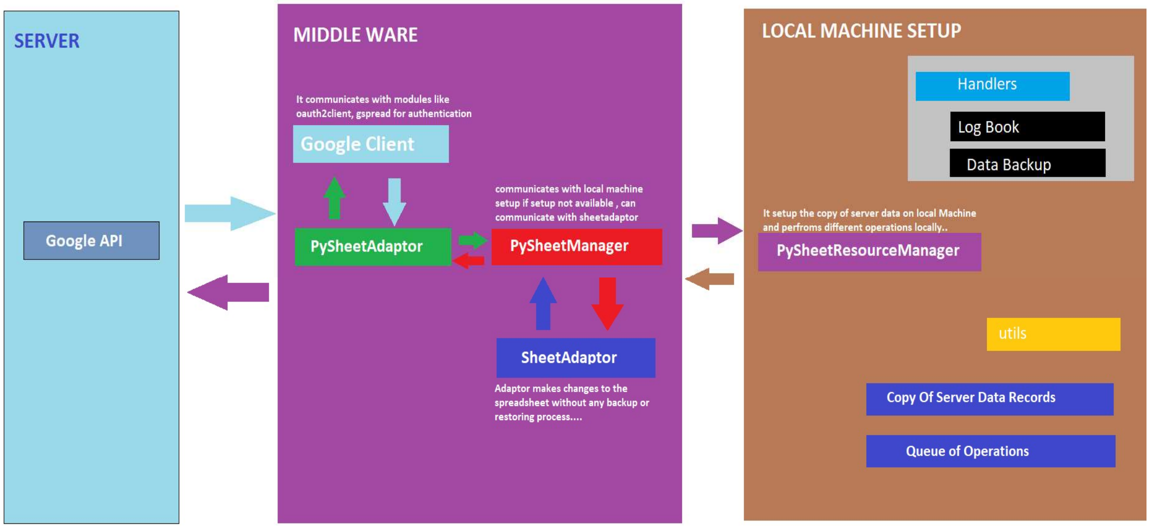
GOOGLE SPREADSHEET INTERFACE WITH LOGGING SYSTEM

This package is developed to interact and work with google spreadsheets, csv files effienctly. You can work with spread sheets from your local machine without any interface. It logs the output of every transaction on your local machine which helps in tracking the different operations performed on that particular spreadsheet or worksheet. You can back up your data into sheets until you have log file of that particular sheet at that particular state.

Here is the package architecture,



It communicates with modules like oauth2client, gspread for authentication

communicates with local machine setup if setup not available , can communicate with sheetadaptor

Adaptor makes changes to the spreadsheet without any backup or restoring process....

It setup the copy of server data on local Machine and perfroms different operations locally..

Figure () : Architecture of Package

Package Structure

- PySheetAdaptor
  - ➢ __init__.py
  - ➢ GoogleClient.py
  - ➢ Handlers.py
  - ➢ PySheetAdaptor.py
  - ➢ PySheetManager.py
  - ➢ PySheetResourceManager.py
  - ➢ SheetAdaptor.py
  - ➢ Utils.py

Now Let us discuss all the modules and their functionalities briefly

PySheetAdaptor()

it intilizes the adaptor and set up the vir env

Adaptor.connect (credentials_path)

***It communicates with the google api by authorizing with the following credentials.
It also continues the local setup with the given extra credentials***

Adaptor.create_spreadsheet (title)

It creates the spreadsheet by using the above parameters.
Return type :: Boolean value
True : if successfully created
False : if error in creating.

Adaptor.open_spreadsheet (title=*None*, url=*None*, key=*None*)

It opens the spreadsheet by using the above parameters.
Return type :: Boolean value
True : if successfully opened
False : if error in opening.

Adaptor.delete_spreadsheet (title=*None*, url=*None*, key=*None*)

It deletes the spreadsheet if exists
Return Type: True if deleted else False

Adaptor.create_worksheet(title)

It creates the work sheet with the given title.
Return Type: True if deleted else False…………

Adaptor.open_worksheet(title=*None*, url=*None*, key=*None*)

It creates the work sheet with the given title.
Return Type: True if deleted else False…………
Now it completes the local setup………

Adaptor.load_from_server()

loads the records from server and saves a copy of records in main memory for operations…………
To verify if data is loaded from the server you can check by using the Adaptor.__print__() method
Prints the data loaded and meta data created for the loaded data….

What is actually dos:
Loads the data from the server..
Saves the Record as the list of list..(Adaptor.records)
For efficient operations also loads into pandas dataframe(Adaptor.dataframe)
Sets all the meta data try(Adaptor.__meta__)

Adaptor.upload_to_server()

It uploads the processed data to the server and reloads the entire local machine setup.

Adaptor.close_connection()

it closes the connection and the adaptor is assigned as None ….

Adaptor.clear_sheet()

It clears the entire sheet…….

Adaptor.set_columns(columns)

It sets the header row for the worksheet…
It is also used the columns for dataframe to manipulate data by using pandas dataframe
Try: adaptor.get_columns
Adaptor.dataframe

Adaptor.insert_record(record, position=*None*)

Places the request to Inserts the given record with the given position
If position is none than append the record at last…

Adaptor.delete_record(keys, values)

Places the request delete the record with given keys from columns and having values..

Adaptor.update_record(keys, values, updated_columns, updated_values)

Places the request update the record with given keys from columns and having values..

Adaptor.search_record(keys, values)

searches the record with given values and returns the record……….

Adaptor.add_columns(column_name, column_values)

Places the request to add column with given column name and column values..

Adaptor.delete_colums(column_name)

Places the request to delete column with given column name..

Adaptor.commit()

It executes all the requested placed and make changes according to our request on the local copy of data…
It can execute batch of requests at a time…

Adaptor.rollback()

It track back to the previous state of the setup……..

Adaptor.back_up()

It backups the data from db log file
If you want to backup data you need to have that particular state of log db file..

Adaptor.to_html(file_path)

it creates a html file with given file path and loads the data in html format.

Adaptor.to_sql(table_name)

it loads the records data as form of table with given table name
for more info please try:
reference: Adaptor.sql_engine

Adaptor.to_csv(file_path)

it creates a csv file with given file path and loads the data in csv format.

Adaptor.to_excel(file_path)

it creates a excel file with given file path and loads the data in csv format.

If you want to directly with spreadsheet without any local setup below are the methods with minor changes…

Adaptor.disable_local_machine(flag)

Disables the local environment and directly interacts with the spreadsheet.

Adaptor.insert(values_list, position=*None*)

Inserts the given record with the given position
If position is none than append the record at last to the sheet without any request and backup

Adaptor.delete(keys, values)

delete the record with the given keys and values to the sheet without any request and backup

Adaptor.update(keys, values, updated_columns, updated_values)

updates the record with the given keys,values to the sheet without any request and backup

Adaptor.search(keys, values)

searches the record with given values and returns the record to the sheet without any request and backup