**Homework 4**

**Steps**:-

**Setting up VMWarePlayer and Ubuntu**

1. Installed VMWarePlayer and installed Ubuntu. Installed vmware tools on Ubuntu.
2. Installed Java OpenJDK on Ubuntu

**Setting up Solr**

1. Downloaded solr from an apache mirror
2. Go to solr-6.5.0 directory
3. Run bin/solr start
4. Run bin/solr create -c hwcore

Creating a core

1. Made changes to managed-schema.xml of hwcore as described in the “IndexWithTikaV3.pdf” file
2. Run bin/post -c core hwcore -filetypes html /hw4data

This creates the index on the data sets in Solr. We perform queries on this index.

1. Made changes to solr-config.xml of hwcore as described in the “IndexWithTikaV3.pdf” file

**Generate Page Rank file**

1. First, we read the mapping file line by line to extract the links. (We use the JSOUP library to extract links from the web pages that we downloaded)

Extraction method : link.attr(“href”).trim();

1. We now utilize the map file and the data to store the extracted links in the form of an edgelist in a temporary file. Let’s call the file “edgelist”
2. Install python on Ubuntu.
3. Install networkx library using pip (pip comes packaged with python)
4. Create a directed graph using the edgelist
5. We use networkx to create a digraph from the edgelist

nx.DiGraph(nx.read\_edgelist("/path-to-folder/edgeList.txt")

1. We use networkx pagerank API to compute the pagerank from the digraph created in the above step

nx.pagerank(digraph,parameter1,parameter2……)

List of Parameters:

alpha=0.85

personalization=None

max\_iter=30

tol=1e-06

nstart=None

weight='weight'

dangling=None

1. We iterate over the values returned by the pagerank API to generate the pagerank file
2. Format of the file is docID:value

**Including Pagerank in SOLR**

1. The pagerank file is named as external\_pagerank.
2. Move the file to hwcore/data directory
3. Made changes to managed-schema.xml and solr-config.xml as per the instructions

**Creation of the SOLR client using angularjs and node.js**

1. Installed node.
2. Created an empty project directory.
3. Run npm init in the directory
4. Created the required files and folders. Added Dependencies to package.json for a simple express app
5. Run npm install to get required dependencies.
6. Run npm install solr-node
7. Wrote the code for the project

**How to run the Project**

1. Go to the node client folder
2. The server.js file consists of post request calls to solr
3. Public/index.html folder consists of the web page structure and the angular code as well
4. Inside the node client directory

Run node server.js

1. Open localhost:8080 in any browser and you should see the browser

**Using your cores:**

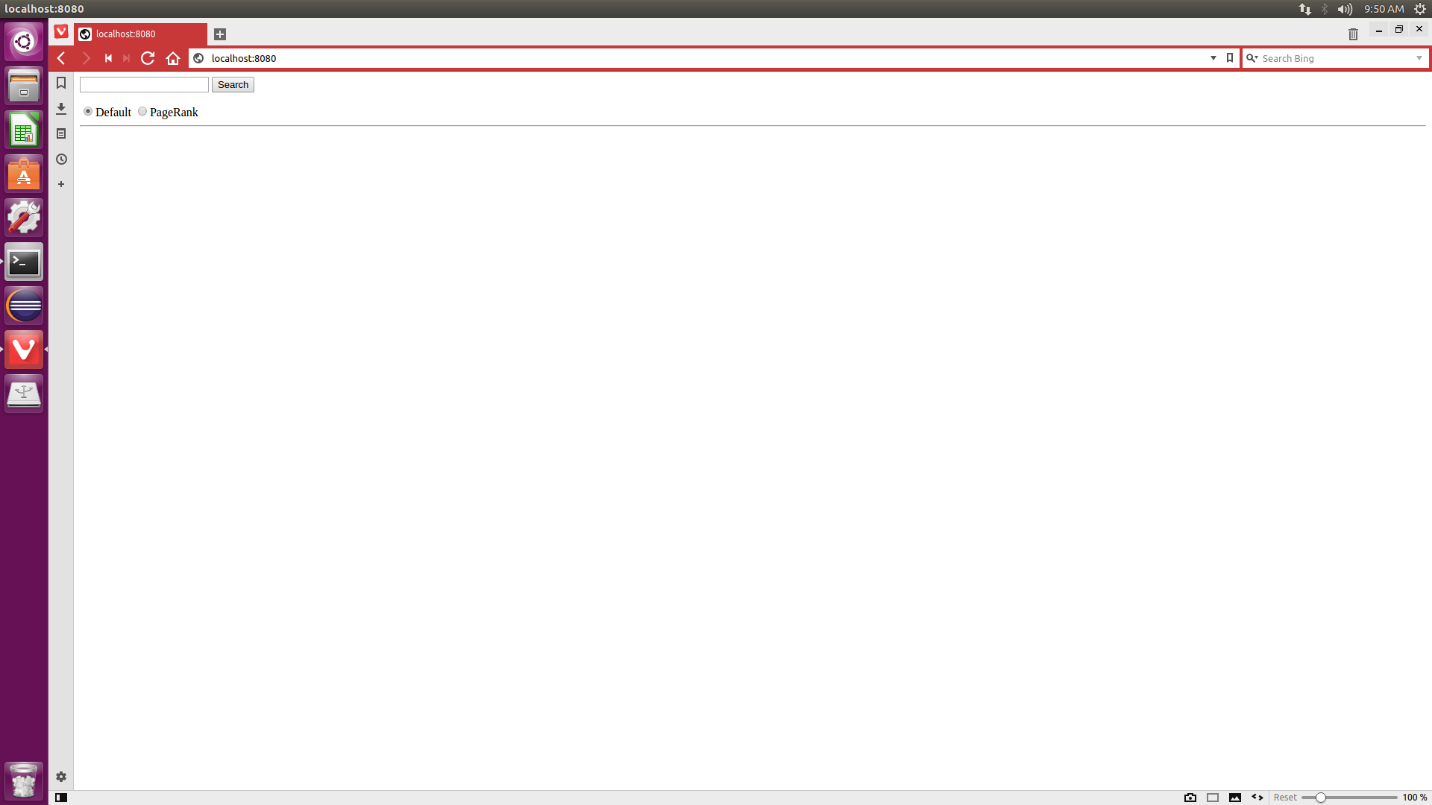
**Inside server.js, inside there are two instances of the variable “url”, one in each post request handler. You may change the name of the core from “hwcore” to your preference. Be careful in modifying as incorrect modification can cause the post request to not work properly.**

**Performance of Lucene vs Pagerank**

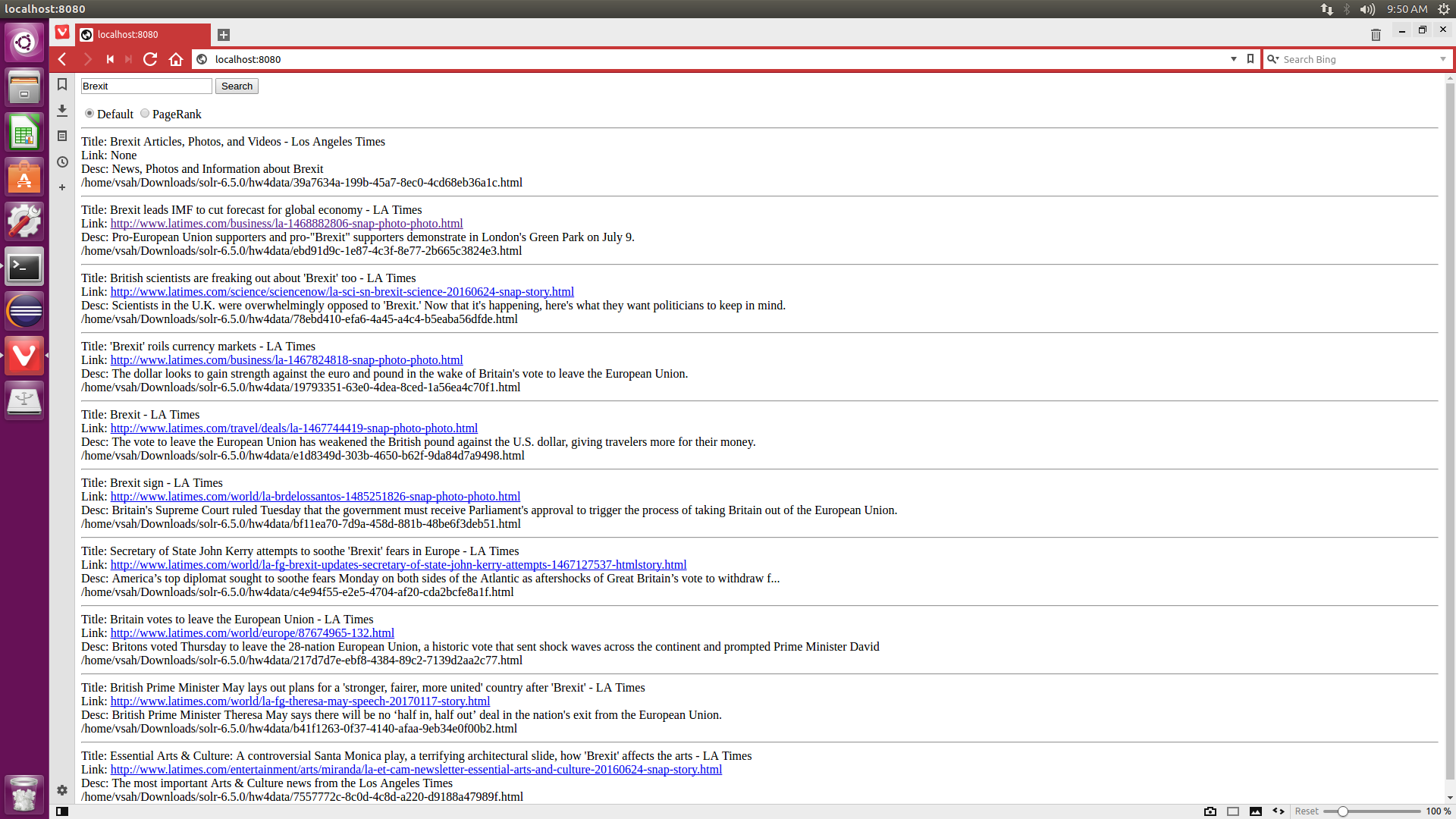
* Lucene outperforms pagerank considerably.
* Lucene utilizes Vector space models and Boolean model to determine the relevance of a page to a query.
* Pagerank on the other hand considers only incoming and outgoing links in pagerank computation.
* In other words, Lucene takes relevance into account, while pagerank doesn’t care about relevance
* Therefore, irrelevant pages with a high page rank show up while using the pagerank algorithm

**FLOW OF THE SOLR CLIENT**

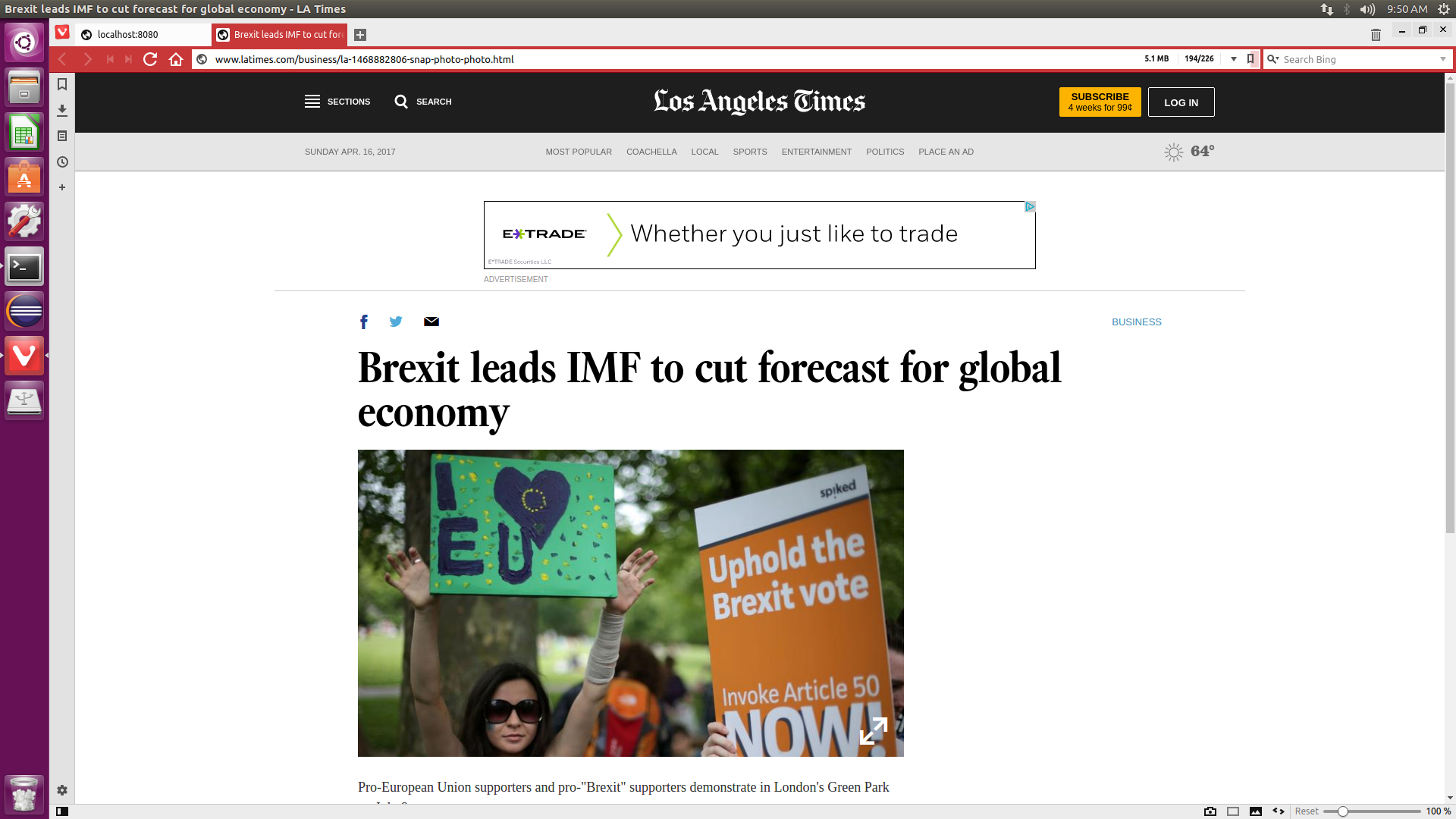
1. Home page of Solr Client



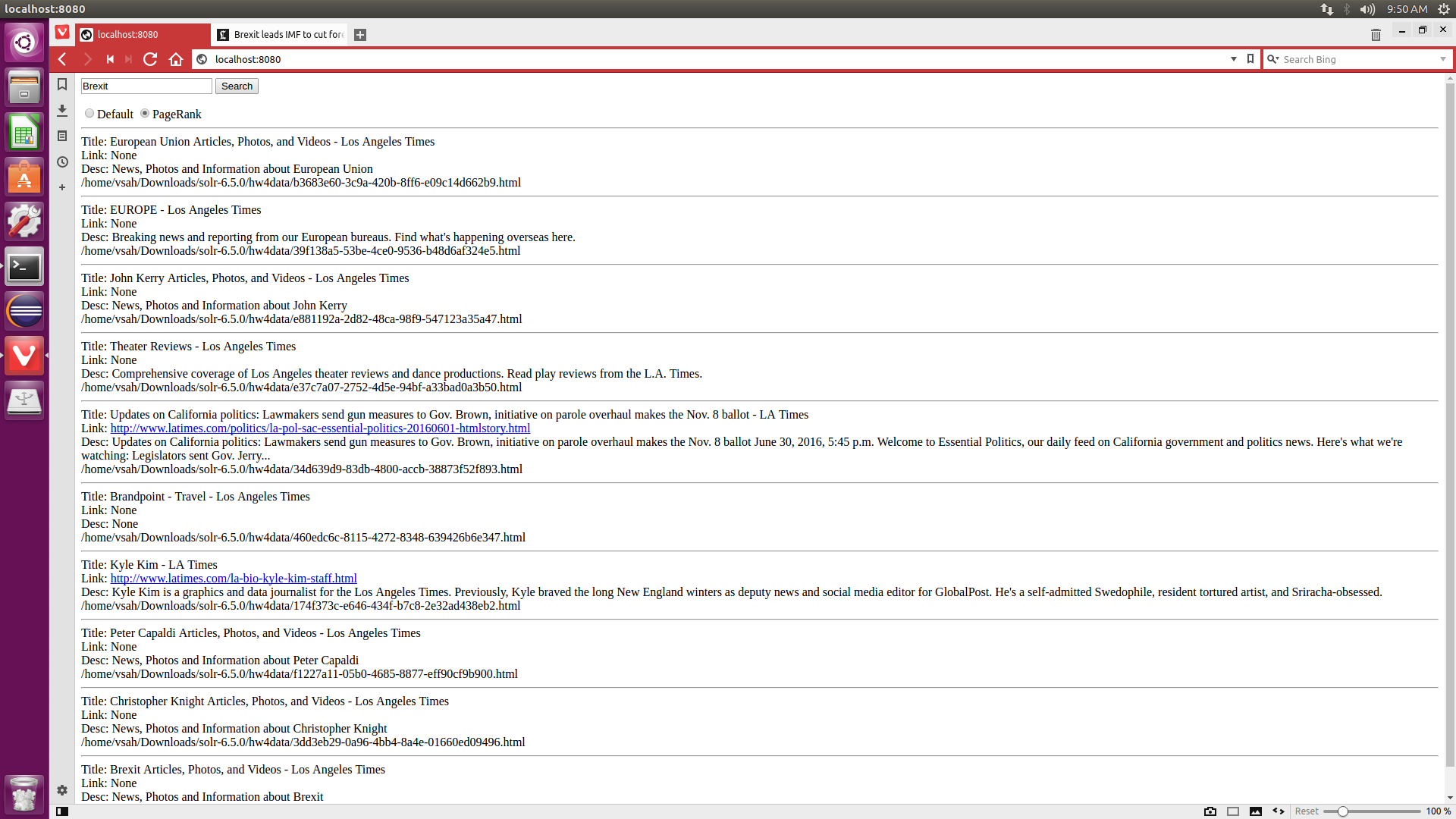
1. Type query (sample: Brexit) and press search. Default selection is Lucene. 10 results are returned.



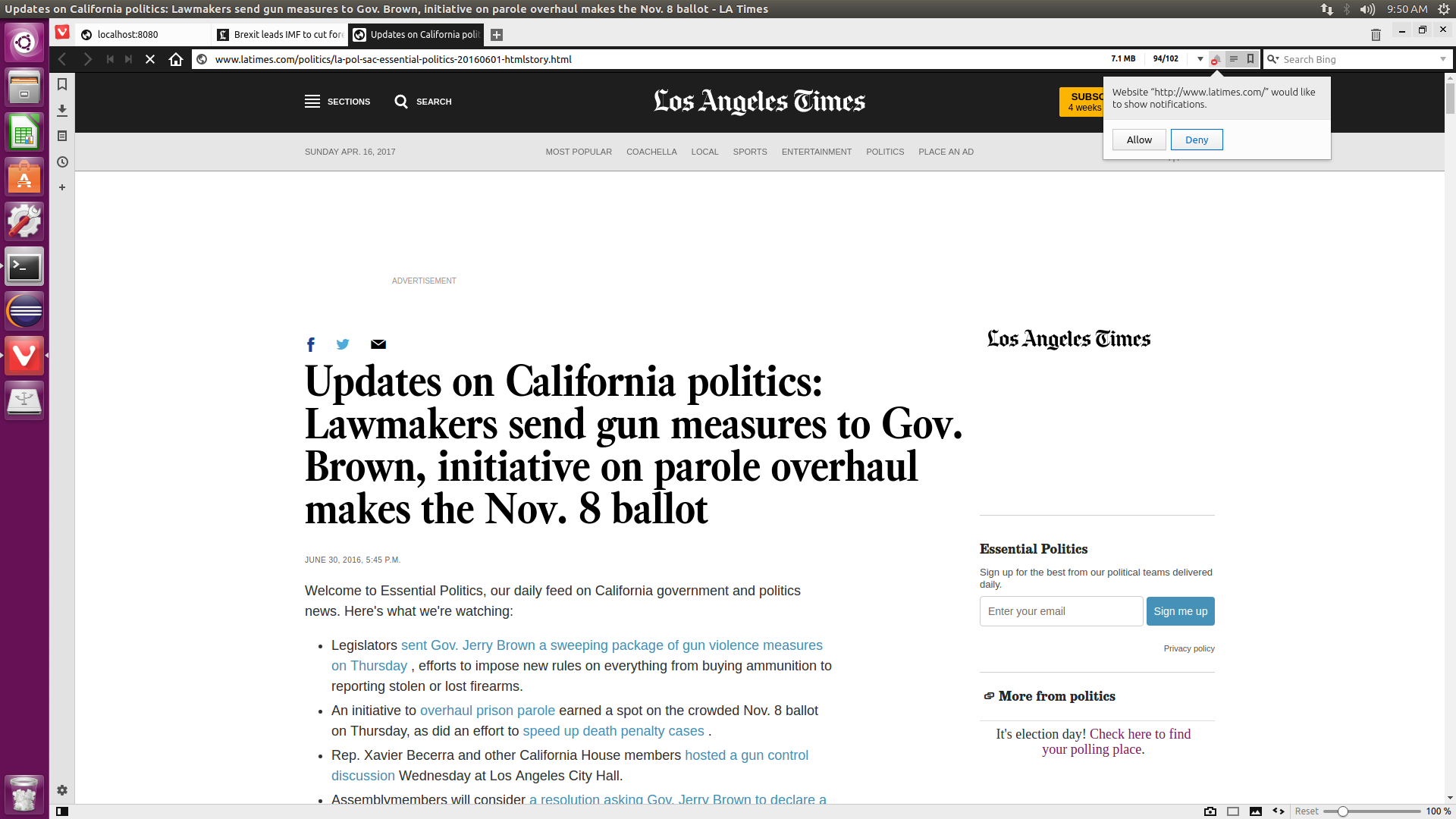
1. We open 2nd result using the link provided <http://www.latimes.com/business/la-1468882806-snap-photo-photo.html>



1. We now use Pagerank to get the results

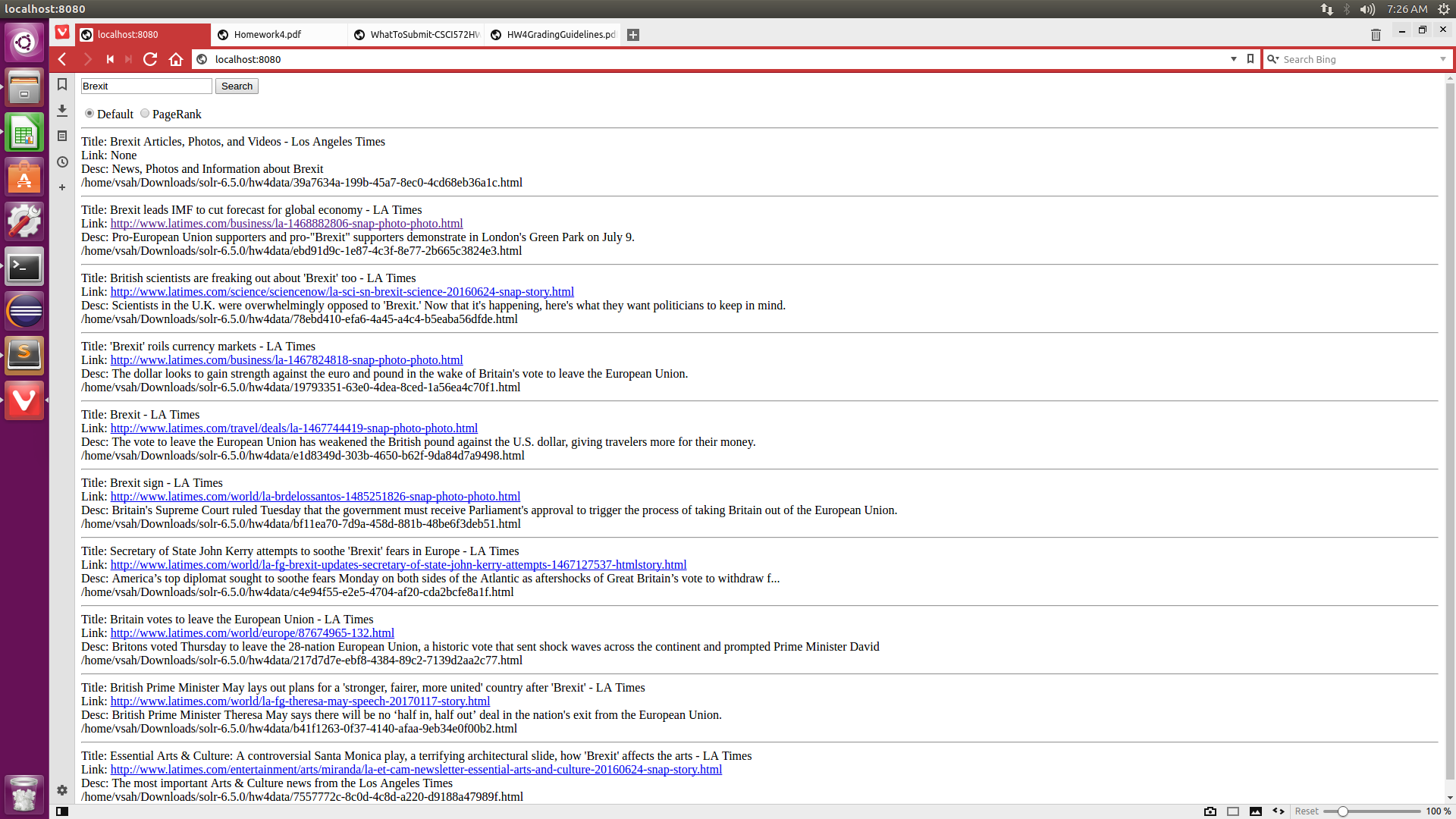


1. We open the 5th result using the link provided <http://www.latimes.com/politics/la-pol-sac-essential-politics-20160601-htmlstory.html>

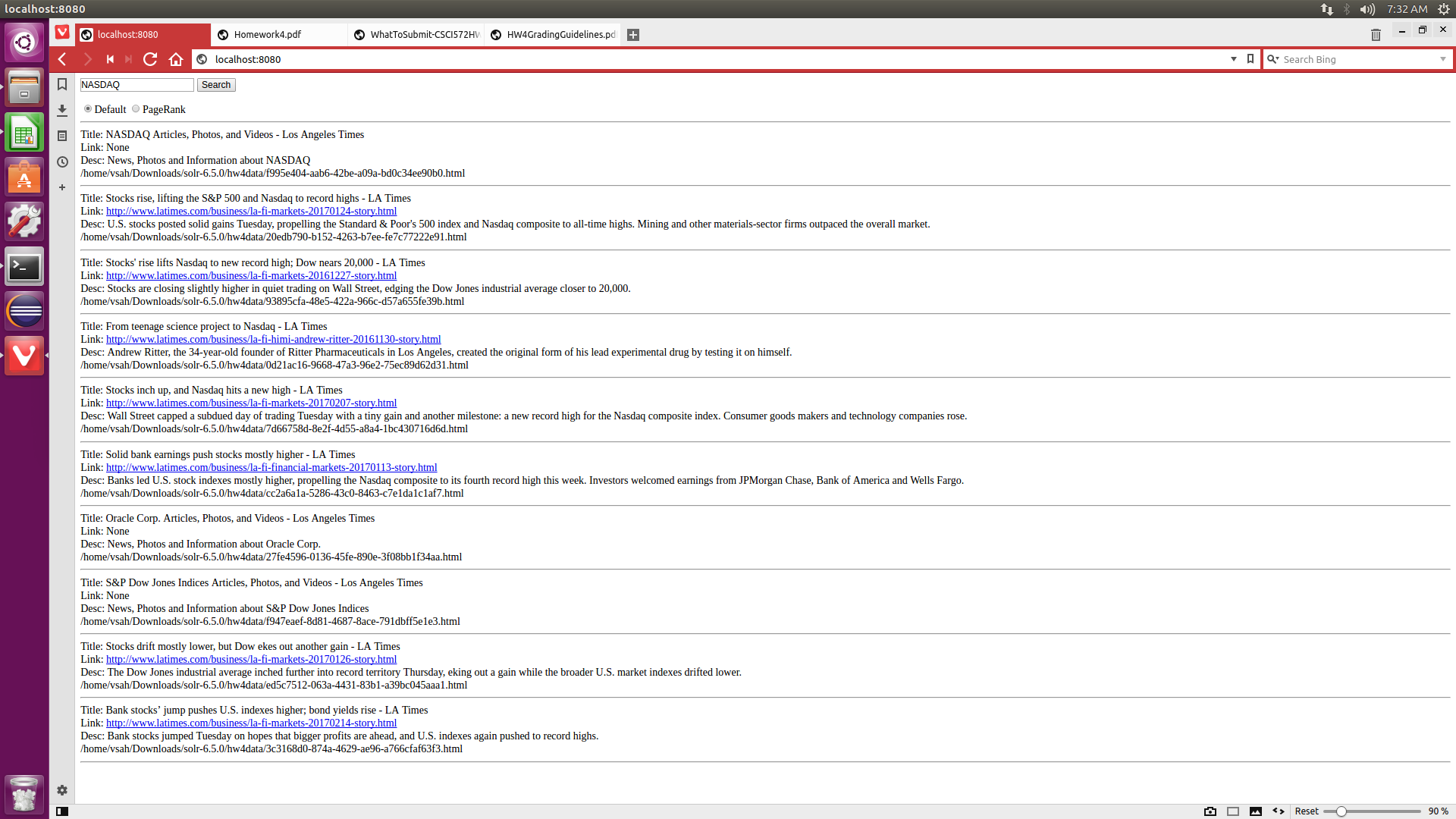


**Query Results (Lucene)**

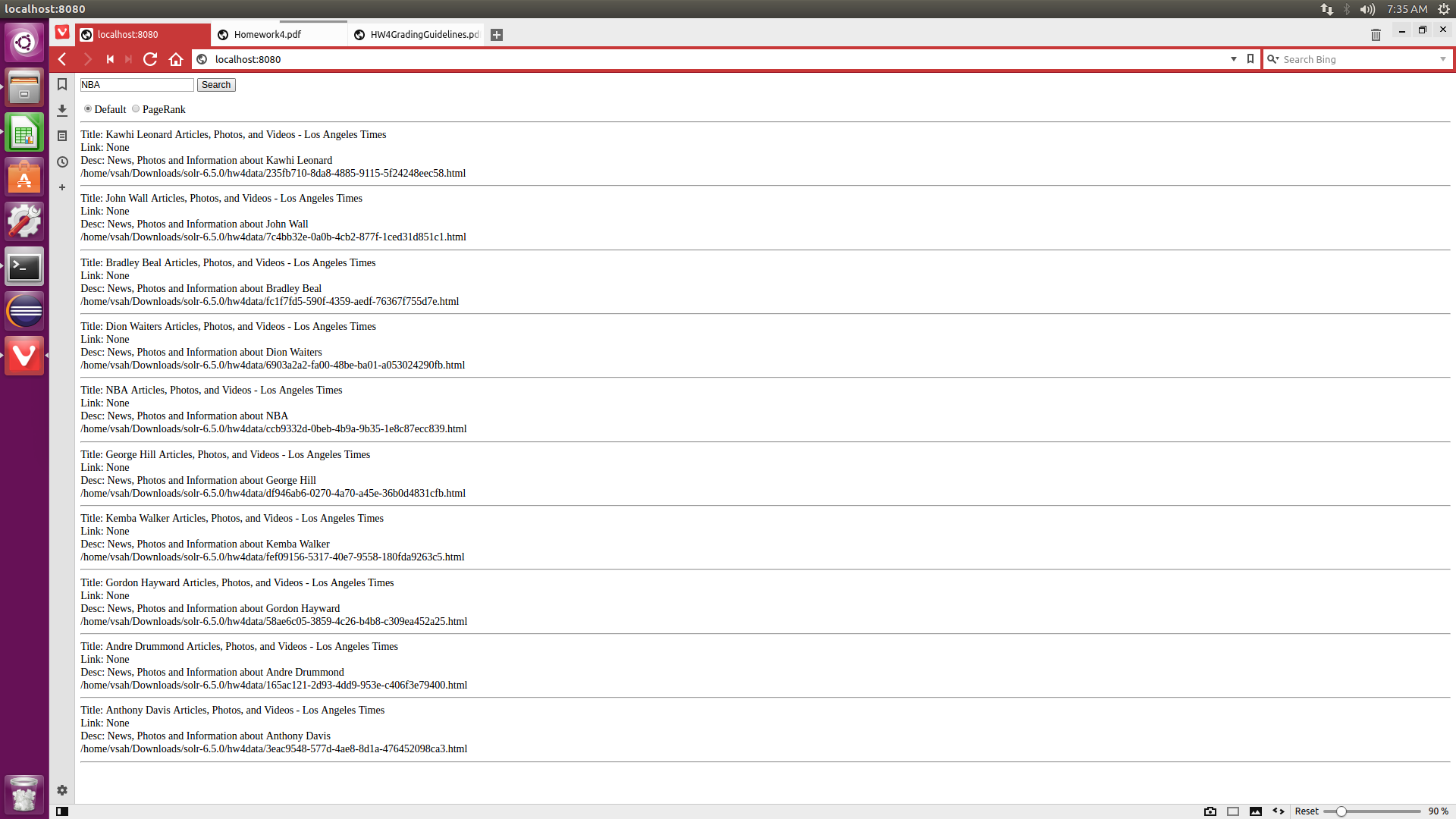
**BREXIT**



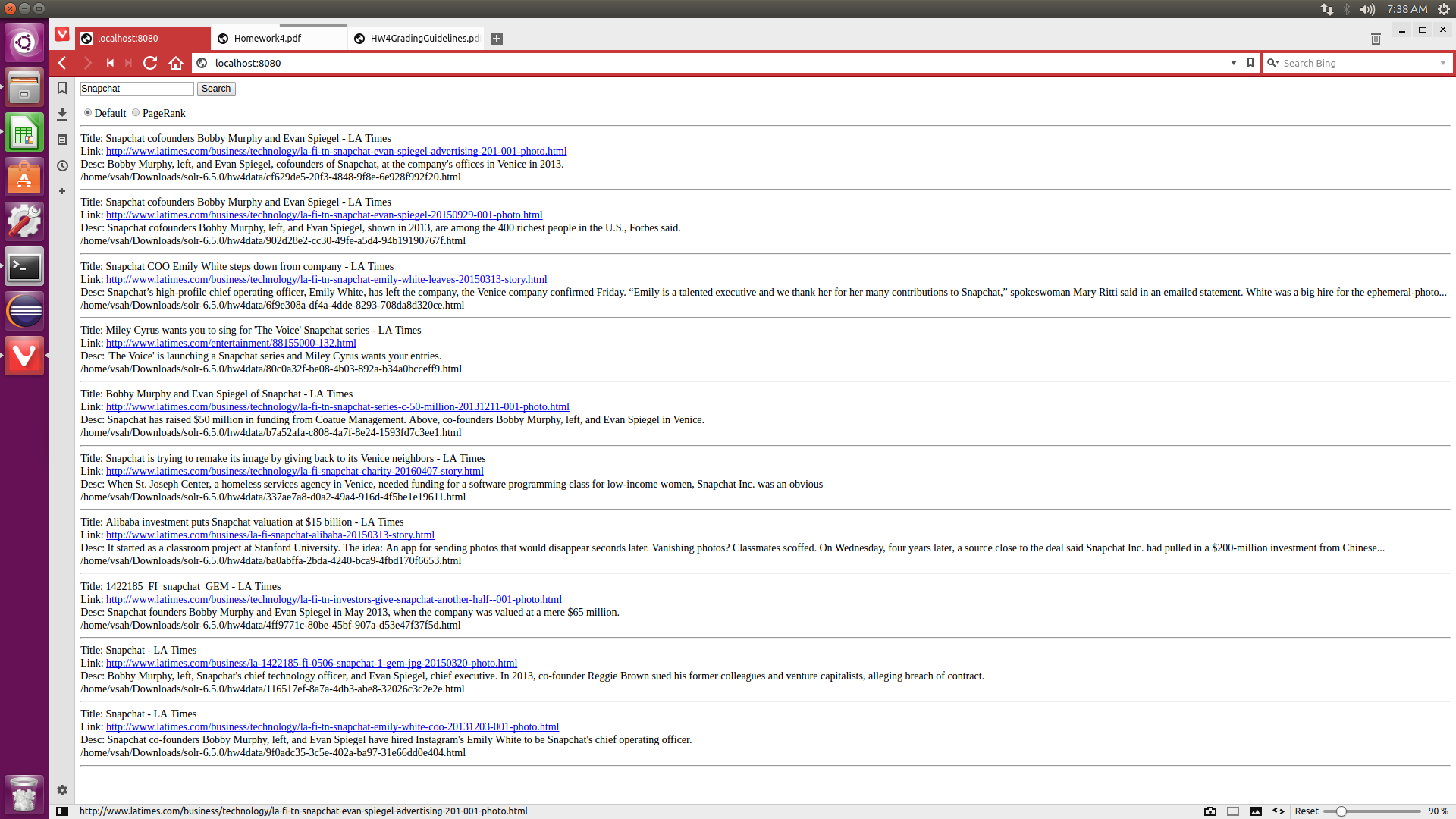
**NASDAQ**



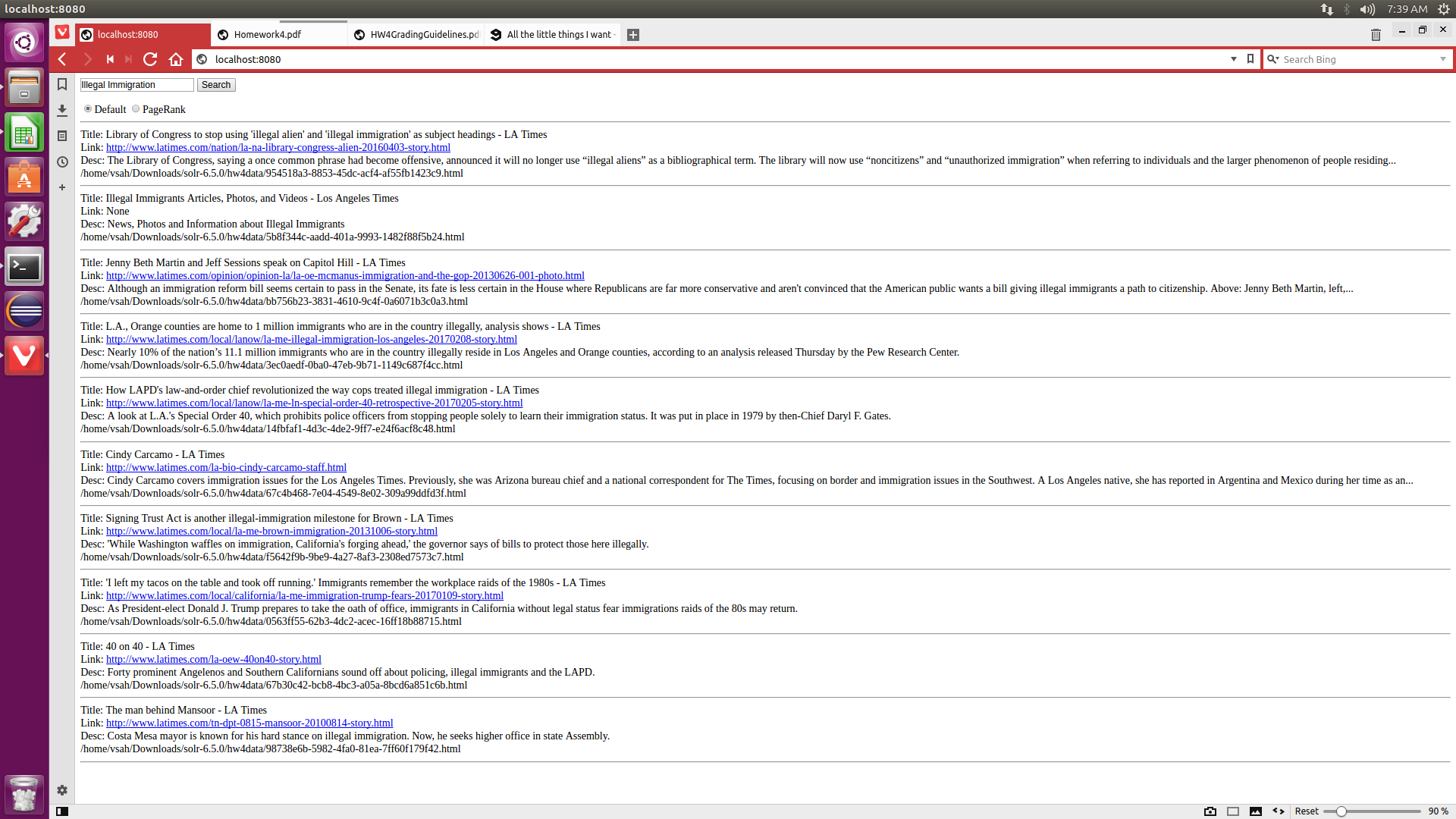
NBA



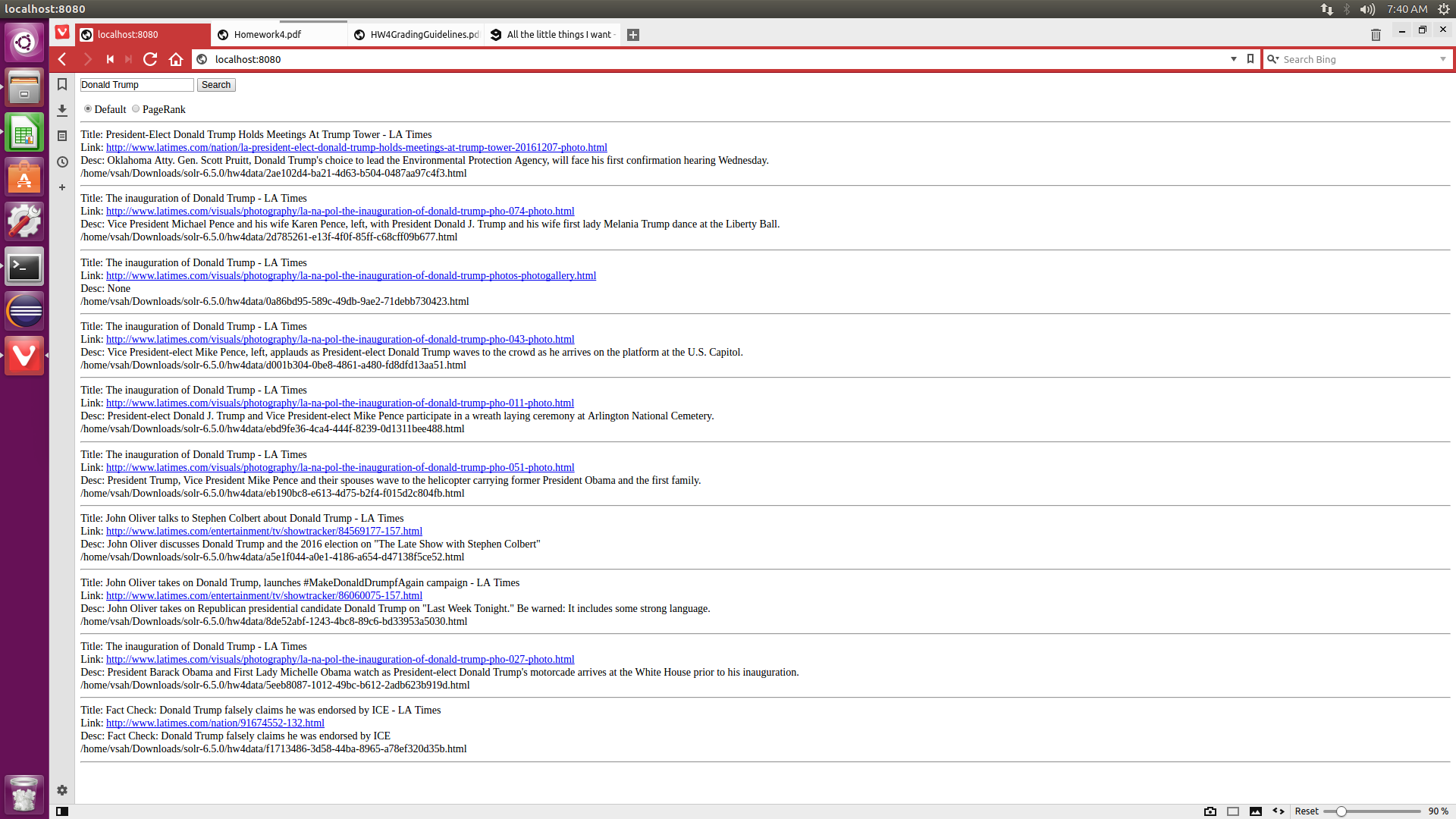
**Snapchat**



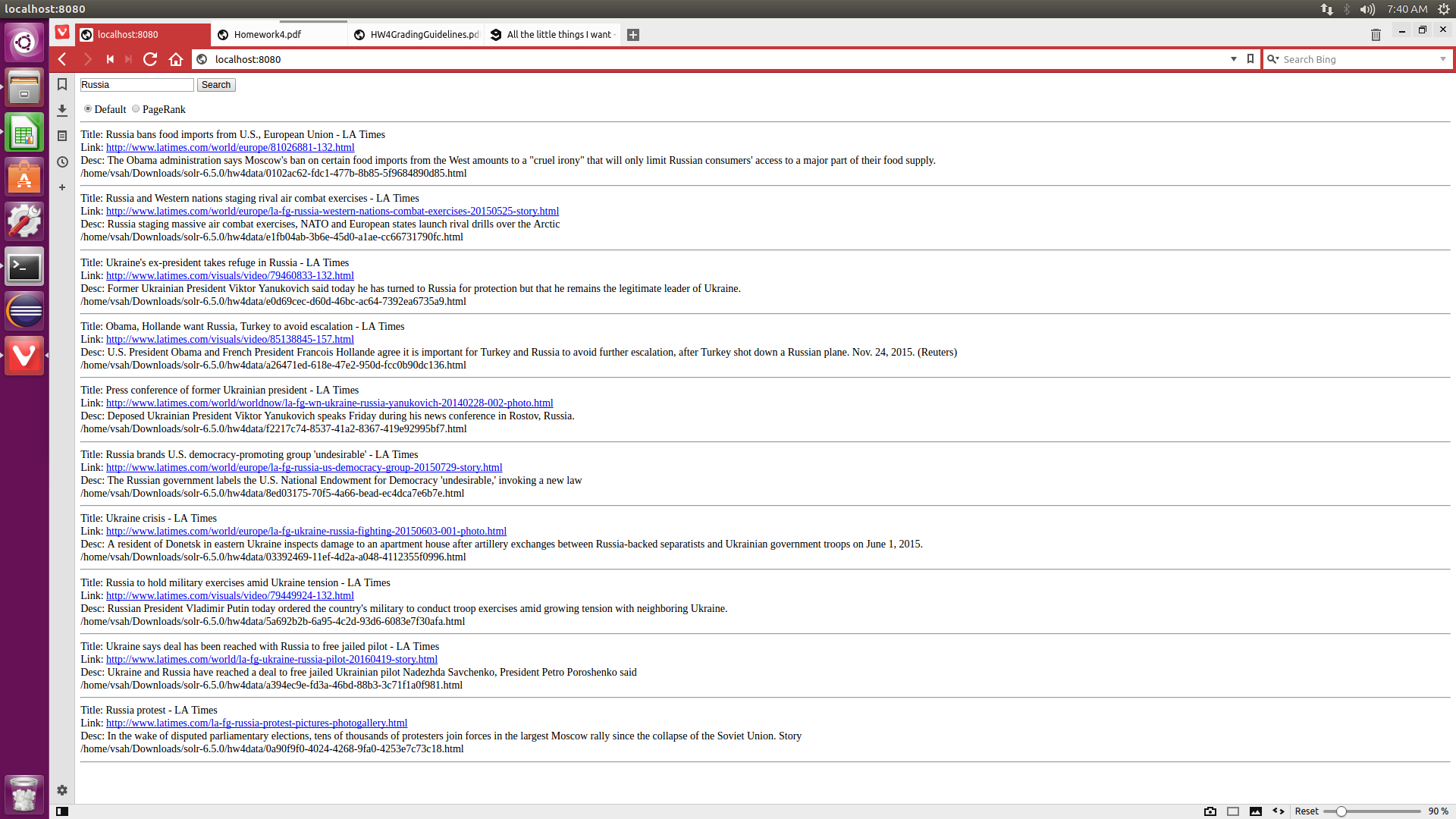
**Illegal Immigration**



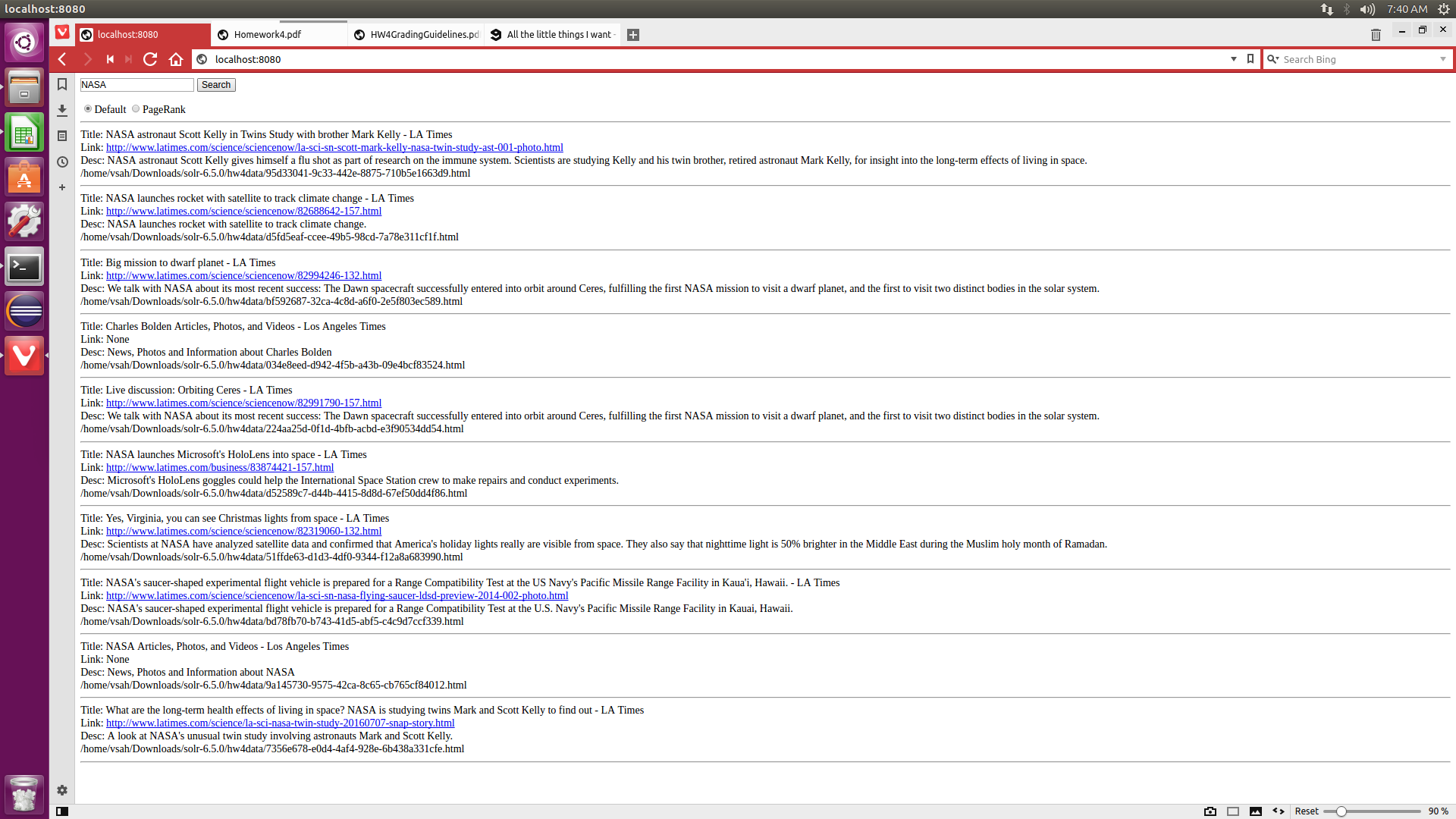
**Donald Trump**



**Russia**

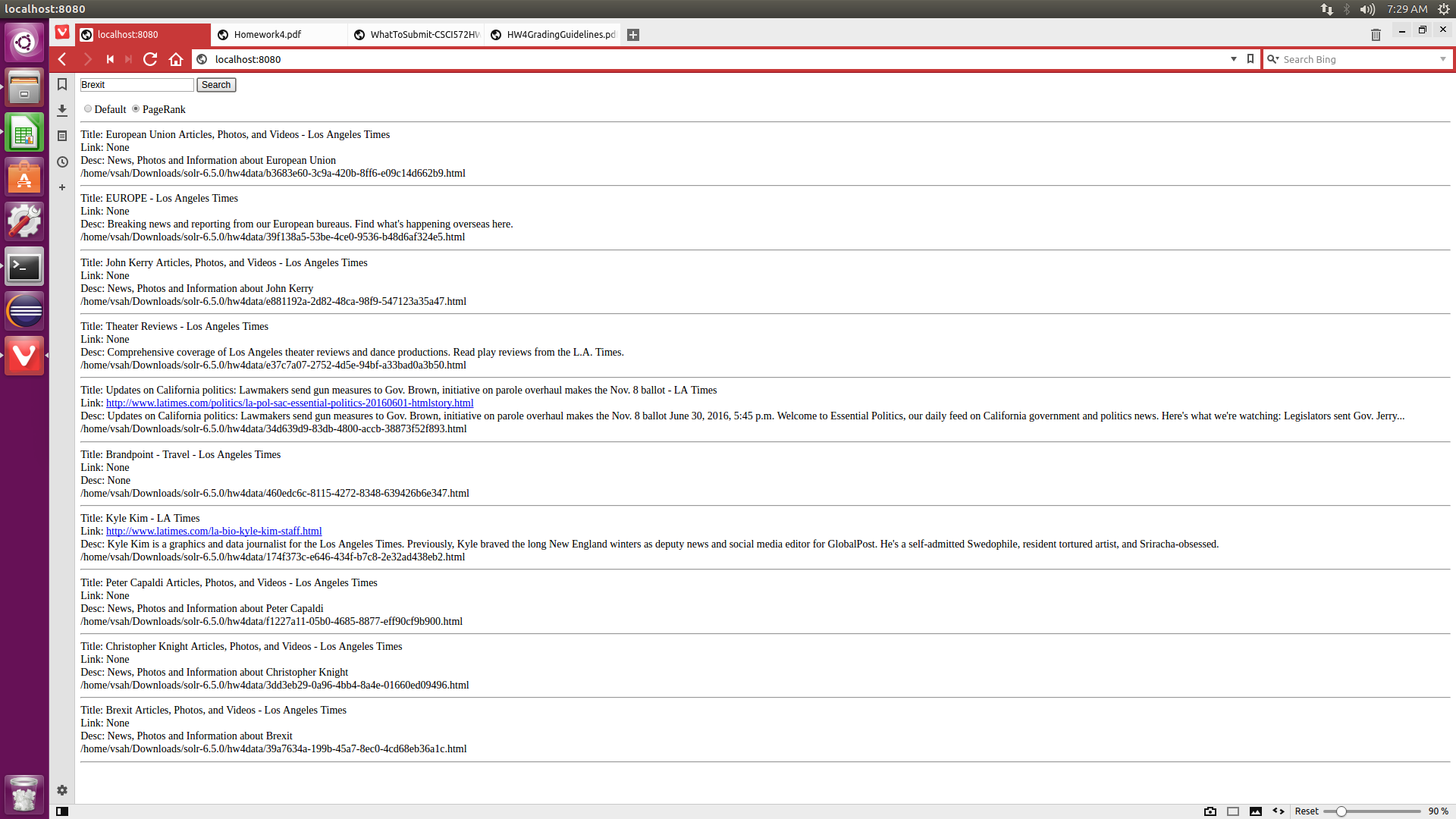


**NASA**

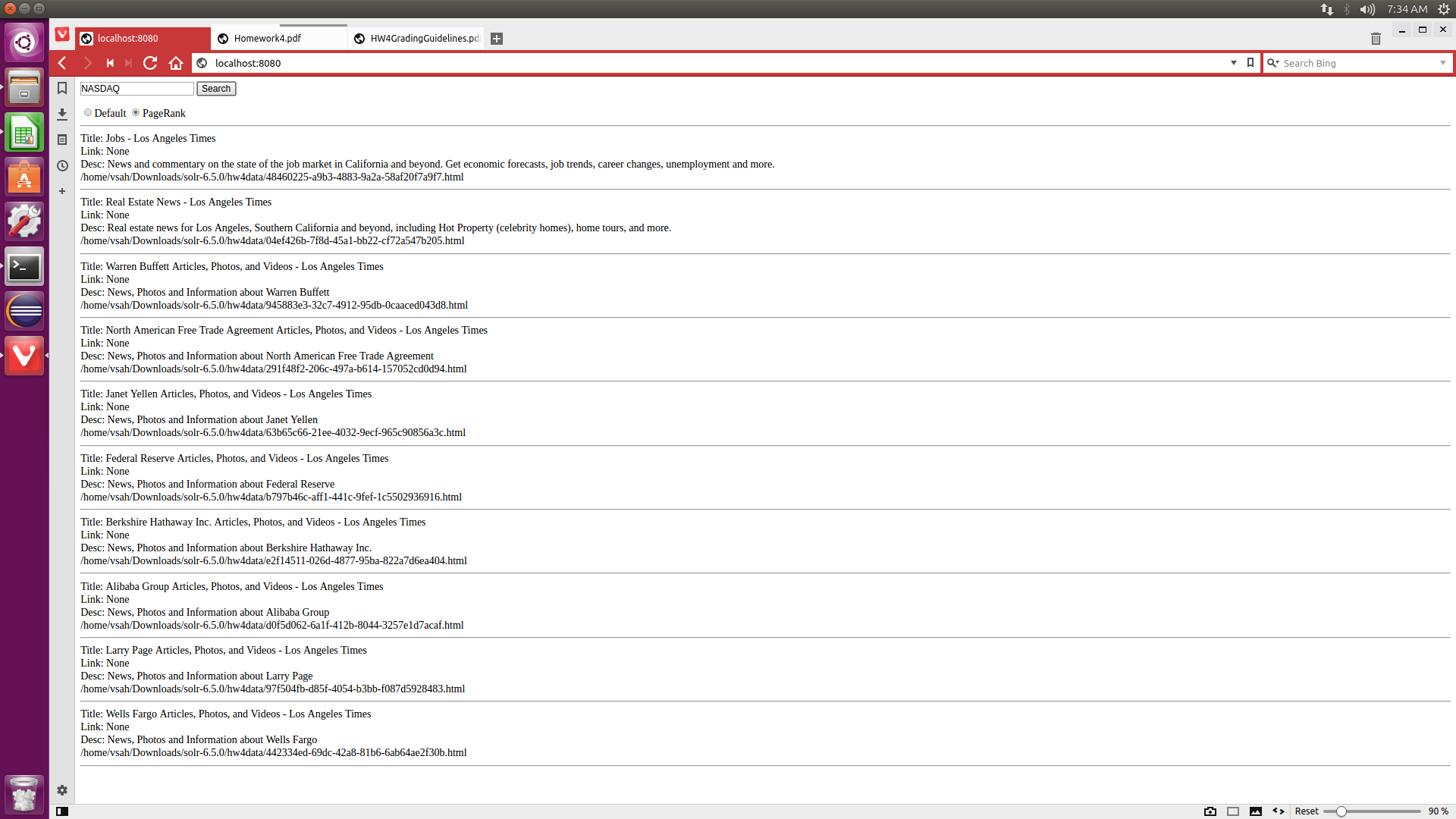


**QUERY RESULTS PAGE RANK**

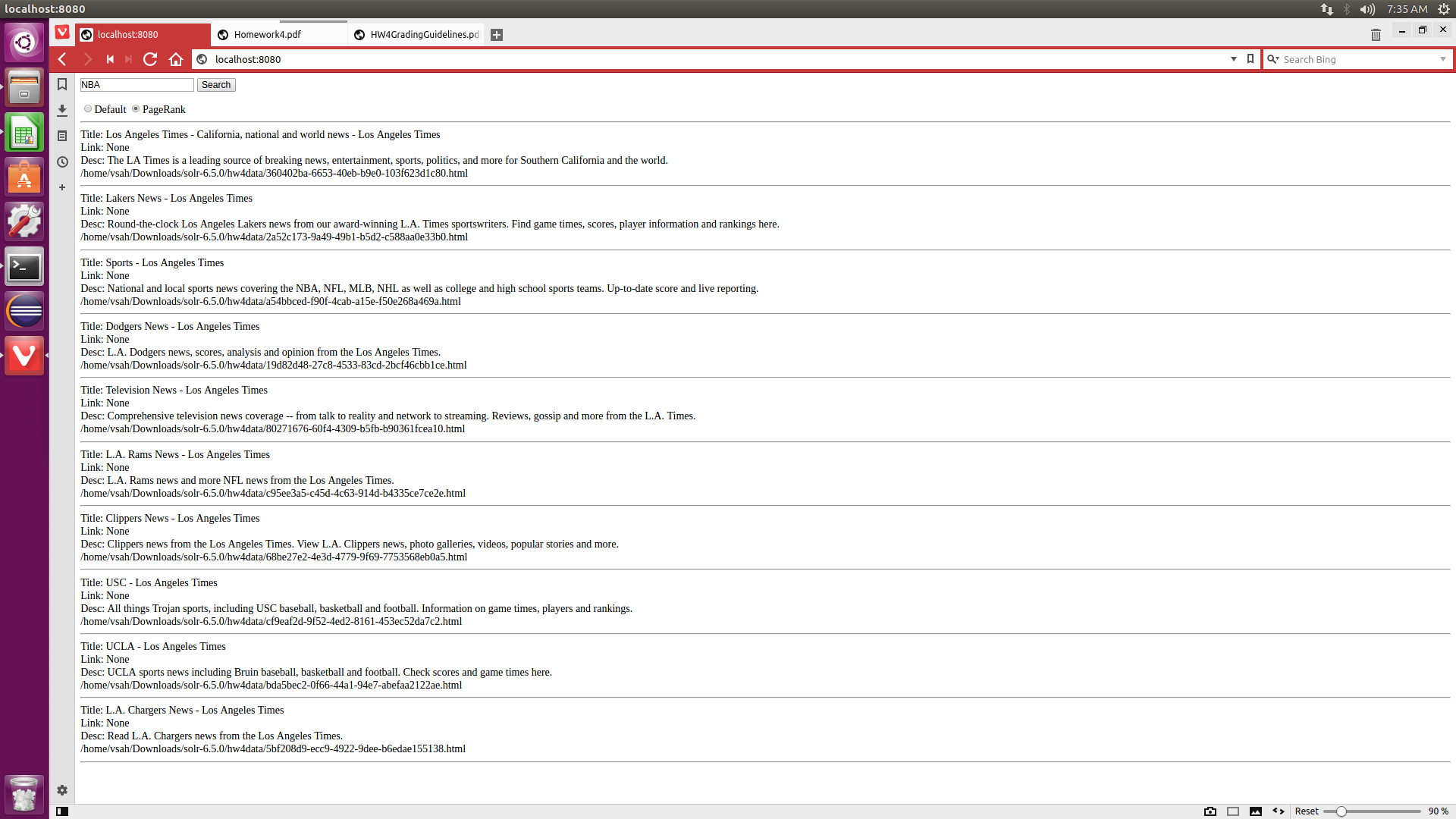
**Brexit**



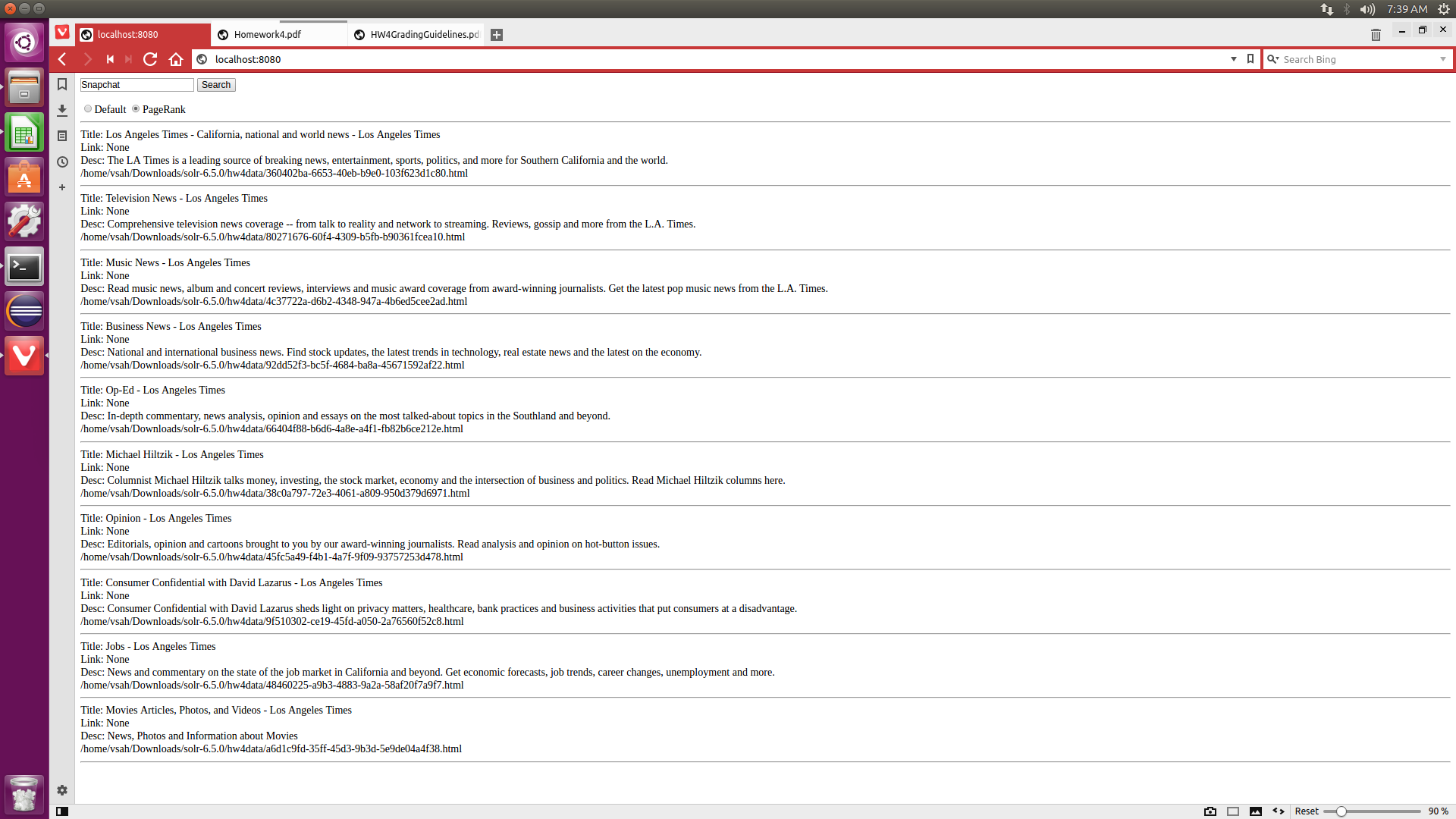
**NASDAQ**



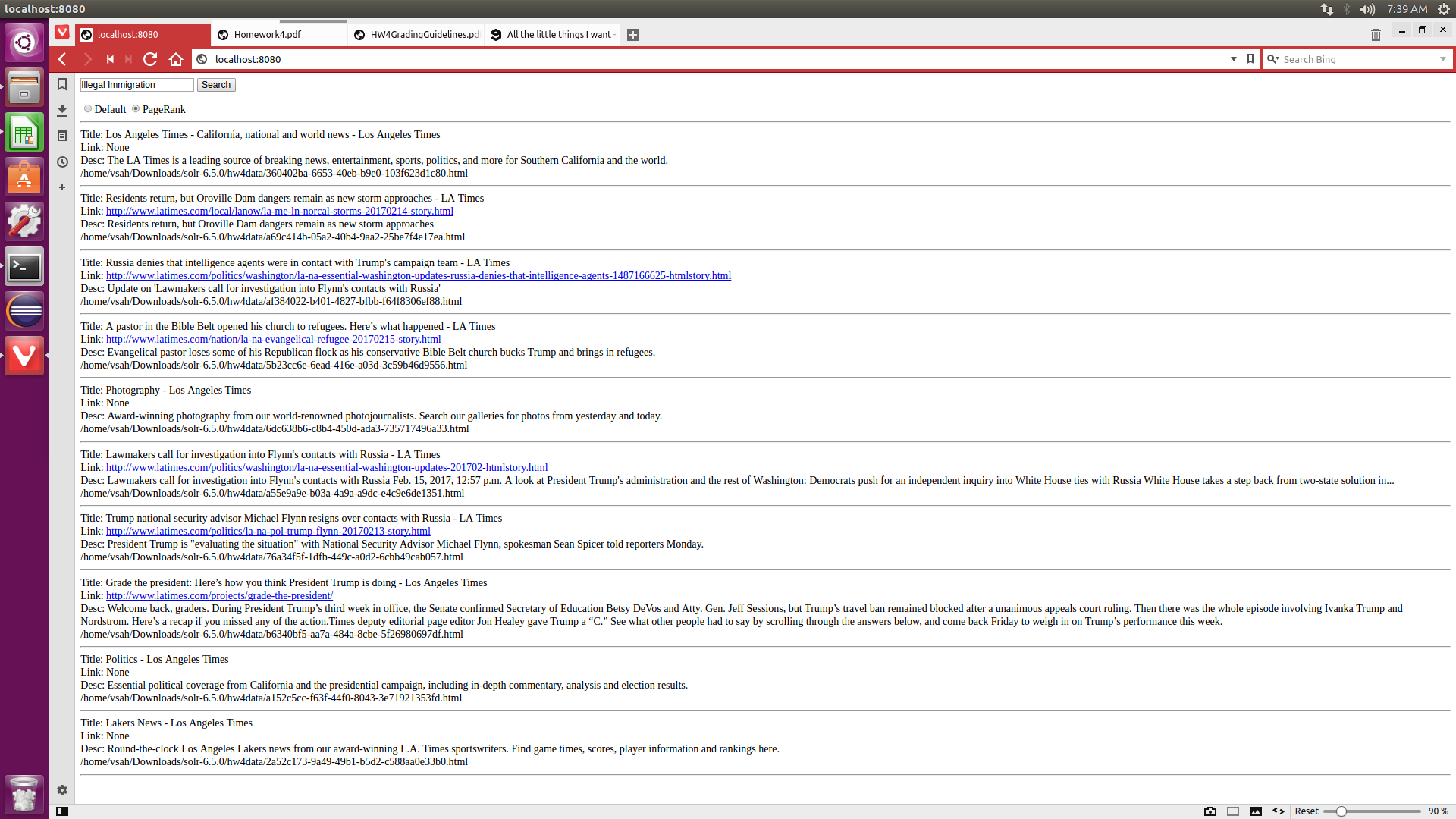
**NBA**



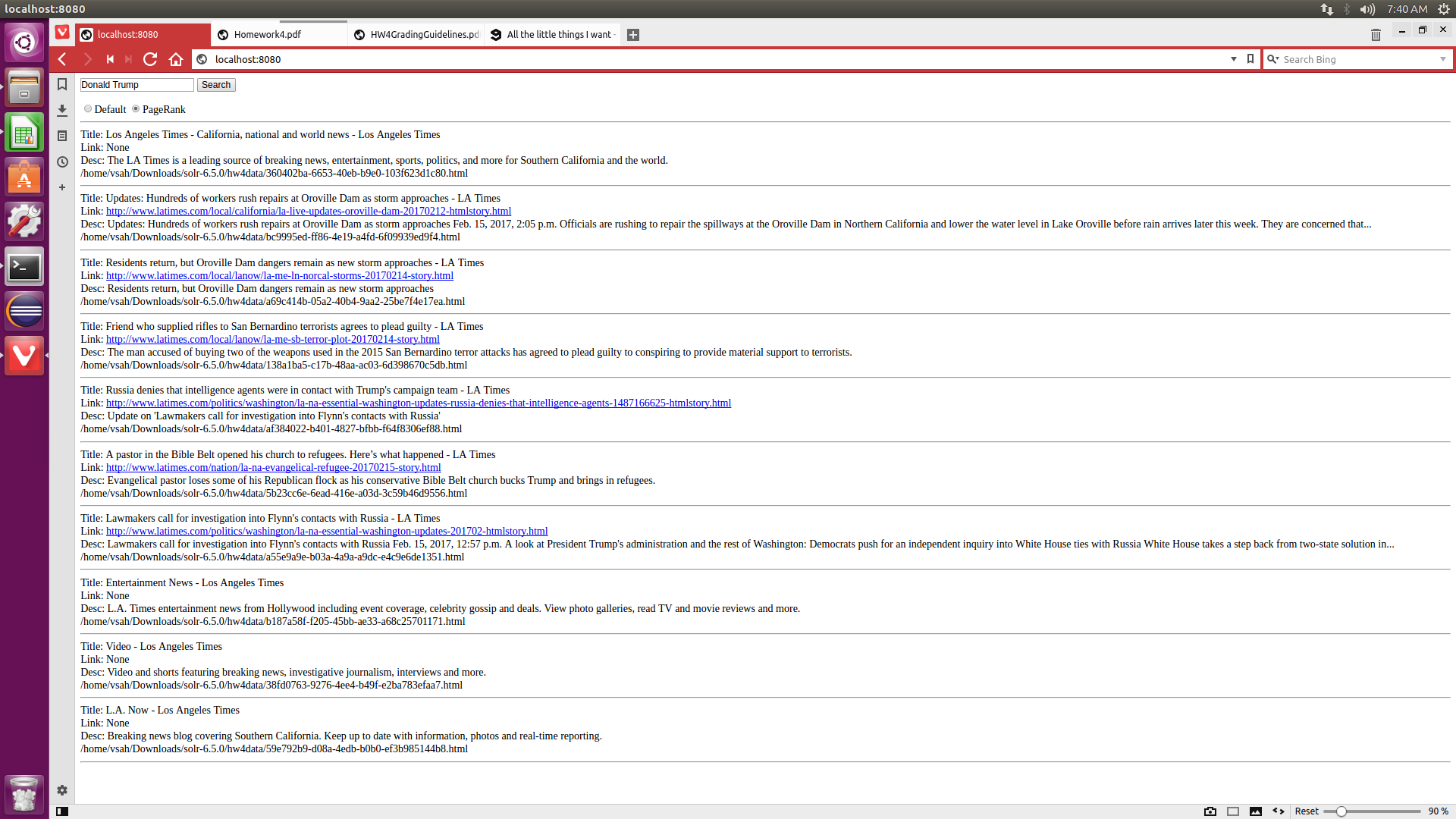
**Snapchat**



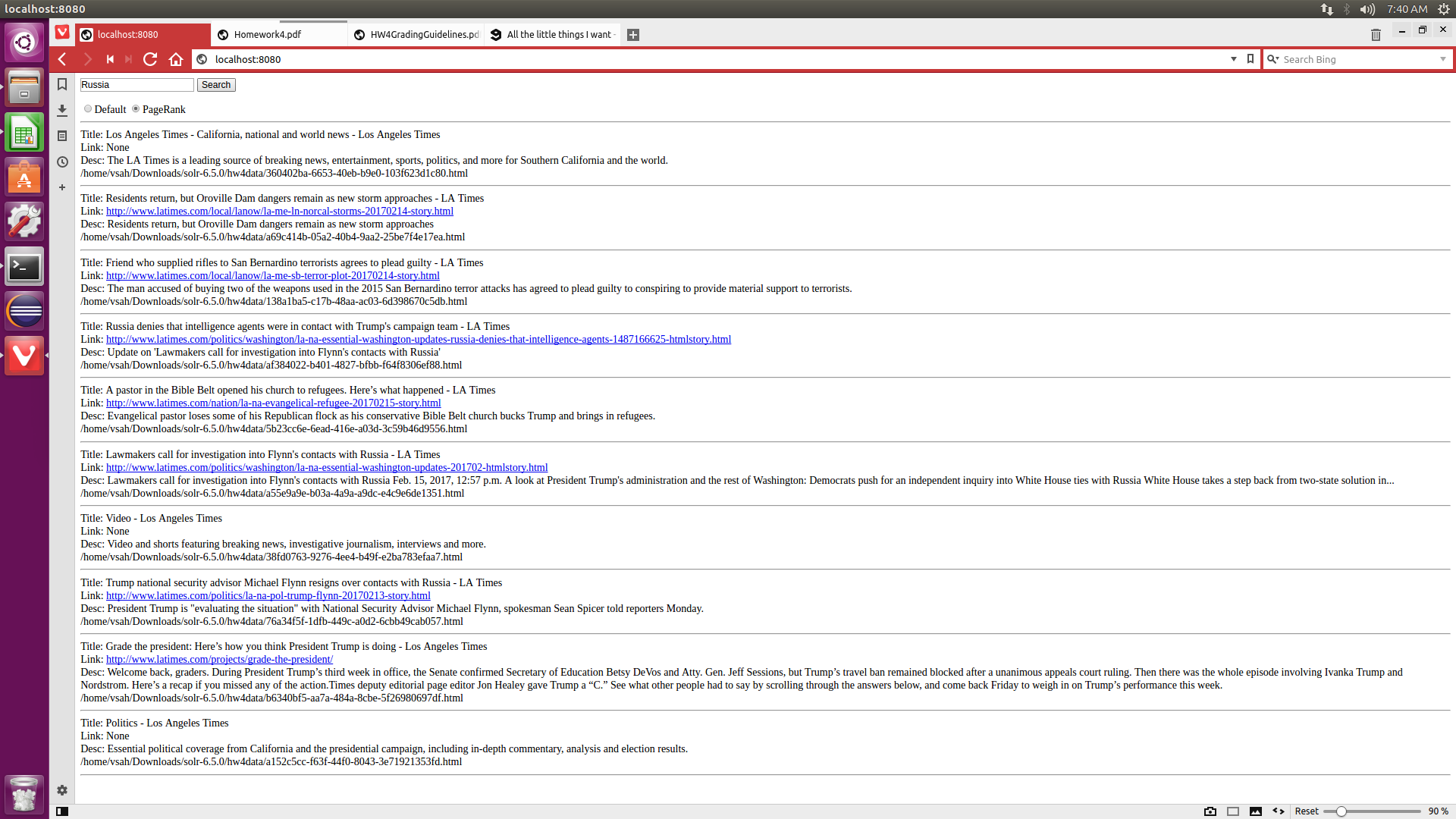
**Illegal Immigration**



**Donald Trump**



**Russia**



**NASA**

