README

for Code to Extract Data from PDFs, Scrape Web Data, and Store in Database

Overview

The provided code performs the following tasks:

1. Extracts well data and stimulation data from a list of PDF files.
2. Uses web scraping to gather additional information about each well based on its well name.
3. Preprocesses the extracted and scraped data.
4. Stores the final data in a MySQL database.

Prerequisites

Before you can run the code, you need to have the following software and Python libraries installed:

- Python (3.6 or later)
- ChromeDriver (compatible with your Chrome version)
- PyPDF2
- pdf2image
- pytesseract
- selenium
- webdriver_manager
- pandas
- numpy
- BeautifulSoup
- glob, os, re, time
- mysql.connector

You can install the necessary libraries using pip:

```
pip install PyPDF2 pdf2image pytesseract selenium webdriver_manager pandas numpy beautifulsoup4 mysql-connector-python
```

Also, you need to have MySQL set up and running on your system.

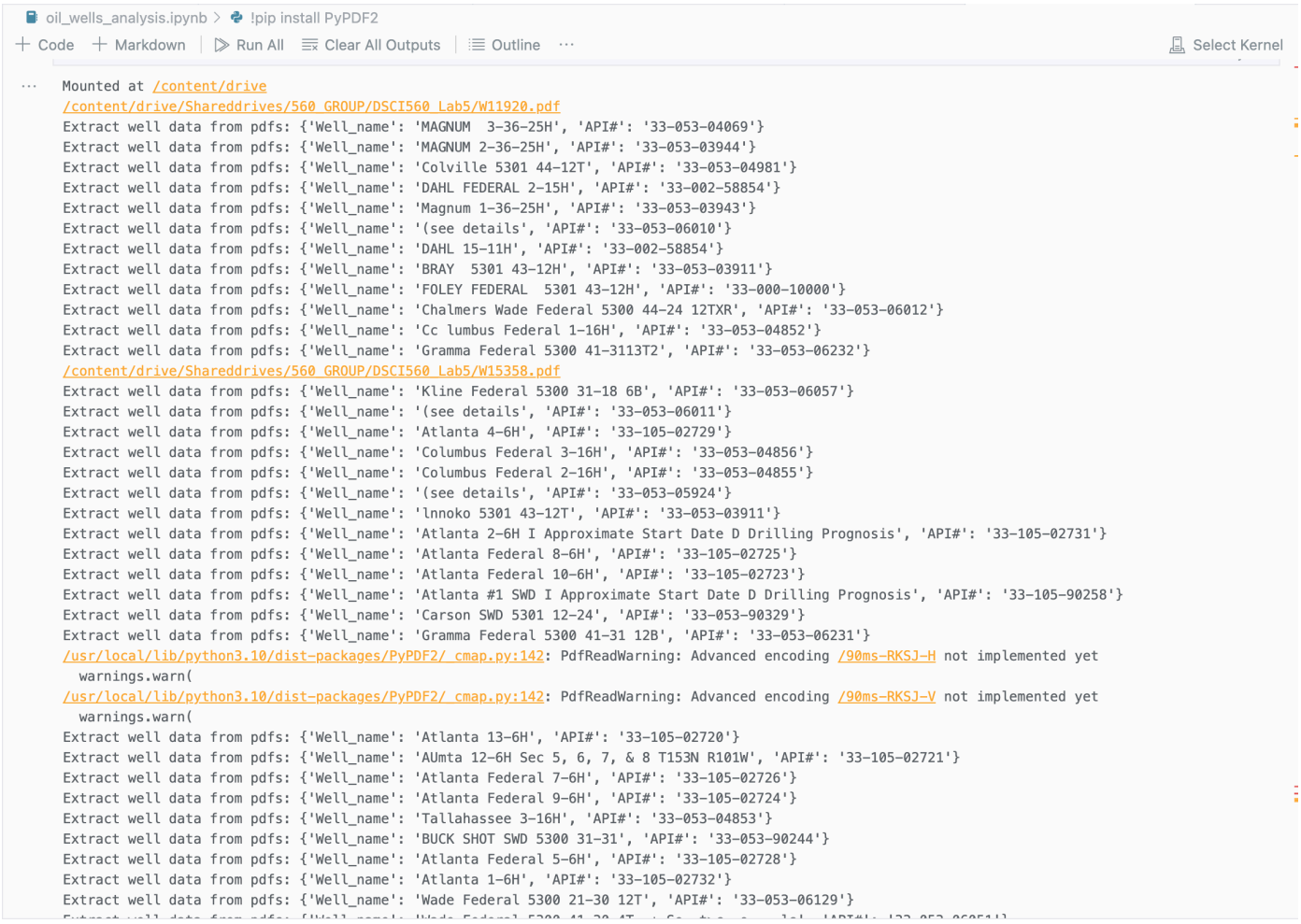
Running the Code

1. Place all the PDF files from which you want to extract data in a folder named `DSCI560_Lab5`.
2. Open a terminal or command prompt.
3. Navigate to the directory containing `Lab5_Part1.py`.
4. Run the script:

```
python Lab5_Part1.py
```

Outputs

1. `Task_PDF_original.csv`: Contains the data extracted from the PDF files.
2. `Task1_PDF_preprocessed.csv`: Contains the preprocessed version of the extracted data.
3. Data stored in the MySQL database in the table named `wells_data`.



Description of Functions

1. `well_name_func(pdf)`: Determines the pattern to extract the well name based on the PDF filename.
2. `api_func(pdf)`: Determines the pattern to extract the API number based on the PDF filename.
3. `api_fix_func(pdf, API_pattern_match)`: Fixes the API number format based on the matched API pattern.
4. `extract_well_data_from_pdfs(folder_path)`: Extracts well-related data from the PDF files.
5. `extract_text_from_pdf(pdf_path, page_number)`: Extracts text from a specific page of a PDF using OCR.
6. `extract_row1_data(row1_string)`, `extract_row2_data(row2_string)`: Extract data from specific strings.
7. `extract_stimulation_data_from_pdfs(pdf_files)`: Extracts stimulation data from the list of PDF files.
8. `scrape_well_data(df)`: Gathers well-related information from a website using web scraping.
9. `merge_and_save_dataframes(files_dict, stimulation_dict, output_folder)`: Merges extracted data and saves it as a CSV.
10. `preprocess_and_save_dataframes(df, output_folder, task)`: Preprocesses the data and saves it as a CSV.
11. `store_db(df)`: Stores the final data in a MySQL database.

Notes

- The PDF extraction process uses patterns matched with regular expressions. If the structure of the PDFs changes, the patterns might need adjustments.
- Web scraping is done using the Selenium library and the Chrome browser in headless mode. If the structure of the target website changes, the scraping logic might need adjustments.
- Before running the code, ensure that the MySQL server is up and running and that you have provided the correct database credentials.

Always make backups of your data and run the code on a small set of files first to ensure it works as expected.