Documentation:

Functionality

The chrome extension does the following based on the page the user is currently on:

(A) Topic pages:

1. Get best questions for topic
2. Get open questions for topic
3. Get related topics for topic
4. Get top stories for topic

(B) User page:

1. Have a look at recent answers (condensed look)

Technical stack:

1. Extension - Its a simple chrome extension with the markup defined in popup.html and the javascript code in popup.js file

2. Server - Django is used as the web framework to handle requests. The app is hosted on heroku server for testing purposes. Python beautifulsoup4 and feedparser packages are the external packages which were used for processing the output.

Features:

1. The extension was built with the purpose of providing the user with more information in a more condensed manner than is available on the quora topic/user pages. For example: Get best questions for a topic page allows a user to quickly browser through the questions. Similarly, a user might just be interested in peeking through the answer activity for a particular user (without reading the answers on a whole).

2. If the url is that of a topic page, the user can retrieve best questions/open questions/related questions/top stories(upto 100, most recent ones first) for the topic. Similarly if the user is on a user profile, the extension allows you to have a cursory look at the questions the user has answered (upto 100 answers, most recent ones first). 3 out of 5 features also have the link to corresponding questions/answers.

3. The chrome extension does not work when the user is not on quora.com domain. The user need not even be logged in quora to use the extension (tested on incognito window).

Implementation:

The implementation was carried out using django as the web framework. When a user clicks on the extension, a cross domain request is made to the server (hosted on heroku) for the desired choice made. The request is performed on the basis of the url the request is sent from. Based on the option selected a GET request is made to the server which then responds back with the data by populating div with id="output" in popup.html page. The values are populated by using rss feeds (using feedparser module)wherever applicable (getting best questions, top stories, user answers - http://www.quora.com/When-will-Quora-add-RSS-support) or by parsing the html pages (using beautifulsoup module).

Dependencies/Requirements:

The server side requires beautifulsoup and feedparser module installed on the system. Also an external chrome extension: Allow-Control-Allow-Origin: *(https://chrome.google.com/webstore/detail/allow-control-allow-origi/nlfbmbojpeacfghkpbjhddihlkkiljbi?hl=en) needs to be installed to allow cross domain request to heroku server (not implicitly allowed by w3.org - http://www.w3.org/TR/cors/). I was unable to fix the setting on heroku server so I had to use the extension for testing purpose. It is more of a fix to enable the core functionality of the extension.

Example test run:

0. The user has to install the chrome extension (https://developer.chrome.com/extensions/getstarted#unpacked) and Allow-Control-Allow-Origin: *(https://chrome.google.com/webstore/detail/allow-control-allow-origi/nlfbmbojpeacfghkpbjhddihlkkiljbi?hl=en).

1. The user has to enable the extension Allow-Control-Allow-Origin.

2. Browse to topic pages/user pages on quora. Click on the buttons in the extension to get appropriate output (sample screenshots attached alongwith for test run).