Yellow Pges

1. Import Statements:
   - The code begins by importing various Python libraries and modules. These include `logging`, `os`, `time`, `numpy`, `pandas`, `ApifyClient` from `apify_client`, `is_same_business` from a custom `lib.data_processing` module, and `pandarallel` for parallel processing.
2. Pandarallel Initialization:
   - `pandarallel.initialize()`: Initializes parallel processing using the `pandarallel` library. This allows for more efficient processing of DataFrames.
3. DataFrame Initialization:
   - `scraped_yellow_pages_data`: A pandas DataFrame is initialized with specific column names, such as "Business Name," "City," "BusinessNameYP," "BusinessAddressYP," "BusinessPhoneYP," and "BusinessWebsiteYP." This DataFrame is used to store Yellow Pages data.
4. Custom Exception Class:
   - `AuthenticationError`: This is a custom exception class defined to handle authentication errors. It inherits from the base `Exception` class and is used to raise exceptions related to authentication failures.
5. `add_yp_columns(data)` Function:
   - This function takes a DataFrame `data` as input and adds columns related to Yellow Pages (YP) data, initializing them with NaN values. The added columns include "BusinessNameYP," "AddressYP," "PhoneYP," and "WebsiteYP."
6. `update_dataframe_with_yellow_pages_data(data)` Function:
   - This function updates the given DataFrame `data` with information from Yellow Pages (YP) based on matching business names.
   - It first calls `add_yp_columns` to add YP-related columns to the DataFrame.
   - It then iterates through each row of the DataFrame, calling the `call_scrape_yellow_page_data` function for parallel processing.
   - After parallel processing, it iterates through the rows again and checks various conditions to update missing information based on YP data.
   - It updates columns like "BusinessNameUpdate," "BusinessNameFound," "PhoneUpdate," "PhoneFound," "WebsiteUpdate," "WebsiteFound," "AddressUpdate," and "AddressFound" based on YP data.
   - Finally, it returns the updated DataFrame.
7. `call_scrape_yellow_page_data(row: pd.Series)` Function:
   - This function is called for each row in the DataFrame.
   - It constructs a search term and location based on the data in the row.
   - It invokes the `scrape_yellow_page_data` function to scrape YP data for the given search term and location.
   - The scraped data is stored in a global DataFrame `scraped_yellow_pages_data`.
8. `scrape_yellow_page_data(searchTerm: str, location: str, maxItems: int, extendedOutputFunction="""...""")` Function:
   - This function uses the ApifyClient to scrape Yellow Pages data for a given search term, location, and other parameters.
   - It initializes the Apify client using a hardcoded API token.

   - It prepares input parameters for the Apify actor that performs the scrape.
   - It executes the actor and logs appropriate messages based on success or failure.
   - It iterates through the actor's results and returns relevant data based on the search term.
9. `login_yellow_pages()` Function:
   - This function attempts to authenticate and initialize the ApifyClient using a hardcoded API token.
   - If successful, it returns the ApifyClient; otherwise, it logs an error and returns False.

Overall, this code is designed to scrape Yellow Pages data, update a DataFrame with the scraped information, and handle authentication errors using custom exceptions. It also leverages parallel processing for efficient data processing.