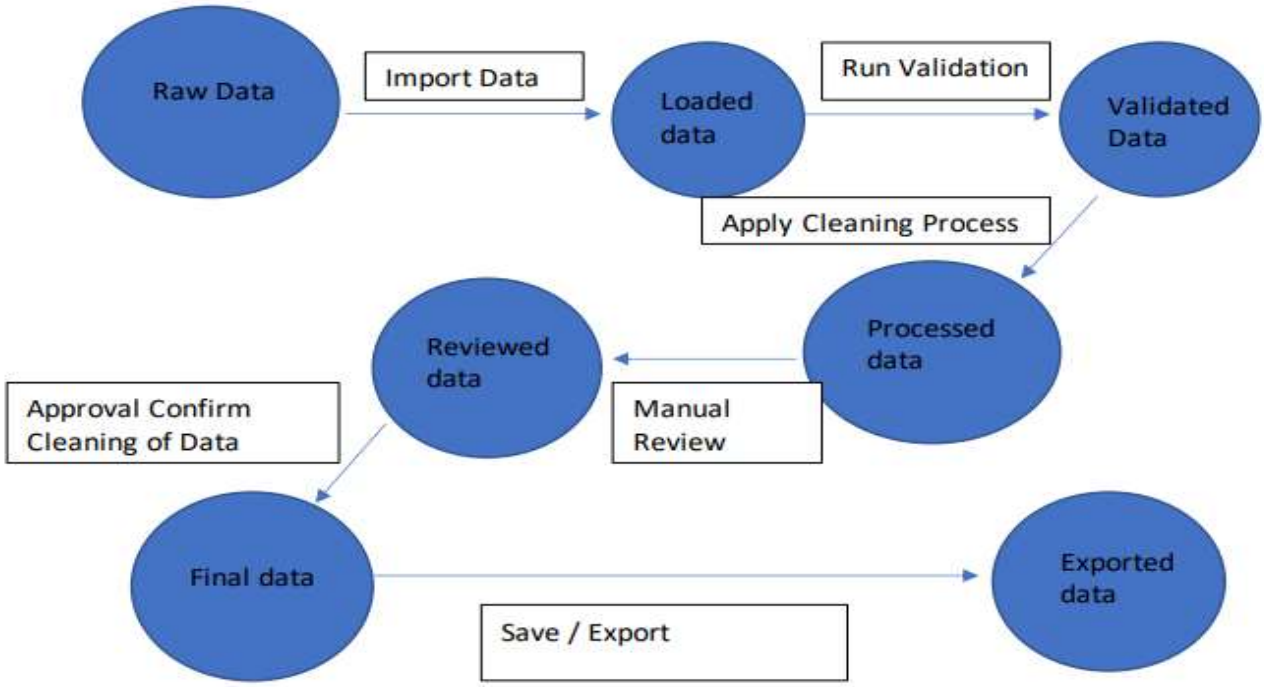
3. Data Model



4. Process Model



5. State Transition Model



a. States:

Raw Data → Data in its original, uncleaned form.
Loaded Data → Data has been imported into the system.
Validated Data → Basic checks (e.g., format, missing values) are performed.
Processed Data → Cleaning steps like deduplication, normalization, and type conversion are applied.
Reviewed Data → The cleaned data is checked for errors or inconsistencies.
Final Data → Data is fully cleaned and ready for export/use.
Exported Data → Cleaned data is exported or saved.

b. Transitions & Events:

Raw Data → Loaded Data (Import Data)
Loaded Data → Validated Data (Run Validation Checks)
Validated Data → Processed Data (Apply Cleaning Rules)
Processed Data → Reviewed Data (Manual Review / QA)
Reviewed Data → Final Data (Approval / Confirm Changes)
Final Data → Exported Data (Save / Export)

6. Examples

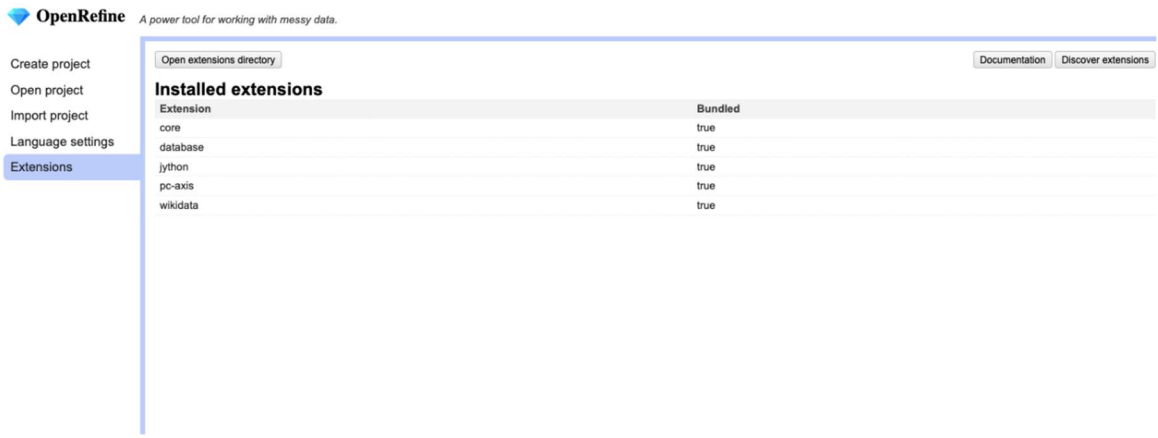
Codes using Python

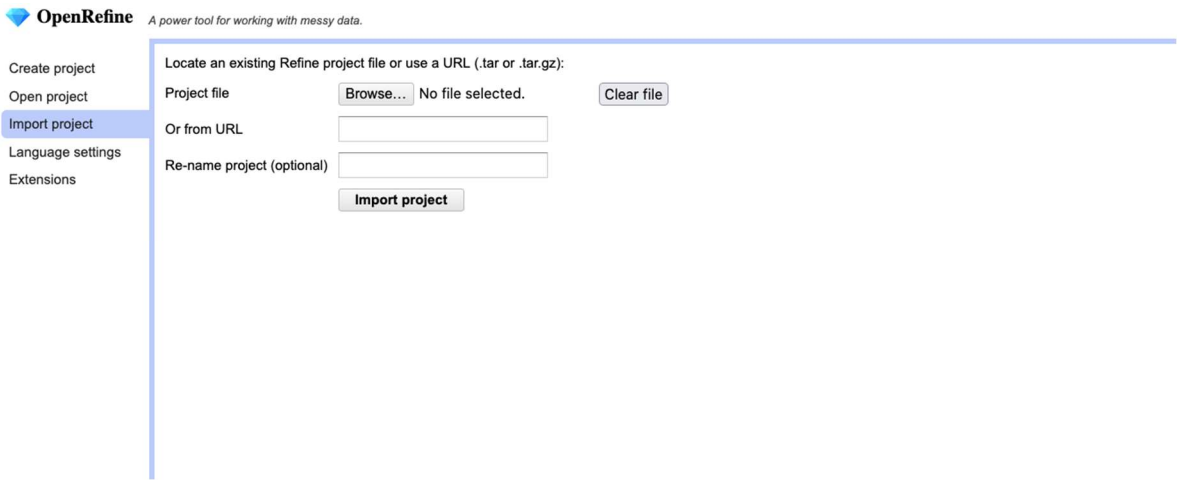
```
# Sample DataFrame
data = {'Date': ['2023-01-01', '2023-02-01', '2023-03-01'],
        'Value': ['100', '200', '300']}
df = pd.DataFrame(data)

# Convert 'Date' column to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Convert 'Value' column to numeric
df['Value'] = pd.to_numeric(df['Value'])

print("Data with correct data types:")
print(df)
```

```
# Sample DataFrame
data = {'Value': [10, 12, 14, 1000, 16, 18, 19, 20, 21, 22]}
df = pd.DataFrame(data)

# Calculating IQR (Interquartile Range)
Q1 = df['Value'].quantile(0.25)
Q3 = df['Value'].quantile(0.75)
IQR = Q3 - Q1

# Filtering out outliers (values outside 1.5 * IQR)
df_no_outliers = df[(df['Value'] >= (Q1 - 1.5 * IQR)) & (df['Value'] <= (Q3 + 1.5 * I

print("Data after removing outliers:")
print(df_no_outliers)
```

```
# Sample DataFrame with duplicates
data = {'Name': ['Alice', 'Bob', 'Alice', 'Charlie'],
        'Age': [25, 30, 25, 30],
        'City': ['New York', 'Los Angeles', 'New York', 'Chicago']}

df = pd.DataFrame(data)

# Removing duplicate rows
df_no_duplicates = df.drop_duplicates()

print("Data after removing duplicates:")
print(df_no_duplicates)
```

Data Cleaning Tool Interface


OpenRefine
A power tool for working with messy data.
Create project
Open project
Import project
Language settings
Extensions
Open extensions directory
Documentation
Discover extensions
Installed extensions
Extension
Bundled
core
true
database
true
jython
true
pc-axis
true
wikidata
true


OpenRefine
A power tool for working with messy data.
Create project
Open project
Import project
Language settings
Extensions
Locate an existing Refine project file or use a URL (.tar or .tar.gz):
Project file
Browse...
No file selected.
Clear file
Or from URL
Re-name project (optional)
Import project