**Amazon Assignment**

**(Due on 24th April Before Class)**

**For this assignment you must work in pairs. If you do not have a partner let us know ASAP. If you end up working alone your submission will not be graded**

**Objective :**

* Build a parser for parsing Amazon Books on Amazon.
* Store the data in a sqlite databse
* Display the data in a specified format

**Parser Descriptions:**

* Write a class for parsing the webpage. The \_\_init\_\_() method class can take two kinds of arguments : a url or a file name. You can use keyword arguments and/or default arguments to achieve this. If a url is provided, your class should use “requests” to fetch the webpage. If a filename is provided you can directly open the file and start parsing (The file will be an html file). From an amazon book page you have to parse the following:

a) Name of the Book

b) Authors (one or many)

c) Price (“Buy New” and “Rent” both if available)

d) Product Details Section : Everything that is available in this section. (For “Amazon Best Sellers Rank” if provided only pick the rank given in “Books” and the chain of subjects that the book belongs to; The “Average Customer Review” should be just a tuple (float, number of customer reviews).

e) Most Helpful Customer Reviews (Only the ones available on the main page; You do not need to do any crawling): Parse the entire comment.

* The parser should return a json dictionary. In case it fails to parse a certain field it should not fail, rather the corresponding field in the dictionary should be left empty.
* For parsing, look up regex, css-selectors and xpath. You can also look into a module here : [http://www.crummy.com/software/BeautifulSoup/bs4/doc/#quick-start](http://www.crummy.com/software/BeautifulSoup/bs4/doc/" \l "quick-start)

Storing Data in Databases:

As in the previous assignment, you shall store your data in a sqlite3 database. But this time you shall do it using an ORM. Choose either “sqlalchemy” or “storm” for this part. You can write your own schema.

Create the database in the working directory.

Display Data in Different Formats:

As in the previous assignment the output format will be either 'html', 'csv' or 'xml'. You have to show your parsing data in that format.

Deliverables and Execution Instructions:

* Your final executable is called amazon.py
* You shall use argparse module to take command line arguments. It should be able to process the following arguments:
* -d <directory name> : This directory will contain some html files and you need to parse

all the html files and store them in the database.

* -u <url name> : Fetch the webpage specified by the url and store the results in the database.
* -o <html / xml / csv> : Output in the format mentioned. If this argument is not specified you DO NOT need to send any output to stdout.
* -h : Help message showing how to run the executable

Please make sure you have 100% unit test coverage for your code and you are pep8 compliant. Make sure your functions and modules do not have any side effects.

Examples :

$ amazon.py -d /tmp/htmlfiles

Here the executable should pick up all the html files available in the directory “/tmp/htmlfiles” and store the results in the database. (Nothing should be sent to stdout)

$ amazon.py -u http://www.amazon.com/bookurl -o html

Parse the url, store all the data in the database and to the stdout, send the parsed content in html format.

* You should submit a .tar.gz file which can be unzipped using simple tar -xvf command in Unix. This file would contain a directory named “amazon” where your .py files and a readme file containing the name of the team members should exist. Any other format would receive a penalty of -10. Make sure you implement a package for this project.

Please Note:

* The grading for this assignment shall be more stringent, compared to the last two assignments.

Hence make sure you follow all the instructions.

* There shall be extra credit for adding a log functionality using python logging module. If you do it please mention this in your readme file.
* There shall be extra credit for parsing “Customers who bought this also bought” list.