SUPINFO Academic Dept.

SUPINFO International University

INSTITUTE OF INFORMATION TECHNOLOGY

Group Projects
A.Sc. 2 – Search Engine

Project presentation

2013 - 2014

Version 1.0
Last update: 20/11/2013
Use: Students
Author: Samuel CUELLA

TABLE OF CONTENTS

1 PROJECT OVERVIEW

Tweek Bros. wants to market a new product: An on-premises search engine that allows companies and individuals to protect their privacy.

The search engine will work on resources made available over http and ftp protocols but is not primarily intended as an Internet search engine, but more as a private/intranet search engine.

You've been selected as a subcontractor to carry the development of this product.

2 FUNCTIONAL EXPRESSION

2.1 SOFTWARE DEVELOPMENT

2.1.1 Front-end

The search engine should be available over http. Users should be able to enter queries from the search engine landing page.

The search engine should present results per page, and provide users with a way to change the number of results per page.

Users should be able to supply existing resources URLs to have the search engine crawls them.

2.1.2 Back-end

The search engine should be able to crawl resources, over http and ftp. The engine crawls "known" URLs by itself periodically to update its tables. It should be able to recursively follow links found in user-supplied URLs and register these links and known URLs.

The search engine maintains a database fed with the result of its periodic crawls. Each resource, identified by its URL is associated with a "significant words" field that is filled with words that describe the resource. This field can be generated using the "meta keywords" for HTML resources, h1 titles, bold words, etc. It's up to you to come up with the best algorithm to generate and maintain this field content.

For text resources, the search engine also keeps the crawled content directly in the database. Binary (archives, images, ...) data is not stored/cached into the database.

For each search query, the search engine first searches the “significant words” to get matching resources. It also then performs a full text search on all text resources. The search result is the union of these two sets of results. Results from the first set have a higher “score” and should be displayed first in order.

Backend must be written in C++. You can made it available over HTTP using any webserver through its (Fast)CGI interface, for example, or even write your own HTTPd, or patch an existing, etc. at your option.

2.2 SUPPORTING ARCHITECTURE

The volume stored in the database by the search engine and the queries submitted will be fairly big. You need to setup a MySQL (or any other relational database you find fit such as: Oracle, MS-SQL, PostgreSQL, etc) cluster to store the data and respond to the queries. Be sure to have load-balancing in your solution, and at least four nodes.

Warning: NoSQL solutions are strictly forbidden.

The search engine webserver should also not be a single machine: Use at least four nodes to handle search queries, using DNS-based Apache (or any other server) clustering. Moreover, these “workers” should not be directly queried by end clients. Use reverse proxies to query the search engine web servers: Clients connect to reverse proxies which impersonate the “real” servers which in turn will query and cache from the servers.

3 DELIVERABLES

Students should include the following elements in their final delivery:

- A zip archive with the project source code.
- All configuration files for supporting architecture, along with setup how-to.
- Project documentation, based on the template.
    - Technical documentation explaining your choices and/or implementation choices/details on the following items (at least):
        - Search algorithms (How did you extract significant keywords)
        - Language/Framework, if any
        - Database optimization, if any
    - Deployment instructions

The first document is an academic document. Address the reader as a teacher, not a client. The last one (game manual) should address the reader as a user. These documents can be in French or in English, at your option.

4 GRADED ITEMS

The project will be graded as follows, on a 120/100 scale:

Software development (60 points)

Front end (20 points)

- It's possible to perform a search (5 points)
- Results are displayed page per page (5 points)
- It's possible to choose the number of results per page (5 points)
- It's possible to submit an URL to be crawled (5 points)

Back end (40 points)

- Maintains a list of "known" resources (1 points)
- Crawls periodically its known resources (3 points)
- Can crawl over HTTP and FTP (3 points)
- Can follow links to other resources (5 points)
- Can compute a set of "significant words" for each resource (12 points)
- Can store text resources (1 points)
- Can perform a search on significant words an full text (5 points)
- Returns a sorted by score union of these two searches (5 points)
- Algorithm coefficient against back-end part: 0.8 - 1.3

Supporting architecture (50 points)

- There is a database cluster (25 points)
- There is a DNS-based webserver cluster (20 points)
- Clients goes through reverse proxies to query the search engine (5 points)

Bonus points (10 points)

- Additional features done by the students (10 points max)