DEVELOPER INTERVIEW ASSIGNMENT 1

SIYAVULA EDUCATION

Task: Implementation of a dynamic web application with Pyramid framework

---

TAKONG RIDICK ROLAND
University of Cape Town,
EGS Department ,
RDCTAK001@myuct.ac.za
takong@aims.ac.za
mobile: +27(0)730814160

February 7, 2015

Abstract

This report contains a brief description of the main things I did. I used the starter scaffold which comes bundle with pyramid and implemented the basic requirements specifications for the web application. With regards to bonus features, I gave the application the ability to produce an error message when an invalid wikipedia URL is entered. I also added automated unit tests for 4 of the 5 methods which I used in my views class.

Contents

1 Introduction

1.1 Problem statement

The following is the problem that was given.

The task consisted in creating a new web application using the Pyramid framework. It was expected that the web application should have the following characteristics

- The homepage must show a welcome message and present the user with an input box that asks for a Wikipedia URL and a submit button.
- When the submit button is clicked, the application must download the HTML source code for the given page and scrape the table of contents for the given page. Keep in mind that some pages may not have a table of contents.
- The table of contents must then be posted back to the user in a new page in html format. Make sure to unescape any escaped html in this step.
- The page showing the table of contents must provide a reset button that will allow the user to restart the process.

Some bonus features which could be included in the web application at my discretion included :

- The application checks for user input that is not a valid Wikipedia URL and posts an error message stating this.
- In the event that a Wikipedia page has no table of contents, a suitable error message is given.
- Each view and method created for this application, has a unit test. The pyramid tutorial shows an example of this.
- The source code is fully PEP8 compliant.

1.2 Development System characteristics

The program was written and tested on an hp pavillion dv4 laptop with the properties stated hereafter.The operating system used is Ubuntu 14.04LTS characterised as follows

```
Linux versatile 3.13.0-44-generic #73-Ubuntu SMP Tue Dec 16 00:22:43 UTC 2014 x86_64 x86_64 x86_64 GNU/

Programming Language : Python 2.7.6
Virtual Environment  : /home/takong/env
```

2 Dependencies and required packages

The following packages are required to compile and run the code:

1. BeautifulSoup 4
2. lxml
3. urllib2
4. pyramid

5. HTMLParser
6. re
7. nose

For details on how to install these packages, please refer to the websites of the respective developers.

3 Solution of Problem

3.1 Assumptions

The main assumptions I made in the development include:

1. The content of each wikipedia page which has a content contains a ***div*** tag whose id attribute is ***toc***. That is

```
<div id="toc" class="toc">
<div id="toctitle">
<h2>Contents</h2>
</div>
```

2. I also assumed that a typical wikipedia URL is of the form:

   `http://en.wikipedia.org/wiki/Dowry`

   Meaning it must contain the keyword ***wikipedia*** between the first two dots.

3.2 Some known issues

1. The program produces a UnicodeDecodeError if the URL `http://en.wikipedia.org/wiki/Aircraft` is inserted.

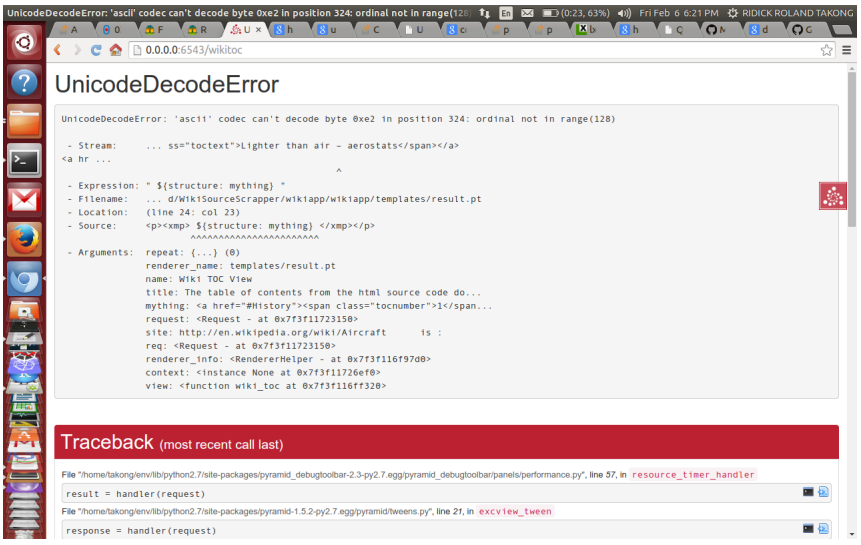

Figure 1: Error (UnicodeDecodeError) obtained when the url `http://en.wikipedia.org/wiki/Aircraft` is inserted inserted

3.3 Project directory structure

The source files and packages are organised as illustrated in the figure below

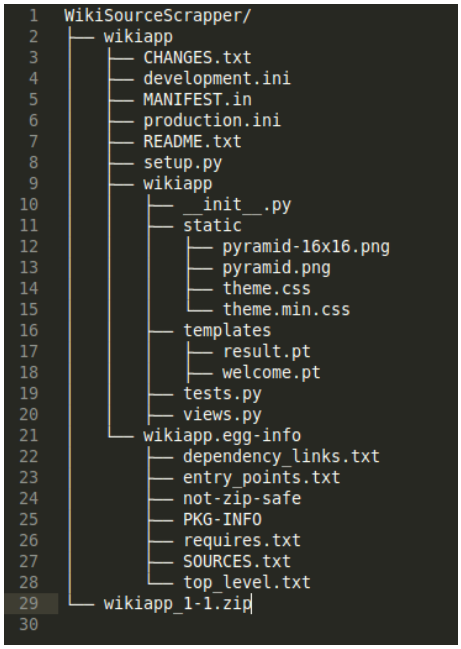

Figure 2: Project structure: Organisation of source files and packages

The starter scaffold was used to create the project.

4 Testing

4.1 Input sample

A number of wikipedia pages were tested and proven to work. Some of these include:

`http://en.wikipedia.org/wiki/Dowry`
`http://en.wikipedia.org/wiki/Mango`
`http://en.wikipedia.org/wiki/Bride_price`

The welcome screen obtained upon starting the application is given in figure 3

By inserting the `http://en.wikipedia.org/wiki/Dowry` as illustrated in 4, we obtain the corresponding output showed in figure 7

4.2 Output Sample

Some sample outputs are displayed below:

4.3 Input and output obtained with non-wikipedia url

- If NO URL or a non wikipedia URL such as `www.google.com` is entered as illustrated in figure 8, We get the output displayed in figure 9.

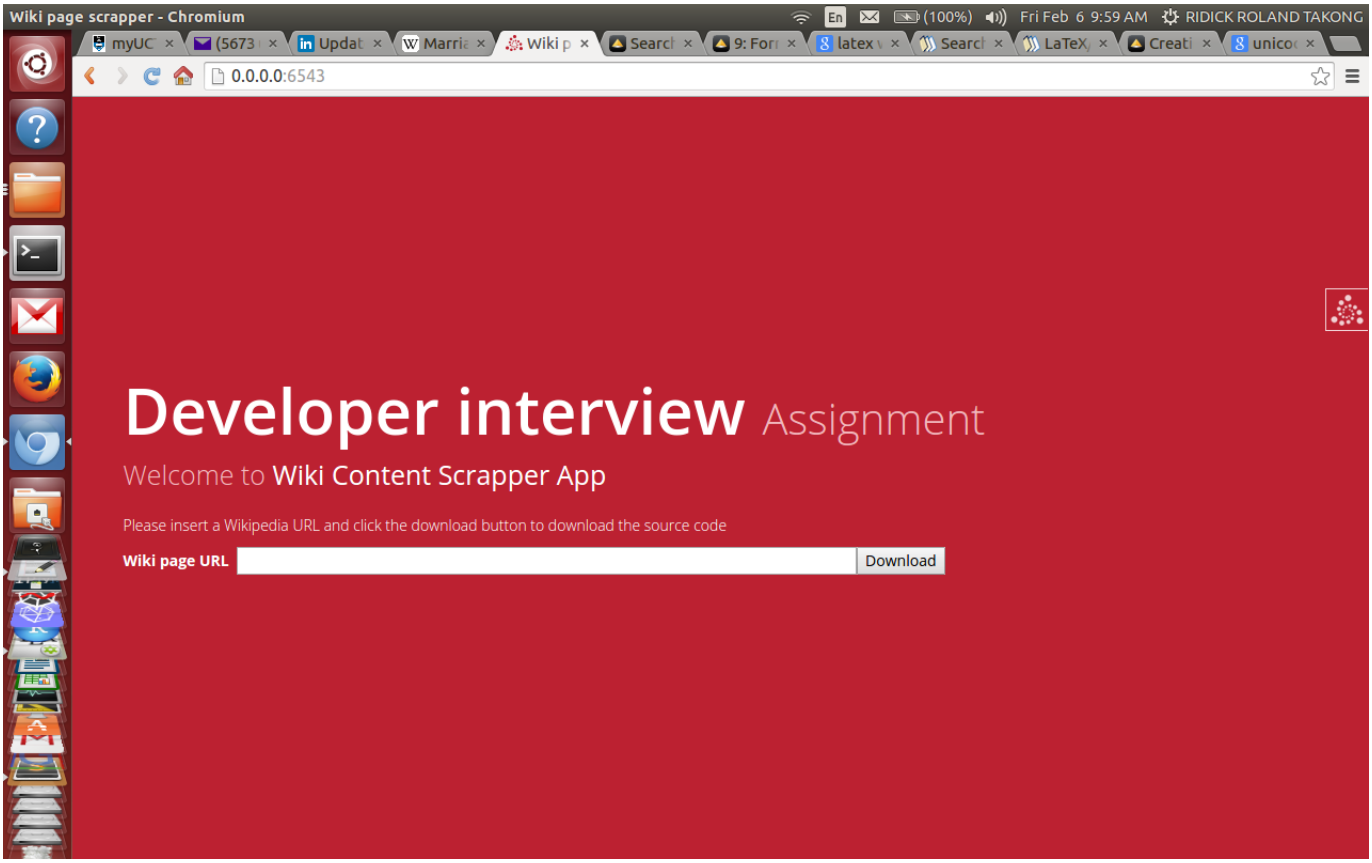

Figure 3: Welcome screen obtained when the application is started

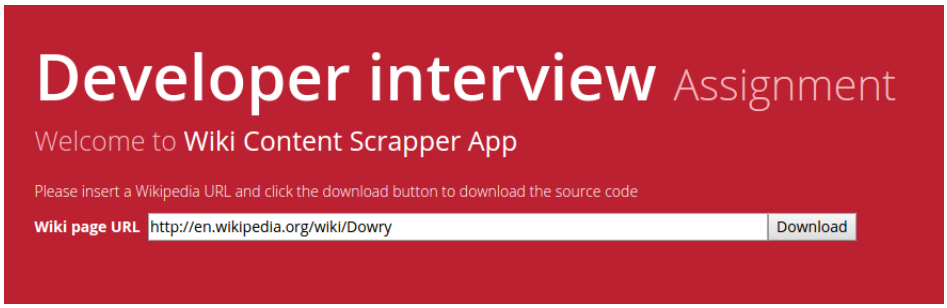

Figure 4: Sample input: `http://en.wikipedia.org/wiki/Dowry`

4.4 Unit Test results

The automated unit tests file, ***tests.py*** which is generated with the starter scaffold was modified to take into account the methods which are used in the *views.py* file. The results of the unit tests for the methods is shown in below:

```
OK
takong@versatile:~/Dev_zone/Pyramid/WikiSourceScrapper/wikiapp$ $VENV/bin/nosetests wikiapp
....
----------------------------------------------------------------------
Ran 4 tests in 1.676s
```

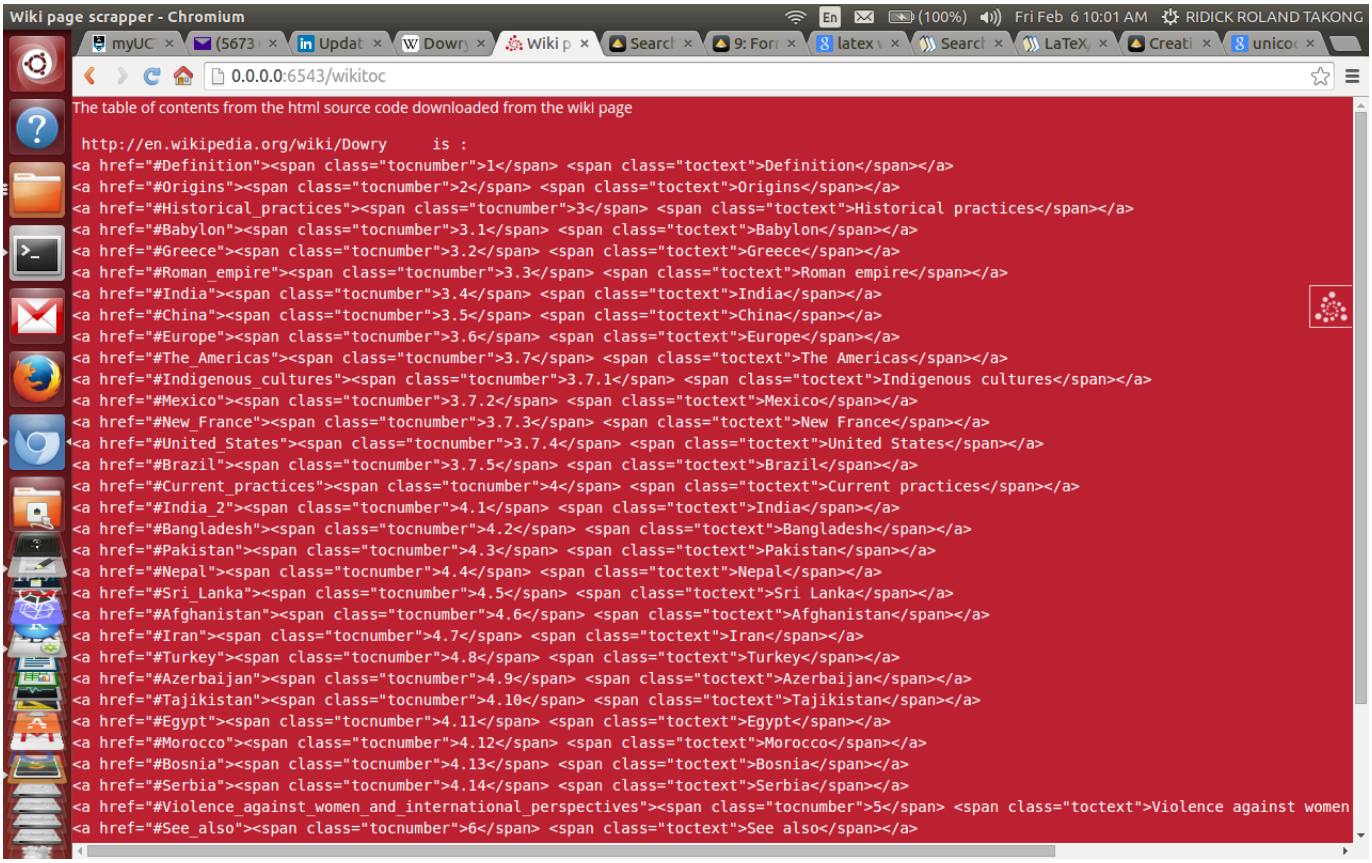

Figure 5: Full screen view: html source code of the table of content of the wikipedia page located at `http://en.wikipedia.org/wiki/Dowry`

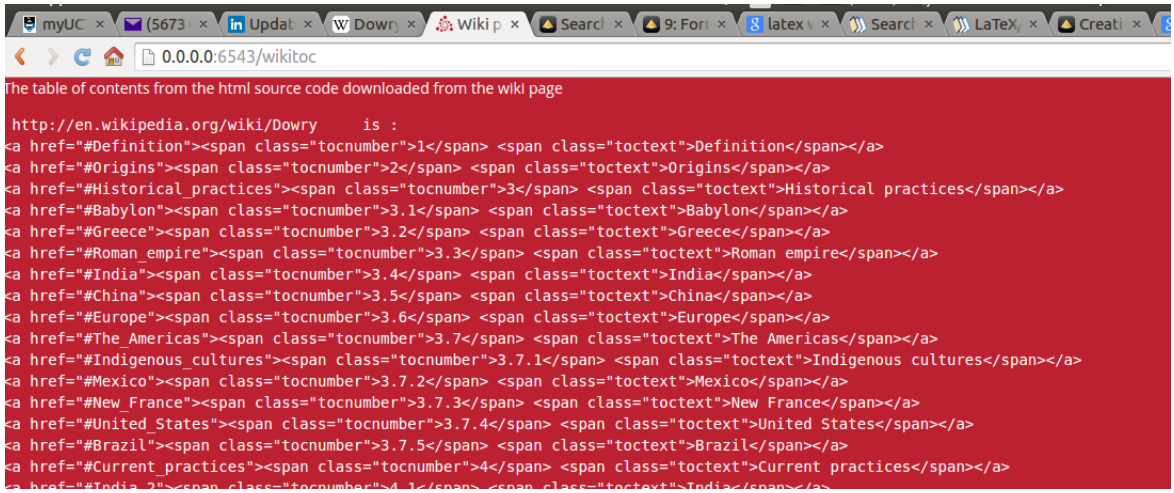

Figure 6: Top screen view: html source code of the table of content of the wikipedia page located at `http://en.wikipedia.org/wiki/Dowry`

Figure 7: Bottom screen view: html source code of the table of content of the wikipedia page located at `http://en.wikipedia.org/wiki/Dowry`

Figure 8: Non-wikipedia page url inserted `www.google.com`

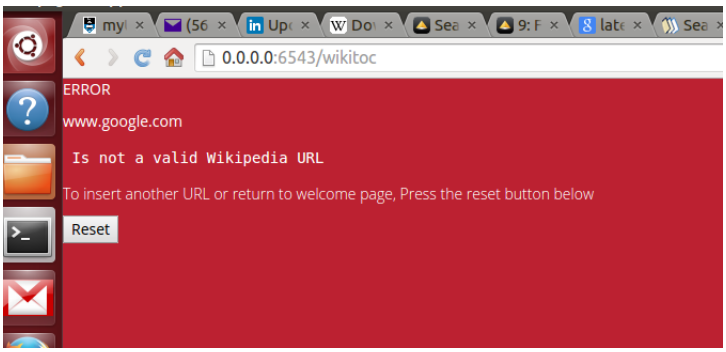

Figure 9: Error message displayed when a non-wikipedia page url eg. `www.google.com` is inserted

```
OK
takong@versatile:~/Dev_zone/Pyramid/WikiSourceScrapper/wikiapp$ $VENV/bin/nosetests wikiapp
....
----------------------------------------------------------------------
Ran 4 tests in 1.635s

OK
takong@versatile:~/Dev_zone/Pyramid/WikiSourceScrapper/wikiapp$
```

To run the unit test, type the following command in the terminal.

```
nosetests wikiapp
```

If you are running from a virtual environment use

```
$VENV/bin/nosetests wikiapp
```

For this to work you must be in the wikiapp directory, whose content include:

```
CHANGES.txt
development.ini
MANIFEST.in
production.ini
README.txt
setup.py
wikiapp
wikiapp.egg-info
```

5 Codes

For brevity reasons, the codes have not been included here. The methods and the codes are commented to facilitate understanding

6 How to run the Web application on Linux

I have only include the guidelines for linux here because siyavula uses only Linux and Mac operating systems. In addition, I used Linux for development.

1. Uncompress the zip archive containing the application.
2. Using a terminal, Change directory to the root folder of the application. Make sure the directory you are found in contains the following files

   ```
   CHANGES.txt
   development.ini
   MANIFEST.in
   production.ini
   README.txt
   setup.py
   ```

3. make sure your computer is connected to the internet else you will get an error from the urllib2.
4. Start the application server. If you are using a virtual environment to run the application, simply type

   ```
   $VENV/bin/pserve production.ini --reload
   ```

   Where the *VENV* is the environmental variable for your virtual environment.

   Otherwise, simply type

   ```
   pserve production.ini --reload
   ```

   Once the server is started you get the following message in the terminal.

```
takong@versatile:~/Dev_zone/Pyramid/\
WikiSourceScrapper/wikiapp$ $VENV/bin/pserve production.ini --reload
Starting subprocess with file monitor
Starting server in PID 11784.
serving on http://0.0.0.0:6543
```

This confirms that the application has been deployed and can be accessed from a web browser through the url `http://0.0.0.0:6543` or `http://localhost:6543`.

5. Open your web browser and insert the url `http://0.0.0.0:6543`.

NOTE That for this to work, your system must have python, pyramid and the packages earlier mentioned installed.