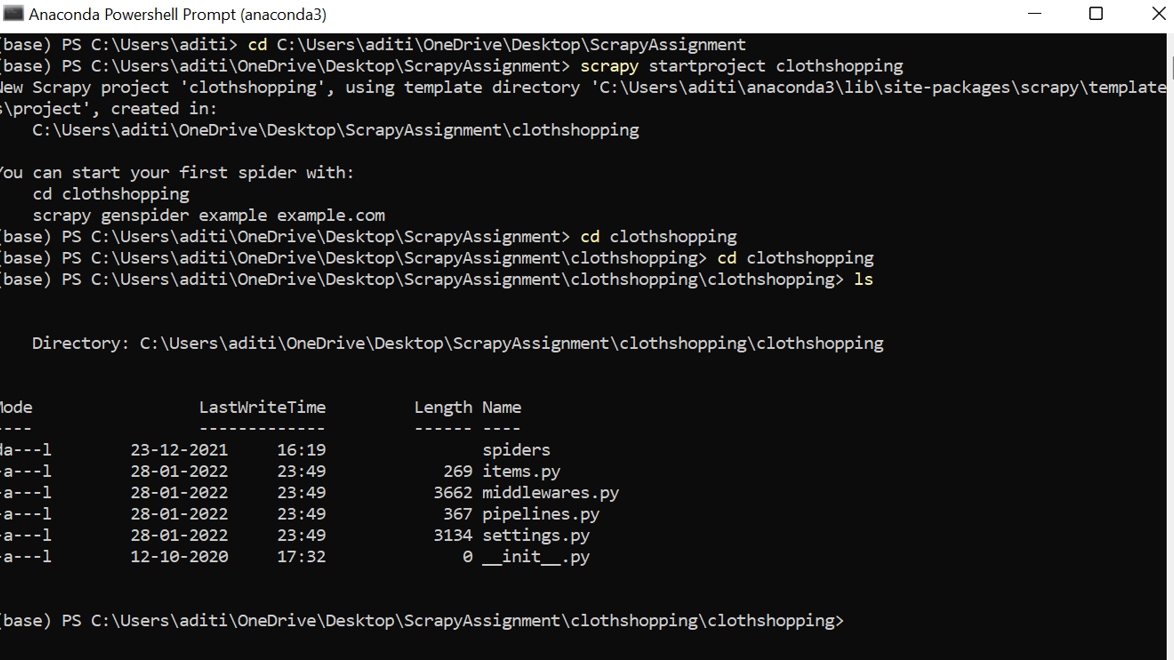
**Lab Assignment- PDV**

**Scraped the website -** [**https://shekouwoman.com/collections/home**](https://shekouwoman.com/collections/home)

To run Scrapy, install the libraries and create a folder to store the files needed for the scrapy project. The command ‘scrapy startproject’ creates the required .py files and spider to scrape the website.



**Scraping the website directly in the shell:**

The ‘scrapy shell’ command is used to scrape the website on the shell.

Text

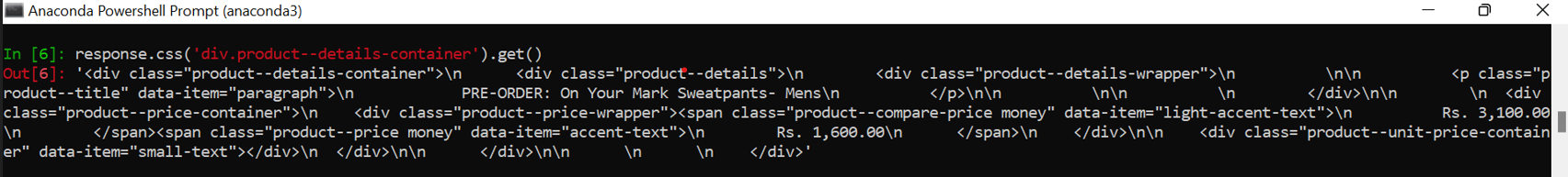
Description automatically generated

The ‘response.css’ command is used to along with the extract() to extract the tag we want and the relevant information.

Shape, rectangle

Description automatically generated

When there are multiple similar tags can use the dot operator along with the class name of tag to refer a particular tag for example response.css(‘div.product-details-container’).get().



All the product details that is the name actual price and the price after the discount is stored in a variable called product for easy access.

Text

Description automatically generated

We use the variable for further accessing the details stored in it.

Text

Description automatically generated

The get () method is used to return the first instance of the data we want to scrape.

Text

Description automatically generated

Similarly, we can use the getall () method to return all the data of a particular tag.

Text

Description automatically generated

A picture containing text

Description automatically generated

We can extract the links using the following command.

Text

Description automatically generated

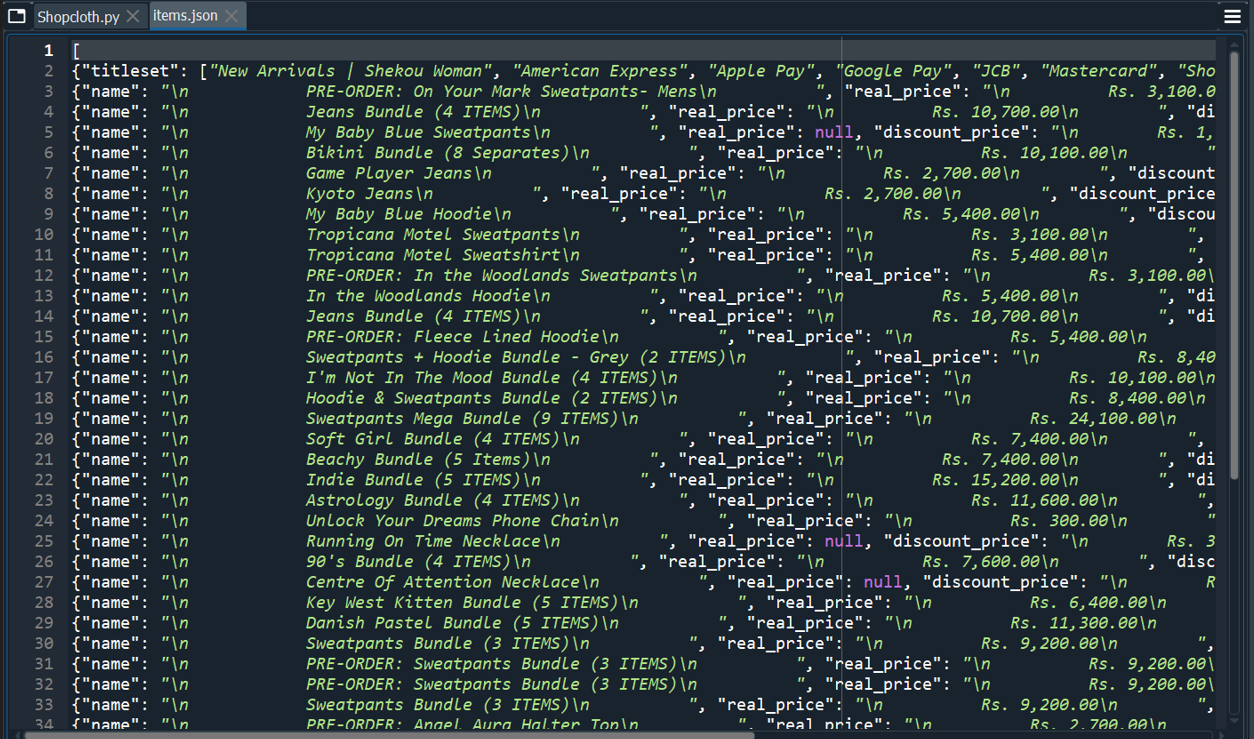
**To scrape the website using spider:**

First, we need to create a .py file in the spider’s folder and then this folder can be used to scrape the website.

**A screenshot of a computer

Description automatically generated**

The command ‘scrapy crawl spidername’ is run on the command prompt to execute the spider. Once the data is scraped the command scrapy crawl spidername -o json/csv is used to save the scraped data in the json/csv format.

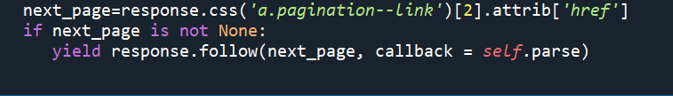


Text, letter

Description automatically generated

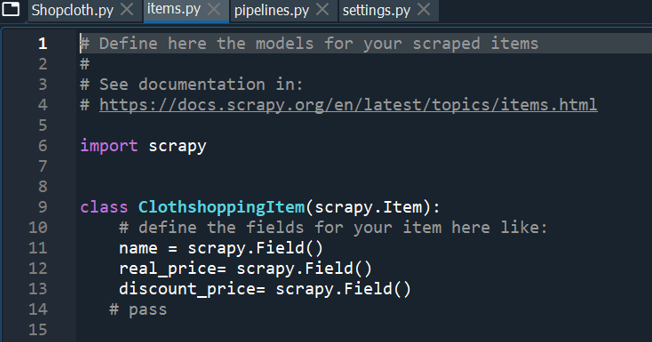
**To iterate through the next pages of the website –**

Used to scrape the subsequent pages.



**Using the Items.py folder as a container –**

The entire website can be scraped and placed into the items.py file which acts like a container. From this file we can then display the results or use this file as an intermediary and send the information to the pipeline and store it in database or display directly.



A screenshot of a computer

Description automatically generated

**Using Pipelines-**

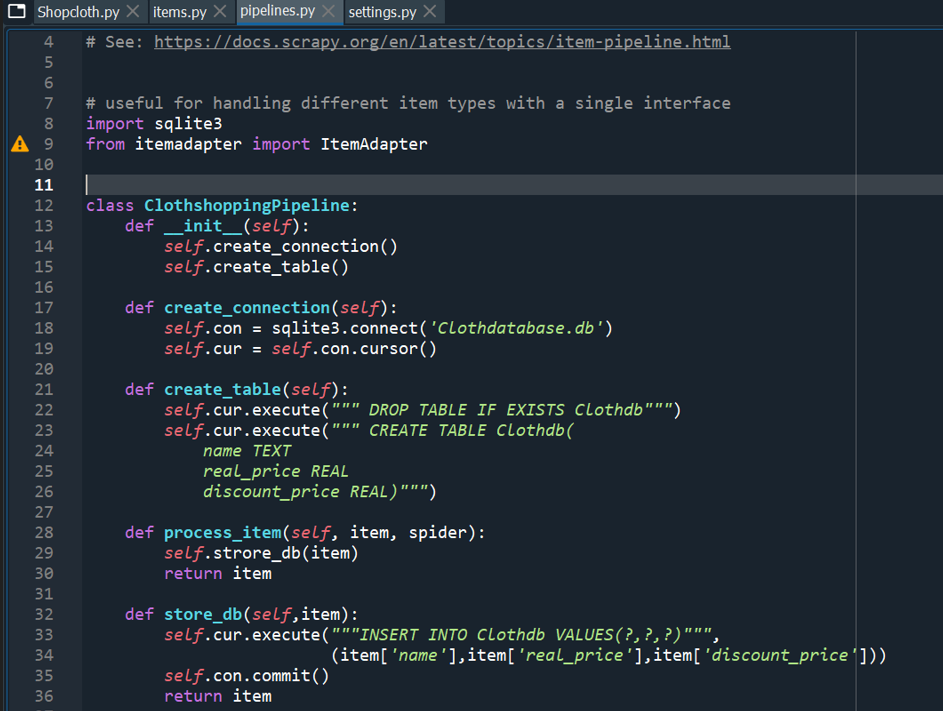
We use pipeline to push the scraped item into the output file/database or to perform any pre-processing steps by making use of middleware.

In the settings.py file, uncomment the code for “Item\_Pipeline”-

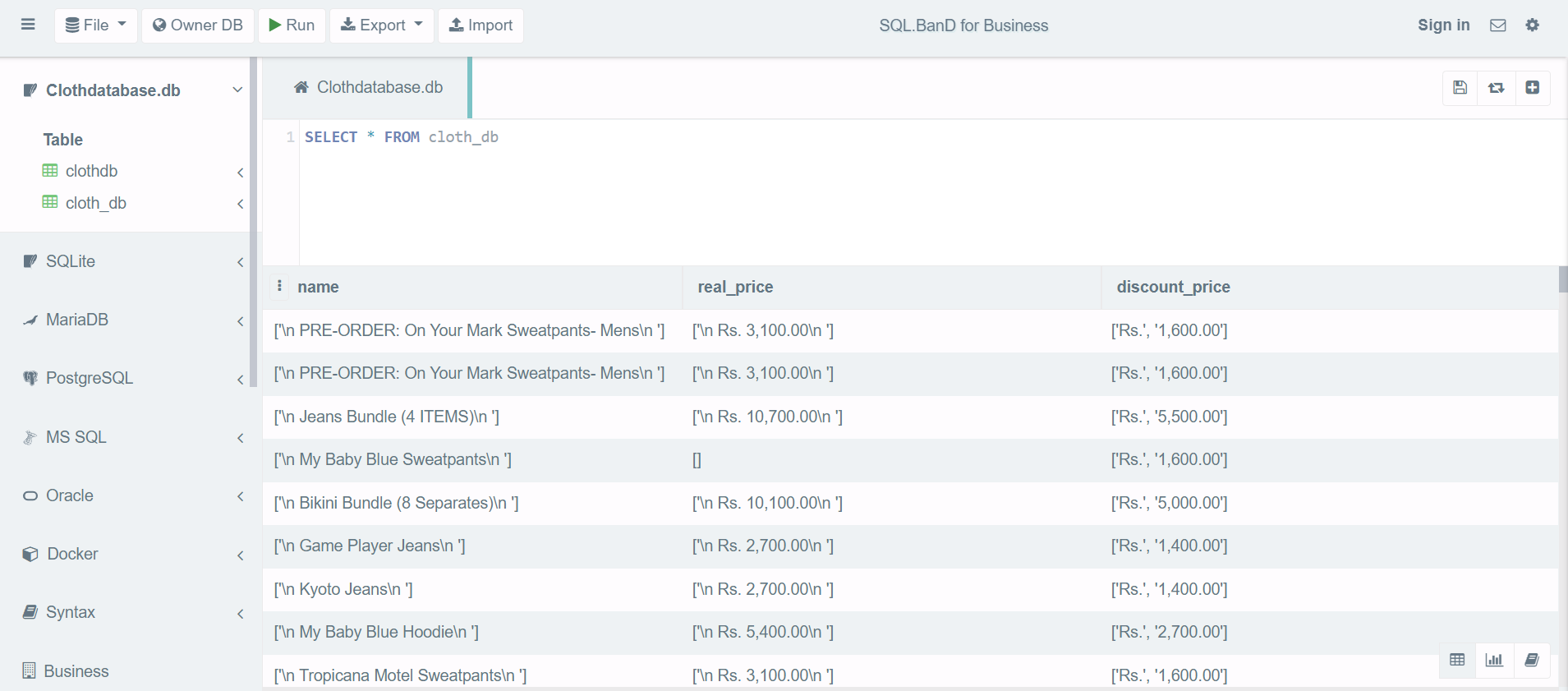
Text

Description automatically generated

In the pipelines.py file, we define classes to create the database –



**Using sqliteonline.com to run and display the database**



The database cloth\_db is created with the columns -name, real\_price and discount\_price. when we use pipelines to scrape the data, Sqliteonline.com can then be used to view the scraped data.

To create an offline database sqlitebrowser.com is used and the database can be downloaded directly from the website.