Preferred Networks Intern Screening 2017 Coding Task for Front-end Development

Change log

- 2017-5-12 : Initial version

Notice

- Server side need to be run on either MacOS or Linux environment.
- Client side should be developed with HTML base, and check the behavior on Google Chrome or Firefox.
- Usage of other services on the Internet is not allowed.
- Feel free to use any library, database and search engines that you want. However you need to specify how to set up the develop environment in the instruction.
- Please use any of the following programming language
- C, C++, Python, Ruby, Go, Java, JavaScript（including AltJS）

Things to submit

- Please submit the code for problems.
- Please include not only the code of server-side and client-side but also configuration files and so on.
- It is not necessary to include third-party libraries in the submit code, if the setup method is specified in the instruction.
- Please **do not** include data to be searched.
- Please include the instruction document which explains how to compile/run your code (such as execution environment, library, and other necessary steps).

How to submit

Submit files should be uploaded to Google drive. After uploading them, please fill in the share URL on the submission form. See the following URL for how to upload files to Google drive:

- Submission form: https://docs.google.com/a/preferred.jp/forms/d/e/1FAIpQLSd_zC_XT2dHM-yRO9WQ-YuRU0sx2HeQIep-NBoqMWpN_j8KNw/viewform
- How to upload files : https://www.preferred-networks.jp/wp-content/uploads/2017/04/intern2017_GoogleUpload_3.pdf

Inquiry

If you have any questions about the problems, please also send these to intern2017@preferred.jp (the same address as used for applying to the internship)

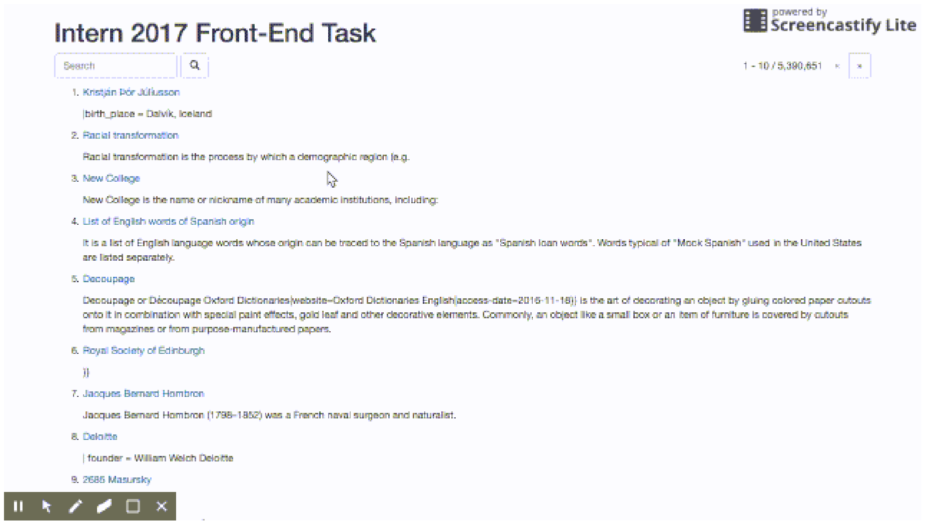
Task description

In this task, you will make **a prototype** of web-service **without** external cloud services. You need to submit source code and environment setup instruction (please refer "Where to submit" section for detail). You will have a demonstration and code reviews in the interview, so please prepare for them.

Problem 1

Please make **a prototype** of database search system with incremental search. We use Wikipedia's dump data enwiki-latest-abstract.xml for the data to be searched.

Example for reference



Intern 2017 Front-End Task

1 - 10 / 5,390,651

1. Kristján Þór Júlíusson

   |birth_place = Dalvík, Iceland

2. Racial transformation

   Racial transformation is the process by which a demographic region (e.g.

3. New College

   New College is the name or nickname of many academic institutions, including:

4. List of English words of Spanish origin

   It is a list of English language words whose origin can be traced to the Spanish language as "Spanish loan words". Words typical of "Mock Spanish" used in the United States are listed separately.

5. Decoupage

   Decoupage or Découpage Oxford Dictionaries|website=Oxford Dictionaries English|access-date=2016-11-18}} is the art of decorating an object by gluing colored paper cutouts onto it in combination with special paint effects, gold leaf and other decorative elements. Commonly, an object like a small box or an item of furniture is covered by cutouts from magazines or from purpose-manufactured papers.

6. Royal Society of Edinburgh

   }}

7. Jacques Bernard Hombron

   Jacques Bernard Hombron (1798–1852) was a French naval surgeon and naturalist.

8. Deloitte

   | founder = William Welch Deloitte

9. 2685 Masursky

Figure 1:

We bundle the picture as task1.gif. It is *not* necessary to make completely same UI with above, if the minimum requirement is satisfied.

Requirement

- The web service suffices to be a prototype, it is OK if it works for demonstration purpose only (it is not necessary to assume heavy access management etc.).
- Search engine function should satisfy following specifications:
- Search scope is both `title` and `abstract` from enwiki-latest-abstract.xml, but please exclude `Wikipedia:` sentence from `title` entry.
- If the search query is lower case, please ignore the difference of capital letter and small letter in the search.
- If the title is completely same with the search query, it is preferable to show it on the top of search results.
- We don't specify the behavior when the search query is empty.
- Search UI should satisfy following specifications:
- Please implement incremental search. Search results should be updated on every stroke of input.
  - UI should be updated with appropriate speed as the input is updated.
- Show the number of search results. It is OK to show round numbers, but in this case please show that the number is rounded explicitly (For example, show "*About* 1,000,000")
- Search results should show `title` and `abstract` for each search result. `title` should be linked to `url`, and open new tab or window when clicked.
- Show 10 items of the search results, and use pagination for the rest.
- For the rest of undetermined behaviors, specifications, or UI, please implement to maximize UX.
- Example: "Previous page" link of pagination in the first page is disabled.

Problem 2

Add a functionality to share the search queries of all clients in real time (we call this "shared search history" later on) to the search service developed in Problem 1.

Example for reference

We bundle the picture as task2.gif. Shared search history is added on the right of search results which is developed on Problem 1.

It is *not* necessary to make completely same UI with above, if the minimum requirement is satisfied.

Requirement

- The web service suffices to be a prototype, it is OK if it works for demonstration purpose only (it is not necessary to assume heavy access management etc.).

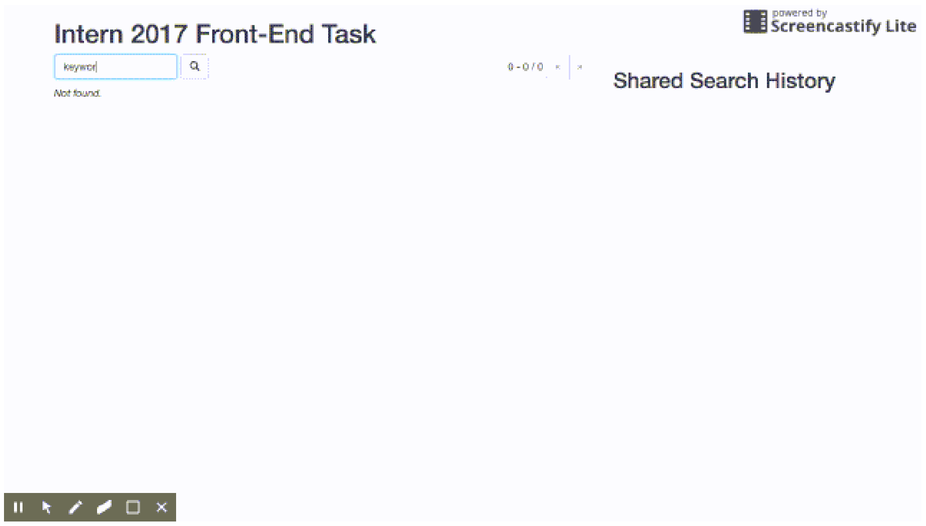


Figure 2:

- Shared search history/UI should satisfy following specifications:
- Other client's search query should be shown in real time (few seconds delay is allowed).
- Search history only shows the latest 10 items at most.
- Same search query should not be shown in duplicated way, even when search is executed with the same query. Please show only the latest search query on shared search history.
- If search query on shared search history is clicked, it should appear in the search form and the search should be executed immediately.
- This web service also implements incremental search, so please consider carefully how not to register a search query to history while user is in the process of inputting a search query.
- Example: search query is registered only a specified number of seconds after the input query.
- For the rest of undetermined behaviors, specifications, or UI, please implement to maximize UX.