CHAPTER I

INTRODUCTION

1.1 Local Search

Local search is any search made with the goal of finding something within a specific geographic location. Local search is searching for something (what) in some place (where). Essentially, anything that you would traditionally look for in the print Yellow Pages becomes a Local Search when it is conducted online [11]. Here, “conducted online” refers to real time search over local search engine i.e not seeking information in pages that are printed some time ago. Typical local search queries include not only information about "what" the site visitor is searching for (such as keywords, a business category, or the name of a consumer product) but also "where" information, such as a location name, street address, city name, postal code, or geographic coordinates like latitude and longitude [15].

Examples of local search:

- Search for the available restaurants in Jawalakhel
- Search for the colleges in Lalitpur district
- Search for the location of Tribhuvan University

Local Search can be categorized as :

- **Nearcasting** - when a searcher is looking for something close to his or her current location.
- **Farcasting** - when a searcher is looking for something elsewhere, often in a place where he or she hopes to be in the future [11].

1.2 Local Search Engine

Local search engine is a tool for local search over the World Wide Web(WWW). Local Search Engine is specialized Internet search engine that allow users to submit geographically constrained searches against a structured database of local business listings [12]. Local Search Engine enables users to search for infrastructures such as hospital, banks, schools, parks etc over certain geographic region i.e within a particular domain.

1. 3 Background

Local search is the natural evolution of traditional off-line advertising, typically distributed by newspaper publishers and TV and radio broadcasters, to the Web. Historically, consumers relied on local newspapers and local TV and radio stations to find local product and services. With the advent of the Web, consumers are increasingly using search engines to find these local products and services online. In recent years, the number of local searches online has grown rapidly while off-line information searches, such as print Yellow Page lookups, have declined. As a natural consequence of this shift in consumer behavior, local product and service providers are slowly shifting their advertising investments from traditional off-line media to local search engines [12].

1. 3. 1 Local search engine Vs Traditional Medias

Traditional medias impose following problems for local search:

- Difficult to know what publication's which date newspaper article contains the required information(what about an article on month ago newspaper)
- With television and radios we need to know when will program with the required information be broad casted and wait till the actual broadcast
- The article(program) may not contain the direct actual data they required
- People's information domain will be smaller than that of search engine.

With the advent of the Web, consumers are increasingly using World Wide Web(WWW) to find these local products and services online because of

- Ease of access (submit query and get result i.e. user does not need to search over pages, listen others, wait for broadcast)
- Can be accessed through anywhere anytime if Internet connection is available
- Can directly query required information the search engine will do the required filtering, so quicker access to information
- Larger scope of available information

1. 3. 2 Yellow Page Vs Local Search Engine

Yellow Page, historically as an offline media in the form of paper, is used as telephone directory. Nowadays, web pages with information as in Yellow Page, referred as Electronic Yellow Pages(EYP) are widely used over World Wide Web(WWW).

The search engine for acquiring local information offers following advantages over yellow page:

i) Ease of Use

People do what is easy for him/her. Access to the Internet via work, home or any library has truly made getting online easy. Over 64% of people who are looking for information online use search engines [13]. If user can type problem or what he or she want into a search engine and find what he or she needs, he or she doesn't need to go through the 100 pages of advertisement in the yellow pages. Search engines are getting more precise down to an industry specialty and geographic focus, while also continuing to be easier to use [13].

ii) Differentiation

Anybody does not want to be presented within a detailed list of competitors, including their addresses and phone numbers. But that is exactly what the Yellow Pages do. Using search engines, you can differentiate yourself from everyone else. So when the ideal client is looking for what you do, they find you, not all your competitors.

iii) Cost Model

In reviewing yellow page and local search engine we need to compare the traits and cost basis of each of them.

	Yellow Pages	Local Search Engine
Printing Cost	Yes	No
Mailing/Distribution Costs	Yes	No
Commitment level	1 year	None
Flexibility of Add	once/yr	anytime

Table 1: comparison of yellow page and local search engine [13]

Clearly, the Yellow Pages model is more expensive while also being significantly less flexible and scalable than online search engines.

1. 4 Motivation

Local search creates new challenges in information retrieval. The amount of information on the web is growing rapidly, as well as the number of inexperienced users in the art of web search seeking the information about local infrastructure over the web. Existing Local search engines and yellow pages with search capability relied on string matching and user submitted keyword based searching. They usually return too low relevant results or too many low quality matches. The situation become even worse when people do not enter the correct spelling for words and unfortunately this is the case for most of the query for local search because people usually do not know standard spelling for colleges, hotels, organization, locations etc.

The following facts depict the need for local search engine in Nepal.

- Lack of reliable and effective local search engine
- Rapidly growing Internet users seeking the seeking information about business or services over the web
- Could be useful tool for development of Tourism industry (could serve as a guide for tourist)

1.5 Problem Definition

We want a user friendly, robust web application for local search based on geographical division of Nepal. In present context, there is lack of good local search service over the World Wide Web (WWW) in Nepal. We want to provide local search service in the form of local search engine with features such as spelling correction , query expansion, query suggestion, map services (such as location identification, direction service).

1. 6 Objectives

Our objective is to develop local search engine which

- provides quality search results
- implements efficient and effective data structures and algorithms improving the search engine's performance
- provides rich and easy to use web interface for local search
- has features such as spelling correction, query-expansion, query suggestion, map services.

1. 7 Scope and Limitation

Existing local search engine and EYPs search results are poor. They also do not provide features such as spelling correction, query-expansion, query suggestion, map services(direction service). They also do not have feature for ranking search results based on distance from user provided origin location.

Whenever there is a need for finding information about businesses or services over certain geographical region, local search engine can be used. We have developed local search engine based on the geographical division of Nepal, so people can use it as a local search engine for Nepal which provides quality search result including features such as spelling correction, query-expansion, query suggestion, map services(direction service).

CHAPTER II

REQUIREMENT ANALYSIS

2. 1 Literature review(Study of existing system)

30 – 40 percent of searches made over web are local search [11]. Local search is increasing with the increase of access to Internet from mobile devices. Now a days, search for businesses or services over certain geographic region can be done on web through Electronic Yellow Pages(EYP) or local search engine. In present context, available EYPs'[1] and local search engine[2] in Nepal does not provide good search results. They also lacks features such as spelling correction, query-expansion, query suggestion, map services.

Based on observation of search results provided existing EYP's and localsearch engines, it seems that most of them implemented prefix based string matching for search. Such prefix matching based search have lower recall because of spelling errors. Recall could be improved by performing string matching that allows errors. The problem in its more general form is to find a text where a text given pattern occurs, allowing a limited number of errors in the matches. Each application uses different error model which defines how different two strings are[8] . One of the best studied case of this error model is edit distance which allows us to insert, delete and substitute in both strings . Edit distance measures distance as the (weighted or unweighted) number of operations required to transform a string into another [14]. The Levenshtein (or edit) metric is a standard tool to estimate the dista[1]nce between two sequences. The edit distance has received a lot of attention because its generalized version is powerful enough for a wide range of applications. An extension of the edit distance enriches it with transpositions (i.e. a sub-stitution of the form ab → ba at cost 1). Transpositions are very important in text searching applications because there are typical typing errors, but few algorithms

1 www.ypnepal.com
2 www. aksbaje.com

exist to handle them. However, many algorithms for edit distance can be easily extended to include transpositions [8]. Another popular noise model is Longest common subsequence. It measures the distance of longest pairing of characters that can be made between both strings. An obvious measure for the closeness of two strings is to find the maximum number of identical symbols in them (preserving the symbol order) [9].

Misspelling is a common phenomena in search engine query. According to Cucerzan and Brill[7], more than 10% of query are misspelled. The situation become even worse for local search query since it typically involves more spelling error than other queries because most of the words in local search query are usually non standard dictionary words. The name of location, name of organizations and various services are non standard dictionary words and we usually do not know correct spelling for such words. So, spelling correction is very important feature for local search engine. The problem of correcting misspelled words in written text is rather old, perhaps the oldest potential application for approximate string matching [8]. To assist users in expressing their information needs, it is important for search engines to automatically generate corrections. Two such mechanisms are in common use. The first corrects a query after it is submitted to the search engine. For confident corrections, the search engine can search the corrected query directly. As the entire query string is given, we refer to such an approach as offline spelling correction. The second technique provides corrections to the query completion suggestions as the query is being entered. Specifically, the search engine responds to each keystroke with a list of query suggestions that best correct and complete the partial query [4].

Use of query expansion generally increases recall [16]. Query expansion is the process of supplementing additional terms or phrases to the original query to improve the retrieval performance. The central problem of query expansion is the selection of the expansion terms based on which user's original query is expanded. Thesaurus helps to solve this problem. Thesaurus have frequently been incorporated in information retrieval system for identifying the synonymous expressions and linguistic entities that are semantically similar. Thesaurus has been widely used in many applications, including information retrieval and natural language processing [1]. Several studies have reported the

construction and use of different types of thesauri as aids to the query-expansion process [2]. In general, thesauri within information-retrieval systems can be categorized as belonging to one of three main types: standard manually constructed thesauri, searching thesauri, and automatically constructed thesauri. Standard thesauri with equivalence, associative relationships have been widely used for search term selection and query-expansion [2]. For local search we need to relate functionally similar businesses or services not the tokens that are related by meanings. So thesaurus with association or equivalence from functional aspect is required.

2.2 Data Collection Methods and Sources

Both primary data and secondary data are used in the project.

2. 2. 1 Primary Data Collection:

Primary data were collected from ward - 11 of Kathmandu Metropolitan City and ward - 10 of Lalitpur Submetropolitan City. Data such as organization name, contacts, spatial information such longitude and latitude, geographical division based location etc were collected.

2.2.2 Secondary Data Collection:

The secondary data were collected from various website like edusanjal.com, tourismkathmandu.com, maps.google.com, ypnepal.com.

2. 3 Software Requirement Specification

2. 3. 1 Functional Requirement

The system should enable users to search businesses or services over certain geographic region. The system should allow user to make search based on geographical division of Nepal.

2. 3. 2 Non Functional Requirement

It includes features such as

2. 3. 2. 1 Spelling correction

If user supplies incorrect spelling for query, the system corrects the spelling for words in query .

For example:

hotle → hotel

2. 3. 2. 2 Query-expansion

When user search for certain businesses or services, it also list other businesses or services which provides similar functions.

For example:

restaurants → restaurants + cafe

2. 3. 2. 3 Query-suggestion

System suggests possible query for partial typed query by users so far.

For example:

ca	→	cafe , campus
(partially typed query)		(suggested queries for "ca")

2. 3. 2. 4 Map services

Two types of map services are provided:

i) locate service in which location is pointed over map is given.

ii) direction service in which direction such as walking direction and driving direction is given to user from his area of interest of origin to business location which he/she searched.

2. 4 Feasibility Analysis

After making an initial investigation, feasibility study is carried out to check the work ability of the system. Feasibility study is testing the system proposed according to its work on ability to meet the user needs and effectiveness use of the resources.

2. 4. 1 Economical feasibility

Since businesses houses or service providers also wants them to be listed in search result we mainly need to collect in depth data of some highly crowded residential areas. Once

the search engine becomes popular over such regions, the business houses or service providers themselves starts to provide their data requesting them to be listed in search engine. Further, the technologies used such as Java, MySQL are open source.

2. 4. 2 Technical feasibility

While discussing about the technical feasibility we need to answer some questions like whether the technology needed for the system exists or not? How difficult it will be to build? And whether the people or the organization has knowledge for using the system? The hardware required for the project are computers, GPS(Global Positioning System) device which are easily available in the market. For database management we required MySQL which is free and open source and was quite easy to use for preparing the project. The system itself can be implemented with JSP(Java Server Pages)+ servlet + POJO (Plain Old Java Object) on java enterprise platform at server side and using HTML(Hypertext Markup Language)+ CSS(Cascading Style Sheet) + javascript at client side. For map service , google maps api can be used. Thus the project is technically feasible.

2. 4. 3 Operational feasibility

User inteface is designed to be user friendly. Any user with simple knowledge on using Internet can use it. Search results are also displayed in a way that can be easily interpreted. Furthermore, our local search engine provides quality search results than rest of the existing EYPs' and local search engine in Nepal, so project is operationally feasible. Once the system has been installed it is easy to operate and Users can easily search to acquire information and it doesn't require any special training . It does not require special knowledge and experience to use it.

2. 5 System Planning

2. 5. 1 Software Development Model or Methods

Evolutionary prototyping was used as software development model. Quick working version was developed first and it was then refined to add additional functionalities. The prototype evolved by

- Adding rich user interfaces
- Optimizing data structures and algorithms to tune performance
- Adding additional functionality not covered in prototype such as map services , relevance feedback, spelling corrections of query, query-expansion, query-suggestion etc.

2. 5. 2 Timeline

The project was completed with activities duration as shown below:

ID	Task Name	Start	Finish	Duration
1	Study and Analysis	2012-05-15	2012-05-29	11d
2	Design	2012-05-24	2012-06-08	12d
3	Data collection	2012-06-05	2012-06-19	11d
4	implementation	2012-06-11	2012-07-19	29d
5	Evaluation and Testing	2012-06-25	2012-07-23	21d
6	Documentation	2012-07-16	2012-08-03	15d
7	Review	2012-08-03	2012-08-09	5d

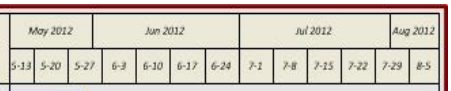


Fig 1: Timeline

Chapter III

DESIGN

3. 1 System Architecture

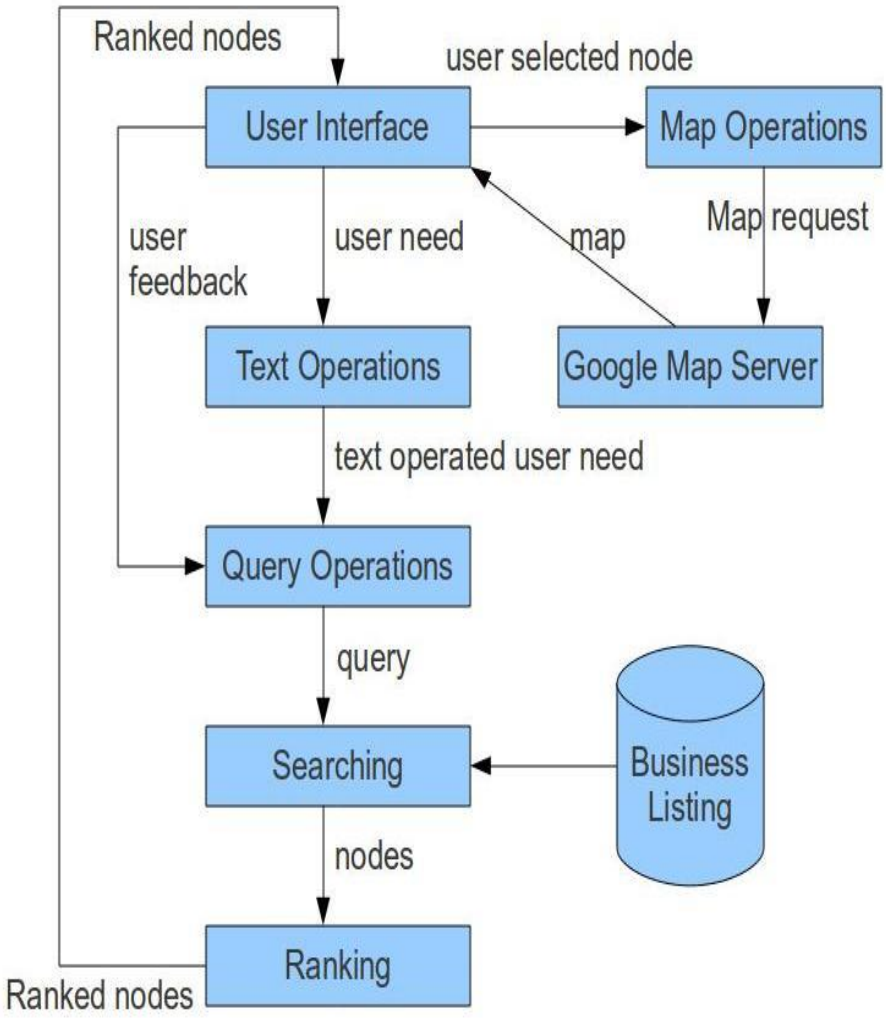


Fig 2: System architecture

3. 1. 1 User Interface

User interface manages interaction with the user. User interface is responsible for

- input of query from user
- visualization of result

3. 1. 2 Text Operation

Text operation involves operations that operates over the text of the user query. It involves operations such as:

3. 1. 2. 1 Lexical analysis of the query

The user query is simply submitted to system in the form of sequence of characters in each field of query interface which should be tokenized before any other operation is performed. Lexical analysis (or tokenization) involves breaking of stream of characters into meaningful words or element called as tokens.

Example of tokenization:

Kathford International College → |Kathford | International | College |

Tribhuvan University → | Tribhuvan | University |

(Tokens)

3. 1. 2. 2 Spelling correction

Sometimes user may supply misspelled words in query resulting the degradation of search result quality. Correction of spelling increases the search results quality.

Examples:-

Schol → School

Universtiy → University

3. 1. 3 Query operations

Query operation involves transforming the query such that it improves retrieval. It involves

3. 1. 3. 1 Query expansion

Query expansion involves adding semantically related tokens to the user query. If

user actually searched the “lodge” and if the search result also includes guest houses, user would be happy. Thus we should add semantically related tokens to user query for improving recall of search which is referred as query expansion. Example of query expansion:-

college → college + campus

lodge → lodge + guest houses

3. 1. 3. 2 Query transformation using relevance feedback

Sometimes the user themselves exactly don't know what they are searching for. Consider user seeking information about nearest hospital from his location. In that case what user can do is formulate query for class search for hospital in his location. Then obtained result can be distance sorted and the result node on top is what user is actually searching for. When user click on the link on that node, the system automatically formulates query for that entity search (i.e query for search for that hospital)

Another approach for query transformation is to suggest the list of possible query for the partial query supplied by user so far so that user can quickly better formulate their query.

Suppose user types “ca” in the “What” field of query interface, as soon as user types in “ca”, application suggests the list of possible query such as

“cafe”

“campus”

to user so that user can simply select their query from those suggestion helping user to better formulate their query easily and quickly.

3. 1. 3. 3 Identification of scope of query

Identification of query scope is query operation that does not actually manipulate the query. Instead what it does is, it characterizes the query. It identifies the scope of query.

There could be two scope of query:

3. 1. 3. 3. 1 Entity scope query

In this scope of query user is interested in search of information about particular entity.

Example: search for information about Kathford International College

3. 1. 3. 3. 2 Class scope query

In this type of search, user is interested in search of list of entities of particular type.

Example: search for list of colleges in Balkumari, Lalitpur district

We need to identify scope of query before actual search to ease searching and make search engine scalable.

3. 1. 4 Searching

After query preprocesing, searching performs the actual search for user need against the database of business listing. It retrieves nodes relevant to user query. The search is based on pattern matching of user query against available business listing. Since there are two scope of query, there will be two kind of search:

3. 1. 4. 1 Entity search

When the scope of query is for particular entity, the search is performed based on pattern matching of name fields of records in business listing.

Example of entity search:

search for information about particular hospital eg. Patan hospital

3. 1. 4. 2 Class search

When the scope of query is for class of certain type of entity, the search is performed based on pattern matching over category of businesses listing records.

Example of class search:

search for list of available hospital in certain region

3. 1. 5 Ranking

It scores all retrieved nodes according to relevance such that most relevant node will be displayed at top to the user.

3. 1. 5. 1 Entity Search Ranking

For entity search, in most of cases we do not need to rank since result consist of only one node. But in cases where two or more businesses or services have similar name, the result will include more than one result and in that case ranking is done on basis of degree of text similarity between query and name of entity.

3. 1 . 5. 2 Class Search Ranking

For class search, there are three rank metric.

3. 1. 5. 2. 1 Alphabetical order

This metric ranks the search results on ascending order of alphabet from their name.

3. 1. 5. 2. 2 Distance

Based on distance as rank metric, search results node are sorted such that node with minimum distance from user submitted location will be at top of results. For distance sorting, the distance is between locations are obtained from Google Map's Distance Matrix API. But, obtaining distance matrix from Google map server is not always a success because of networking problem and sometimes google maps could not determine distance between given points. In that case, the nodes are ranked on basis of displacement distance obtained from spherical law of cosine given by

$d = \text{acos}(\sin(lat1).\sin(lat2)+\cos(lat1).\cos(lat2).\cos(long2-long1)).R$ [15]

where R is radius of earth taken as 6371km.

(lat1,long1) and (lat2,long2) are two locations whose distance is being calculated.

3. 1. 5. 2. 3 User Rating

Using this rank metric, node with highest user rating will be at top.

3. 1. 6 Map Operations

Once the relevant nodes are presented to user, user may be interested to see the location of infrastructure on the map or get the direction of target location which user searched from user submitted origin location. The user requests for map option is intercepted by map operation module and makes necessary request to Google map server for map and response map from Google server is finally presented to user.

3. 2 Diagram

3. 2. 1 Use Case Diagram

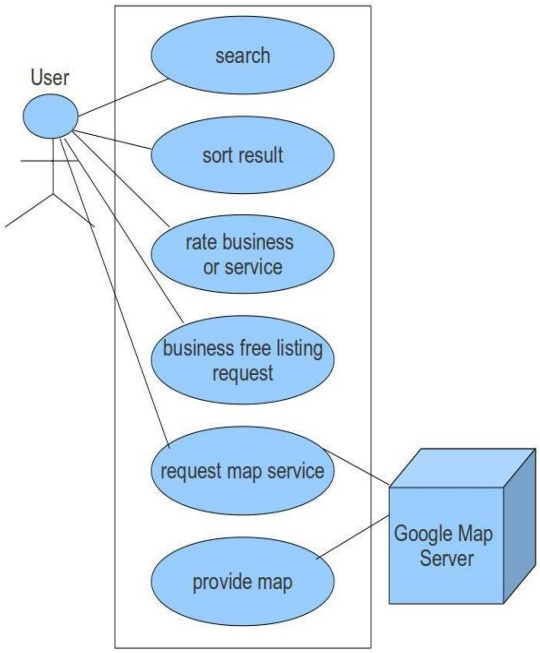


Fig 3: Use case diagram

3. 2. 2 Sequence Diagram

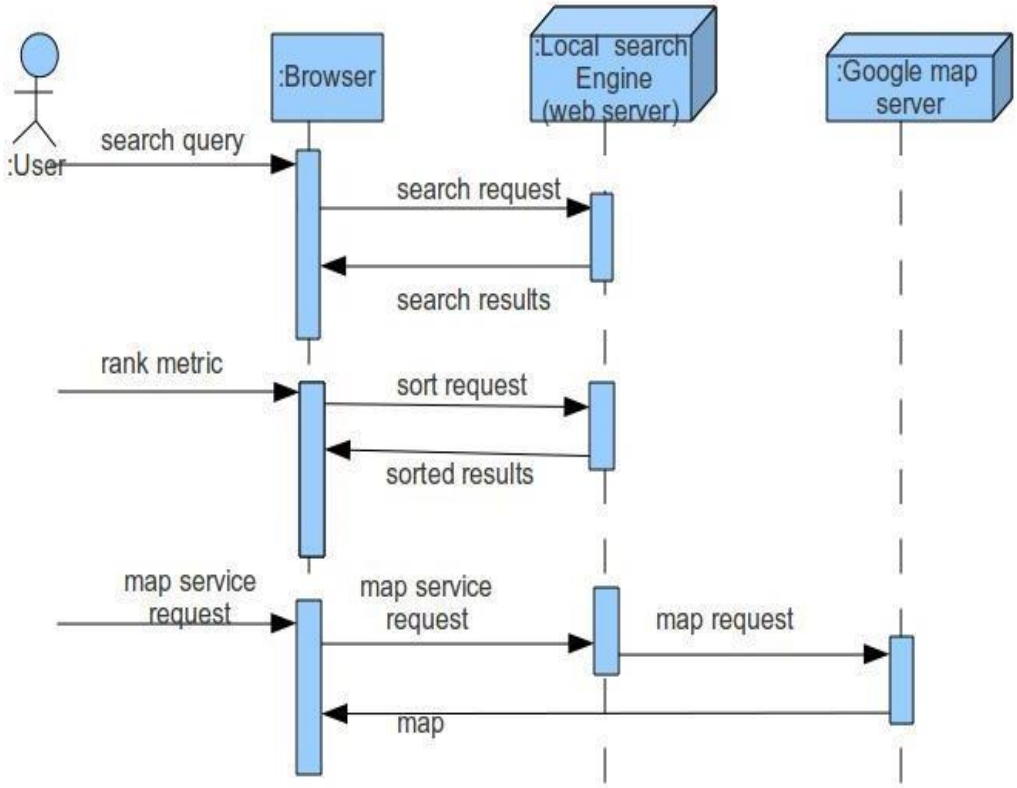


Fig 4: Sequence diagram

3. 2. 3 Class diagrams

3. 2. 3. 1 Packages used

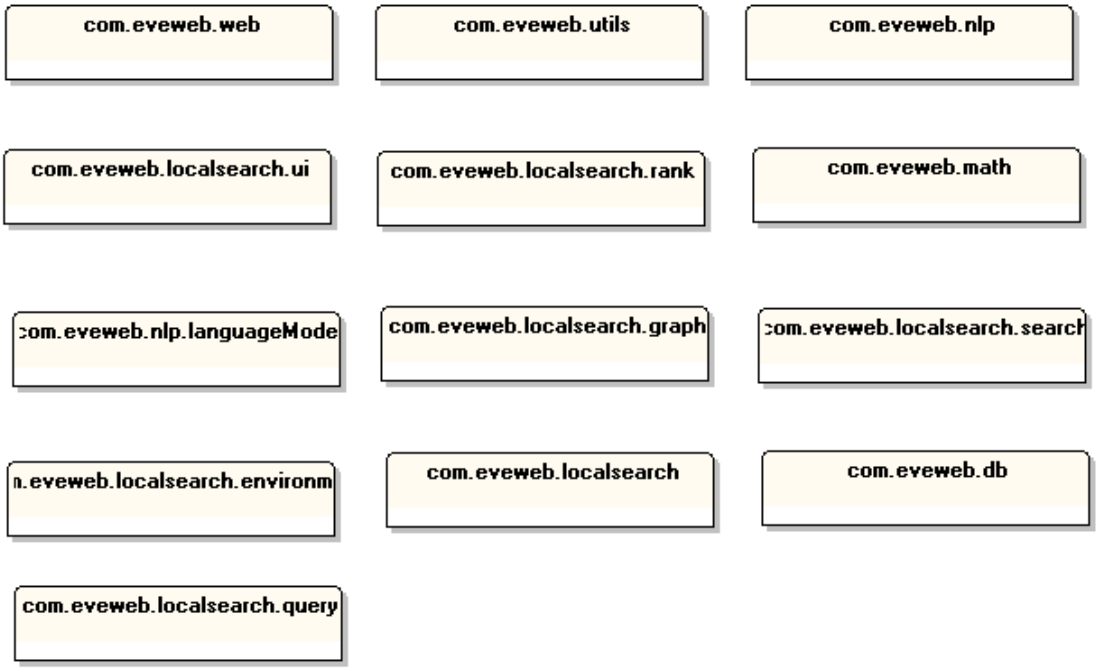


Fig 5: Packages used

3. 2. 3. 2 Class diagram for package com.eveweb.localsearch.ui

Legend
+ : public
- : private
: protected

::com.eveweb.localsearch.ui

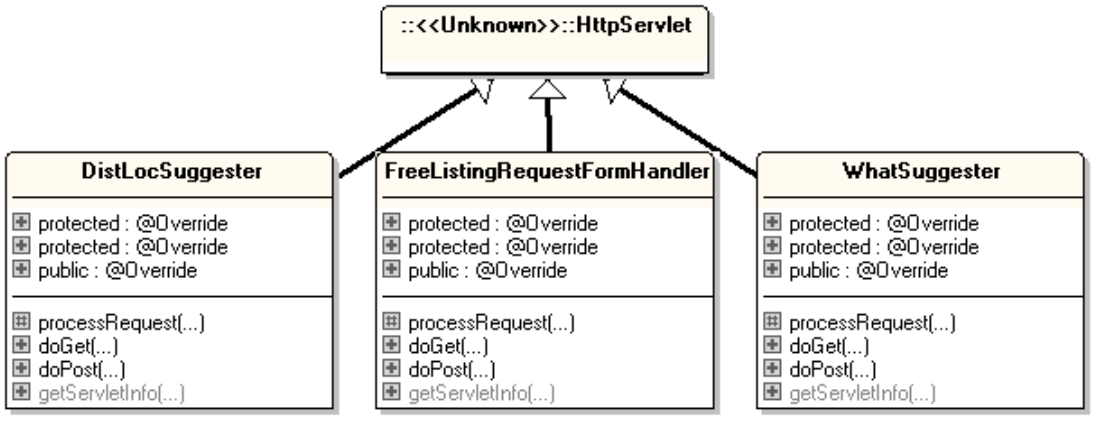
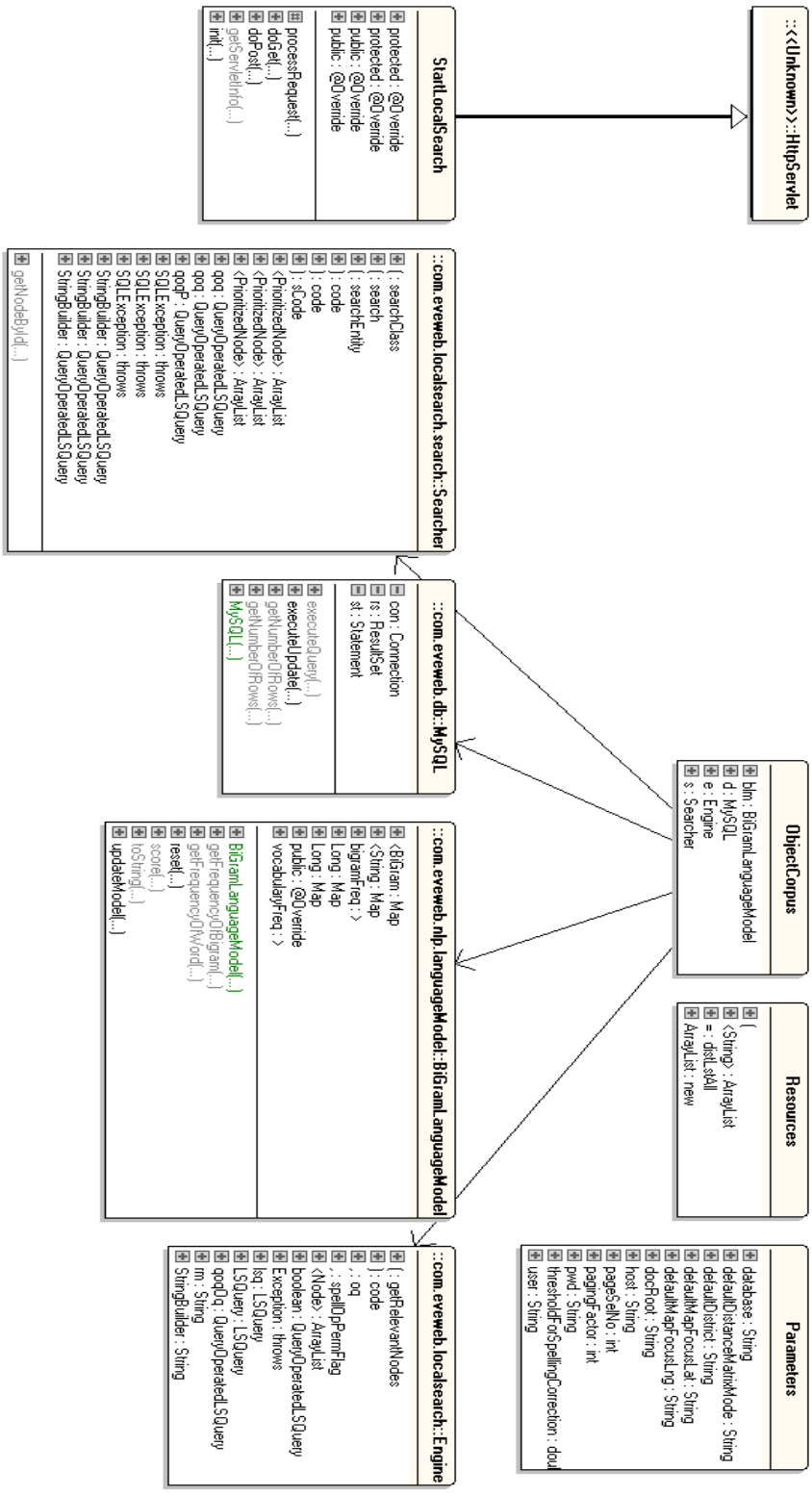


Fig 6: Class diagram for package com.eveweb.localsearch.ui

3. 2. 3. 3 Class diagram for package com.eveweb.localsearch.environment



Note: class diagram for other packages are kept in Annexure (at page 50, 51, 52)

3. 2. 4 ER (Entity Relationship) Diagram

Fig 7: Class diagram for package com.eveweb.localsearch.environment

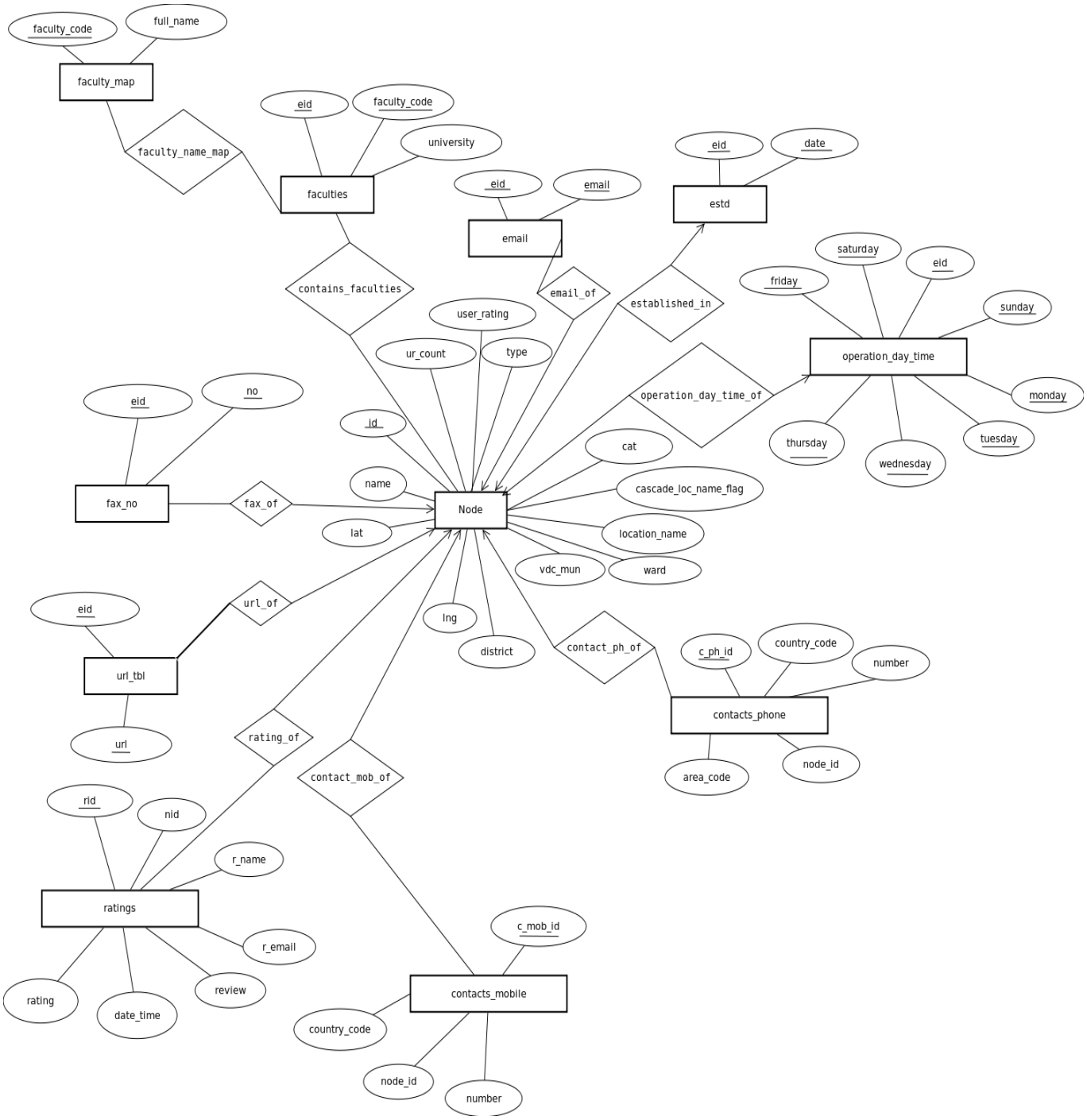


Fig 8 : ER Diagram

3. 3 Modules Description

3. 3. 1 Spelling correction module

Spelling error is very often in search query. Spelling error degrades search result.

When user submits query with spelling error, correction of spelling error improves search quality. For example if user submits query for what field as “hotle” instead of “hotel” which is very likely in terms of typo (el → le) as well as confusion with standard spelling of “hotel”. This type of spelling error degrades search quality. To improve search quality of local search engine, a spelling operator module is implemented whose job is to check presence of spelling error in user submitted query and to correct it.

3. 3. 1. 1 Steps for spelling correction

1) make vocabulary of words from data available in database of business listing
2) make lexical analysis of user submitted query
3) check whether the keyword in user submitted query is present in our vocabulary
4) if keyword is not present in vocabulary, spelling error is detected.
5) calculate string similarity degree of spelling error keyword with words in vocabulary
6) the word in vocabulary with highest similarity degree with keyword is picked up and if this words similarity degree with misspelled keyword is above spelling correction threshold, the word is selected as correct spelling of misspelled keyword.

3. 3. 1. 2 Issues in Spelling correction

3. 3. 1. 2. 1 How to measure similarity of two strings

Edit distance is widely used measure for string similarity. Edit distance measures distance as the (weighted or unweighted) number of operations(such as insertion, deletion, substitution) required to transform a string into another [14].

Pseudocode(unweighted edit distance)

```
editDistance(string s1, string s2)
{
```

```
    m = s1.length();
    n = s2.length();
    for (i = 0; i <= m; i++) {
        v[i][0] = i;
    }
    for (j = 0; j <= n; j++) {
        v[0][j] = j;
    }

    for (i = 1; i <= m; i++) {
        for (j = 1; j <= n; j++) {
            if (s1[i-1] == s2[j-1])
                v[i][j] = v[i-1][j-1];
            else
                v[i][j] = 1 + min(min(v[i][j-1],v[i-1][j]),v[i-1][j-1]);
        }
    }

    return v[m][n];
}
```

Fig 7: pseudocode for unweighted edit distance [13]

Issues with edit distance as only measure for string similarity

Consider the maximum allowed edit distance for spelling correction is 3 and consider vocabulary has word “bar” and user submitted word is “tup”. Since “tup” is not present in vocabulary, spelling error is detected and since allowed maximum edit distance for spelling correction is 3, top is corrected to “bar”(edit distance = 3 for “bar” and “top”).

This sort of spelling correction is some kind of non sensible, even though user has not submitted even single letter corresponding to word “bar”, “bar” is selected candidate for misspelled word “tup”.

The reason for this problem is length of pattern string. If it's length > 5 editDistacne based correction of with distance = 3 would be sensible. Then problem could be solved with making variable maximum edit distance threshold =1 for spelling correction for pattern with length <= 4. But when the pattern size

is very large say above 30, then edit distance of 3 is not very flexible. We want to give more flexibility for user say about 5 edit distance. So this approach is not feasible when pattern can grow arbitrarily long and we want to provide more flexibility.

This problem in edit distance is created due to substitution operation. We can remove this problem by not allowing substitution and since for our purpose, all operation costs 1, the resulting edit distance problem will be Longest Common Subsequence(LCS) problem.

It measures the distance of longest pairing of characters that can be made between both strings. An obvious measure for the closeness of two strings is to find the maximum number of identical symbols in them (preserving the symbol order).

Pseudocode:

```
LCS-Length(X, Y)
{
   m ← length[X]
   n ← length[Y]

   for i ← 1 to m
      c[i,0] ← 0
   for j ← 1 to n
      c[0,j] ← 0

   for i ← 1 to m
      for j ← 1 to n
         if (x[i ]== y[j]) {
            c[i,j] ← c[i-1,j-1] + 1
   return c[m][n];
}
```

But LCS consist of following problem.

Consider the pattern “edcated” with two target patterns “educated” and “uneducated”.

The lcs of

“edcated” and “educated” = 7

“edcated” and “uneducated” = 7

So, lcs could not distinguished either educated or uneducated (being semantically opposite words) as close to edcated. It should actually be “educated” and this could be solved with edit distance.

So what we can do is use lcs to identify only possible candidate for matching list. And then pick one with the minimum edit distance among the list with equal maximum lcs.

For case given above, the lcs candidate are “educated” as well as “uneducated”, and since edit distance of “educated” will be less than that of “uneducated”, “educated” is picked word that is close to misspelled word “edcated”.

3. 3. 1. 2. 2 Same similarity degree of pattern with two or more strings

There will also be cases where selection of closest candidate from above mentioned approach fails. Consider, the pattern “principl” and two target patterns “principle” and “principal”, both cases have lcs = 7 and edit distance = 1. So above, mentioned procedure fails here to suggest closest word to “principl”. This can be solve with the help of language model which gives the probability of occurrence of given word in the given language model. So “principle” is selected candidate if Unigram probability of it is greater than that of “principal” i.e

pr (“principle”) > pr(“principal”).

3. 3. 2 Query-expansion module

When user search for certain businesses or services, user would be happy if system does not only list businesses or services which exactly matches his search query but also those business or service which provides similar functions.

For example:

search for restaurants also resulting in inclusion of cafe (restaurants → restaurants + cafe)

search for quest house resulting in inclusion of lodges (guest house → guest house

+ lodge)

Query-expansion module is resposible for search query expansion. Query-expansion has implemented with “Thesaurus based query-expansion “. The thesaurus is constructed manually for the localsearch purpose based on the businesses or services that can be searched in the local search engine.

Steps in query-expansion:

1) identify the scope of query

2) if scope of query is class search then seek in thesaurus vocabulary for presence of query word(s) in thesaurus-vocabulary.

3) if word(s) is(are) present in thesaurus-vocabulary, retrieve semantically related tokens for query from thesaurus and add it to query.

4) during search also perform search for added semantically related tokens in query and add those results to original query results.

3. 3. 3 Query suggestion module

The job of query suggestion module is to suggest list of possible query for partial query supplied by the user so far. The purpose of query suggestion is to enable user better formulate their query quickly and more accurately. When user types misspelled word in field of query interface, the possible query with correct spelling is suggested to user on the fly i.e possible correct spelling is suggested to user before actual query is submitted to system. So query-suggestion in fact is result of online spelling correction. Query suggestion is provided on all three field of query interface.

3. 3. 3. 1 Query suggestion for district field

Query suggestion for district is based on dictionary based approach with vocabulary consisting of 75 districts of Nepal. Approximate string matching based on concept of string similarity measured used for spelling correction as mentioned on section **3. 3. 1. 2. 1** and **3. 3. 1. 2. 2** is used for choosing the candidate for suggestion such that one with higher similarity degree is suggested at top than

other with lower similarity degree than it.

3. 3. 3. 2 Query suggestion for location field

Query suggestion for location field is based on concept similar to district field but vocabulary is based on specified district by user. Thus, there are altogether 75 location vocabulary one for each of 75 districts of Nepal.

3. 3. 3. 2 Query suggestion for what field

What field suggestion is based on Bi-gram language model. So, system has capacity to learn query for what field based on past history of query. Let us assume, “kathford ” is partial query typed by user so far and also assume selected candidate for suggestion for “kathford” are “kathford international college” and “kathford institute”. Then “kathford international college ” will be ranked at top if p(“kathford international college ”) is higher than p(“kathford institute”) in corresponding language model.

First of all approximate string matching, as for location field, based on district is used to select possible candidate for suggestion. The candidate are then ranked on the basis of probability provided by the bi-gram language model such that higher the probability higher will be its rank. For what field there are altogether, 76 language model. One for national level search and remaining 75 as one for each of 75 districts of nepal.

A language model assigns a probability to a piece of unseen text, based on some training data [6]. Given the word w_{i-1}, bi-gram language model measures the probability of occurrence of word w_i after w_{i-1}. i.e

$$p(w_i|w_{i-1}) = \frac{c(w_{i-1}w_i)}{\sum_{w_i} c(w_{i-1}w_i)}$$

$$p(s) = \prod_{i=1}^{l+1} p(w_i|w_{i-1})$$

p(s) is bigram probability for sentence s in given language model [3].

When language model have 0 count for certain bigram say (w_{i-1} w_i), then system is going to give 0 probability for any pair which is not seen before in language model which is undesirable. To solve this problem we can assume that every word is seen once more and the technique is referred as Laplace Smoothing(Add one smoothing). The probability now is given by [3]

$$p(w_i|w_{i-1}) = \frac{1 + c(w_{i-1}w_i)}{\sum_{w_i}[1 + c(w_{i-1}w_i)]} = \frac{1 + c(w_{i-1}w_i)}{|V| + \sum_{w_i} c(w_{i-1}w_i)}$$

where |V| is size of vocabulary of language model.

3. 4 User Interface Design

User Interface is one of the major component that manages the interaction between human and machine. The goal of interaction between a human and a machine at the user interface is effective operation and control of the machine, and feedback from the machine which aids the operator in making operational decisions.

User interface is mainly responsible for

i) managing user inputs

ii) visualization of results

The followings are various components of user interface:

3. 4. 1 Query Interface

Query interface is interface for input of user query. It consists of three fields:

- district field: in which district where search to be performed is specified
- what field: in which what is actually searched is specified
- location field: in which location on district where search to be performed is specified

There is also facility for national search which could be done by checking the checkbox. For national search only what field is required since the search is made over the whole country. Other search is based on district division of Nepal and location filter over district is optional.

Fig 9: Query Interface interface

3. 4. 2 Visualization of search results

The following figure depicts how class search results are displayed to user.

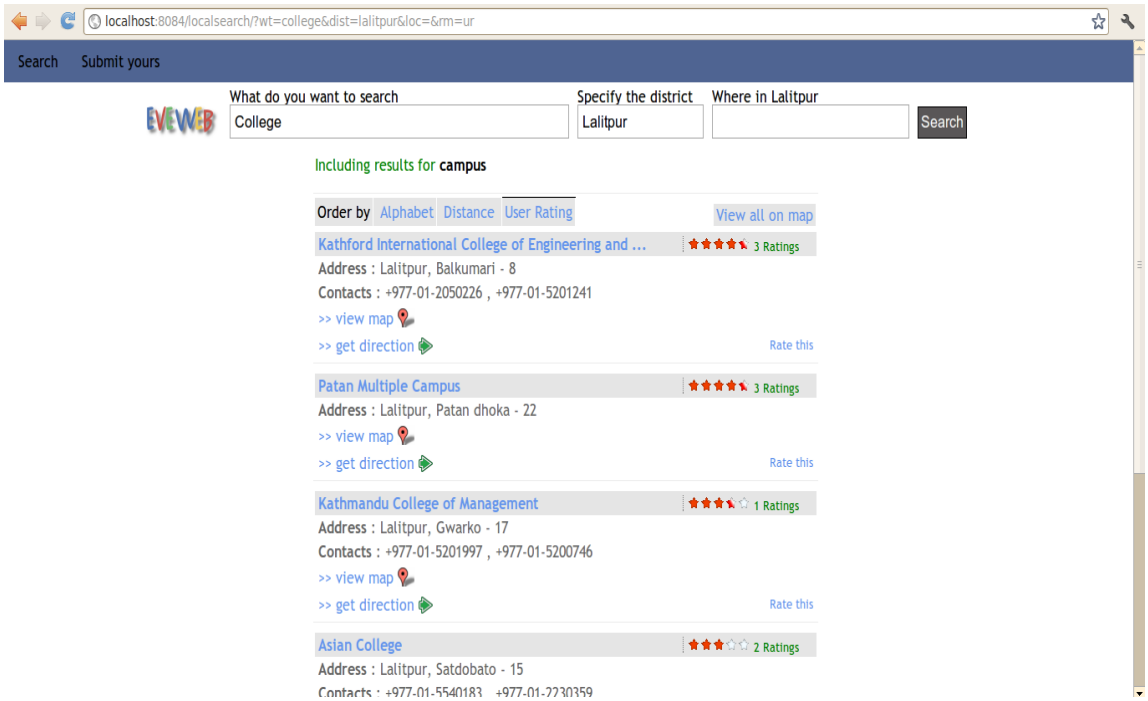


Fig 10 : Visualization of search result

The search result can be sorted on basis of

- alphabetical ordering
- distance
- user Rating.

The following figure depicts how particular entity is visualized to user.

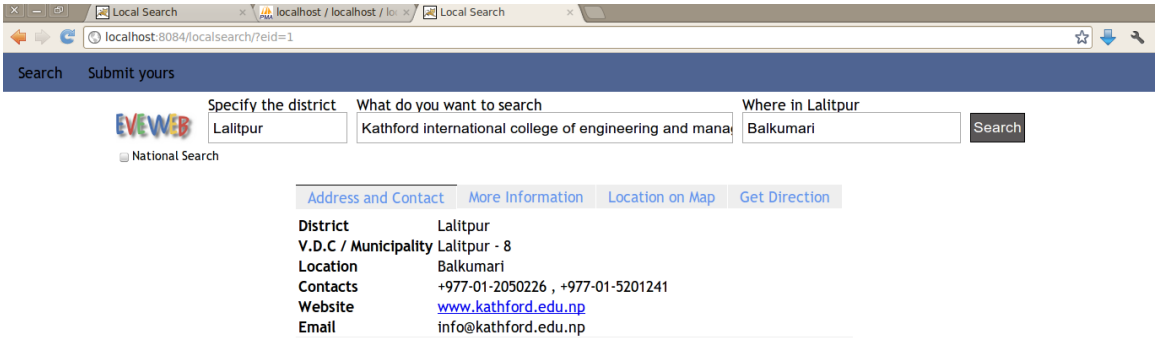


Fig 11 : Visualization of entity

The entity's more information is availabe on more information link of page. Entity can be located on map with location on Map link. Similarly direction to entity can be obtained from user specified origin to entity from get direction link of page.

3. 4. 3 Free Listing Request Form

This form is request form for any business or service wanting to be listed on our search engine. Business name or service, district, location on district with at least one contact number should be submitted.

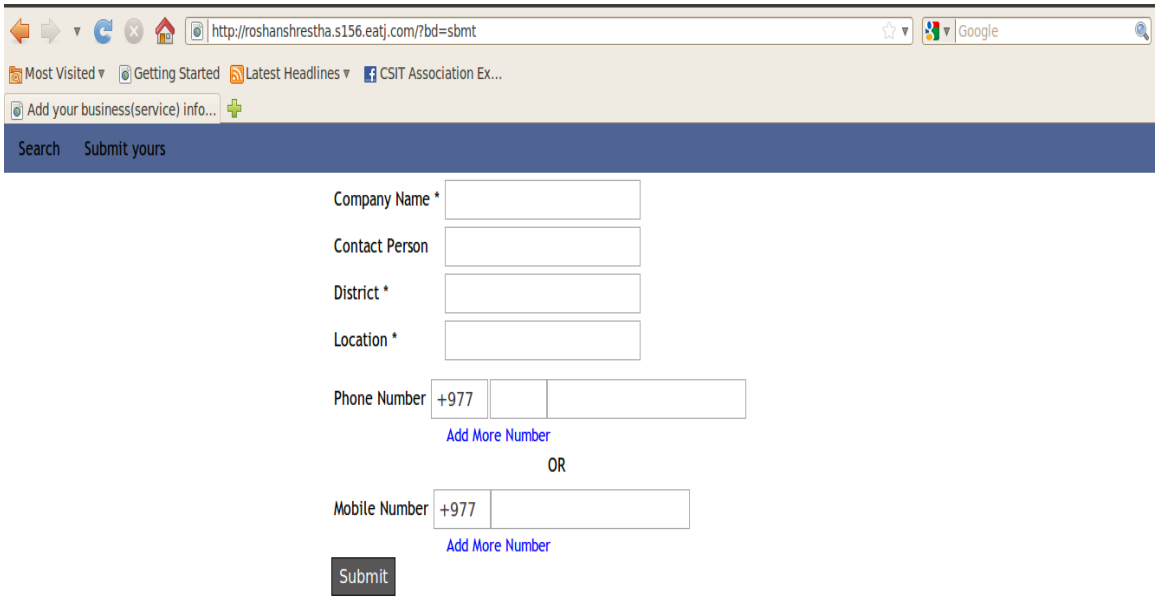
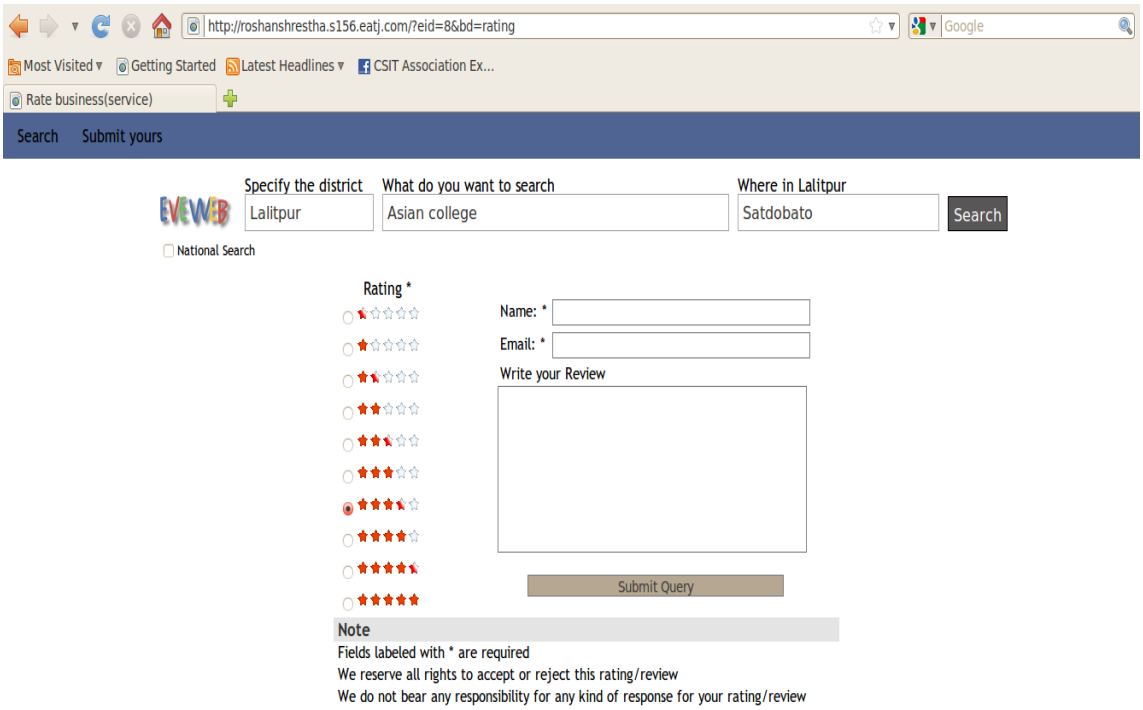


Fig 12 : Free listing request form

3. 4. 4 Rating Form

User can rate particular business or services by submitting the following form.


http://roshanshrestha.s156.eatj.com/?eid=8&bd=rating
Google
Most Visited
Getting Started
Latest Headlines
CSIT Association Ex...
Rate business(service)
Search
Submit yours
Specify the district
Lalitpur
What do you want to search
Asian college
Where in Lalitpur
Satdobato
Search
National Search
Rating *
Name: *
Email: *
Write your Review
Submit Query
Note
Fields labeled with * are required
We reserve all rights to accept or reject this rating/review
We do not bear any responsibility for any kind of response for your rating/review

Chapter IV

DEVELOPMENT AND TESTING

4. 1 Coding Tools

4. 1. 1 Front End Tools(Client Side Technology)

4. 1. 1.1 HTML

HTML, which stands for Hypertext Markup Language, is the predominant markup language for web pages. HTML uses markup tags to describe web pages. HTML is written in the form of HTML elements consisting of tags, enclosed in angle brackets (like <html>), within the web page content. HTML tags normally come in pairs like <h1> and </h1>. The first tag in a pair is the start tag, the second tag is the end tag (they are also called opening tags and closing tags). The purpose of a web browser is to read HTML documents and compose them into visual web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts in languages such as Javascript which affect the behavior or HTML markup.

4. 1. 1. 2 Javascript

JavaScript (sometimes abbreviated JS) is a prototype-based scripting language that is dynamic, weakly typed.

JavaScript is a client side scripting language meaning that JavaScript code is written into an HTML page. When a user requests an HTML page with JavaScript

in it, the script is sent to the browser and it's up to the browser to do something with it. It is used to make webpage more interactive, check or modify the contents of forms, change images, open new windows and write dynamic page content.

4. 1. 1. 3 AJAX

AJAX(Asynchronous Javascript and XML) is a group of interrelated web development methods used on the client-side to create interactive web applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. Data is usually retrieved using the XML Http Request Object. Despite the name, the use of XML is not needed, and the request need not be asynchronous.

4. 1. 1. 4 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation semantics (the look and formatting) of a document written in a markup language. Its most common application is to style web pages written in HTML and XHTML. CSS is designed primarily to enable the separation of document content (written in HTML or a similar markup language) from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed.

4. 1. 2 Back End Tools (Server Side Technology)

4. 1. 2. 1 Java

Java is an object-oriented programming language with a built-in application programming interface (API) that can handle graphics and user interfaces and that can be used to create applications or applets. Because of its rich set of API's, platform independence, Java can also be thought of as a platform in itself. Java also has standard libraries for doing mathematics. When a program is compiled, a byte code is produced that can be read and executed by any platform that can run Java.

4. 1. 2. 2 JSP

JavaServer Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. JSP is similar to PHP, but it uses the Java programming language.

To deploy and run JavaServer Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty etc, is required.

4. 1. 2. 3 Servlet

A servlet is a Java programming language class used to extend the capabilities of servers that can be accessed by a host application via a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by Web servers. A Servlet is an object that receives a request and generates a response based on that request.

Servlets can be generated automatically from JavaServer Pages (JSP) by the JavaServer Pages compiler. The difference between Servlets and JSP is that Servlets typically embed HTML inside Java code, while JSPs embed Java code in HTML.

4. 1. 2. 4 MySQL

MySQL is the world's most used open source RDBMS that runs as a server providing multi-user access to a number of databases. MySQL is characterized as a free, fast, reliable open source relational database.

MySQL is available for many platforms such as Linux, Unix and Windows. This flexible, multi-platform environment brings unity and scalability to development and Web application. MySQL is widely used for web applications.

4.2 Testing

Testing is an activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results. Although crucial to software quality and widely deployed by programmers and testers, software testing still remains an art, due to limited understanding of the principles of software. Testing is more than just debugging. The purpose of testing can be quality assurance, verification and validation, or reliability estimation. Correctness testing and reliability testing are two major areas of testing. Testing is a trade-off between budget, time and quality.

Our Software testing methodology is applied in following distinct phases:

4. 2. 1 Unit Testing

Unit Testing concentrates on each unit of the software as implemented in the source code. Unit testing will not catch every error in the program. It only test the functionality of the units themselves.

In unit testing we have done testing of :

i) every java class (including servlet) operational correctness.

ii)correctness for user interface corresponding to each JSP page.

iii)correctness for each module of written javascript.

iv)correctness of layout created by used HTML and CSS on browser.

4. 2. 2 Module Testing

Also known component testing , module testing is concerned with the testing of the smallest piece of software for which a separate specification exists. In module testing we have tested modules present in our search engine as follows:

i)User interface

It is tested for it's correct layout, user friendliness, consistency across different vendors' browsers . The user interface test succeeded for following browsers:

a) firefox >= version 3

b) internet explorer >= version 6

c) google chrome >= version 2

d) safari >= version 3

e) opera >= opera 10

ii)Text operation module

It is tested for it's correctness for

a) lexical analysis (tokenization)

Tokenizer passed its test in presence of noises such as whitespaces, commas“,”.

b) spelling correction

Based on threshold set for spelling correction, spelling correction is done correctly for string with 70% or greater similarity with target string .

iii)Query Operation module

Query operation module is tested for correctness of

i)query expansion

Based on associations available in thesaurus, query-expansion model operated correctly.

ii)identification of query scope

For case of correct formulation of query, identification of query scope is also accurate but when user query includes misspelled words, identification of query scope is still acurate for query with misspelled word with similarity degree >= 80%.

iii)query transformaton using relevance feedback

Query transformation is also done by query operation module according to its specification.

iv) Search Module

For entity searh,

Recall = 1

Precision = .93 (for test data set with 50 nodes)

For class search,

Recall = .98

Precision = 1(for test data set with 50 nodes)

Thus, search engine's

Recall = 0.99

Precision = 0.965

v) Ranking Module

Since there are no heuristic ranking metric, the rankings based on different rank metric also behaved to its specification.

vi) Map Operation Module

For location map service, provided accurate longitude and latitude, location map service is accurate.

But for direction service, since map service is based on google map and google do

not have very detail data about all driving and walkable paths of Nepal, the direction given in few cases could not meet its specification.

4. 2. 3 Integration Testing

It is the logical extension of unit testing . In this type of testing two or more units that have been already tested are combined into a component. The idea is to test combination of pieces and eventually expand the process to test the modules with those of the other groups.

In this, we have tested the integrated functionality of all the java classes, JSP pages, javascript modules, html files and all the interface as well as the control flow to check whether they are working properly or not. All the functionalities such as spelling correction, query-expansion, map files are tested to verify whether it is working properly and fulfilling the information requirement of the user or not.

4. 2. 4 System Testing

System testing was used to test the overall interaction of components. It involves the load, performance, reliability and security testing. So we can test the overall performance and perfection of that software. The purposed system is also tested by using this technique to make our project more reliable. In this test we take a test machine and tested our system by using that machine. This test is done usually in simulated environment. There are two types of system testing we have done in our project.

4. 2. 4. 1 Alpha Testing

The system was tested by the project members individually and in group so as to determine the errors. The system was tested in concern with the requirements specified in Requirement Analysis to check if the system satisfied the requirements.

4. 2. 4. 2 Beta Testing

The product has already passed through the first-level, internal pilot-test called alpha test and major defect has been removed. But the developed system may still have some minor problems that require user participation. It is released to selected users for testing under normal, everyday conditions of use to remove the remaining error.

CHAPTER V

LIMITATION AND CONCLUSION

5.1 Limitations and Future Enhancements

Currently, acronym based searching feature is not present. But many businesses or services are known by many peoples just by their acronym. So, one future enhancement involves, addition of acronym based search feature.

Further, special design of web pages forwarded to browser is not made for mobile devices like cell phones, PDAs etc. So, special design of web pages for mobile devices is another possible futre enhancement.

5. 2 Conclusions

Aside from the quality of search, local search engine developed in the project is scalable too. Though it currently does not have data from whole region of country, the design of search engine based on geographical division of Nepal made it scalable enough to operate over the data over the whole region of country. Use of categorical filter, identification of query scope also contributed for scalability.

Use of features such as spelling correction both offline spelling correction as well as online spelling correction, query expansion, query suggestion, fuzzy string matching based search algorithm improved search results quality. Spelling correction has improved both recall and precision of search and query expansion, fuzzy string matching based search algorithm has improved the recall. Query suggestion has made searching easy enabling user to better formulate their query quickly and in more relevant manner. Since query suggestion is implemented in form of online spelling correction, it improved the search result as well as search experience of user.

References

[1] Aditi Sharan2 and Hazra Imran1, THESAURUS AND QUERY EXPANSION International Journal of Computer science & Information Technology (IJCSIT), Vol 1, No 2, November 2009

[2] Ali Shiri and Crawford Revie, Query Expansion Behaviour within a Thesaurus-Enhanced Search Environment: A User-Centered Evaluation , Journal of the American society for information science and technology 2006

[3] Bill MacCartney, NLP Lunch Tutorial: Smoothing, April 2005

[4] Bo-June (Paul) Hsu and Huizhong Duan, Online Spelling Correction for Query Completion, 2011

[5] Bruno Martins and Mário J. Silva, Spelling Correction for Search Engine by Queries, 2004

[6] Djoerd Hiemstra, LANGUAGE MODELS University of Twente, 2009

[7] E. Brill and S. Cucerzan and Spelling correction as an iterative process that exploits the collective knowledge of web users, 2004

[8] Gonzalo Navarro , A Guided Tour To Approximate String Matching, 2001

[9] H.Hakonen , L.Bergroth and T.Raita , A Survey of Longest common subsequence algorithms, 2000

[10] KENNETH W. CHURCH and WILLIAM A. GALE , Probability scoring for spelling correction AT&T Bell Laboratories, 600 Mountain Ave., Murray Hill, NJ, USA Received August 1990 and accepted December 1990

[11] Mary Bowling Blizzard,Local Search Engine Marketing,Internet Marketing Inc 2008

[12] www.en.wikipedia.org/wiki/Local_search_(Internet) (July 15, 2012)

[13] www.expertlaw.com/library/practice_management/website_roi.html (August 1, 2012)

[14] www.wikipedia.org/wiki/Edit_distance

[15] www.movable-type.co.uk/scripts/latlong.html (July 15, 2012)

[16]www.nlp.stanford.edu/IR-book/html/htmledition/query-expansion-1.html(July 2012)

APPENDIX

PART A : LOCAL SEARCH ENGINE OPERATION SNAPSHOTS

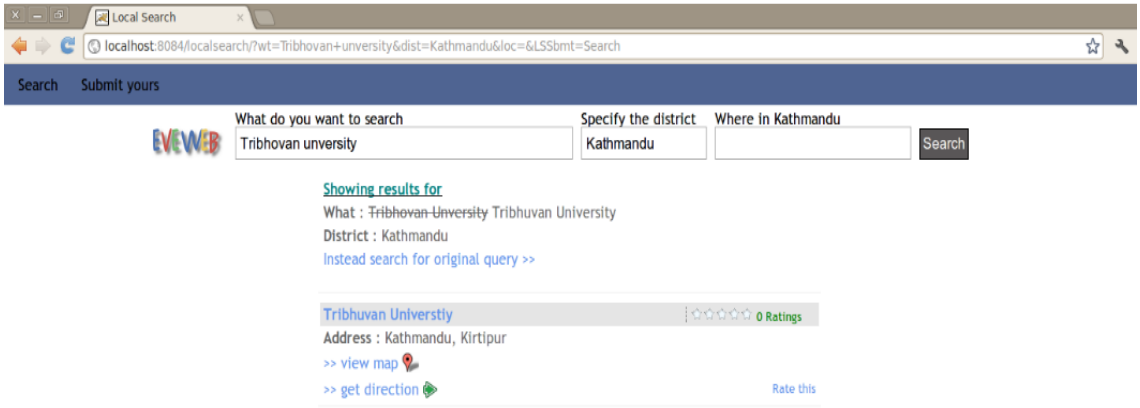


Fig 14: Snapshot for spelling correction

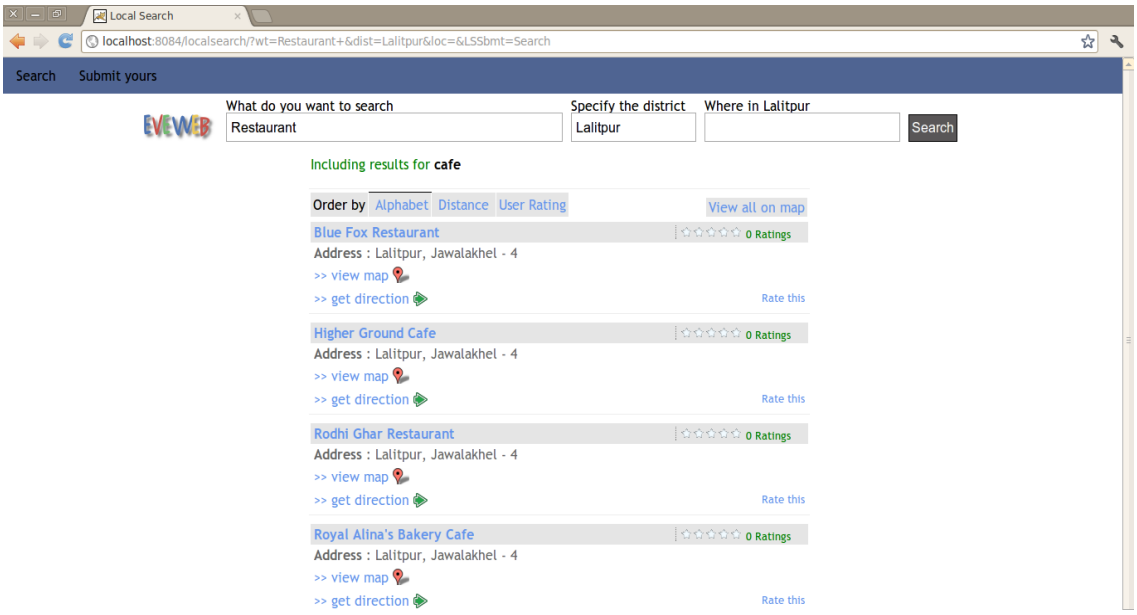


Fig 15 : Snapshot for query expansion

Fig 16: Query suggestion what field

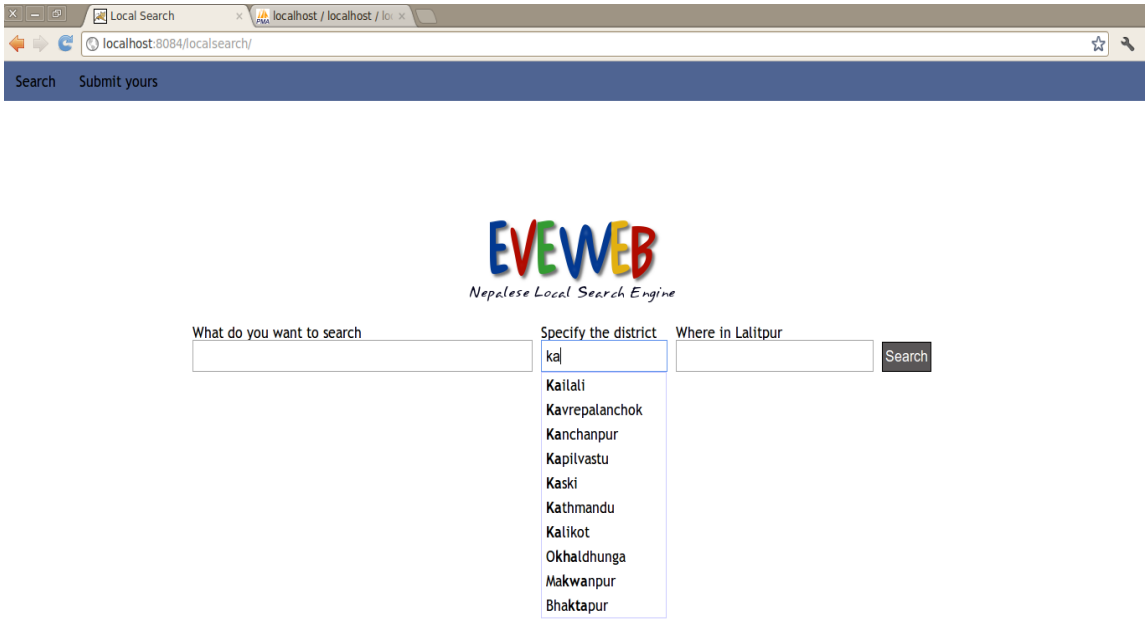


Fig 17 : Query suggestion district field

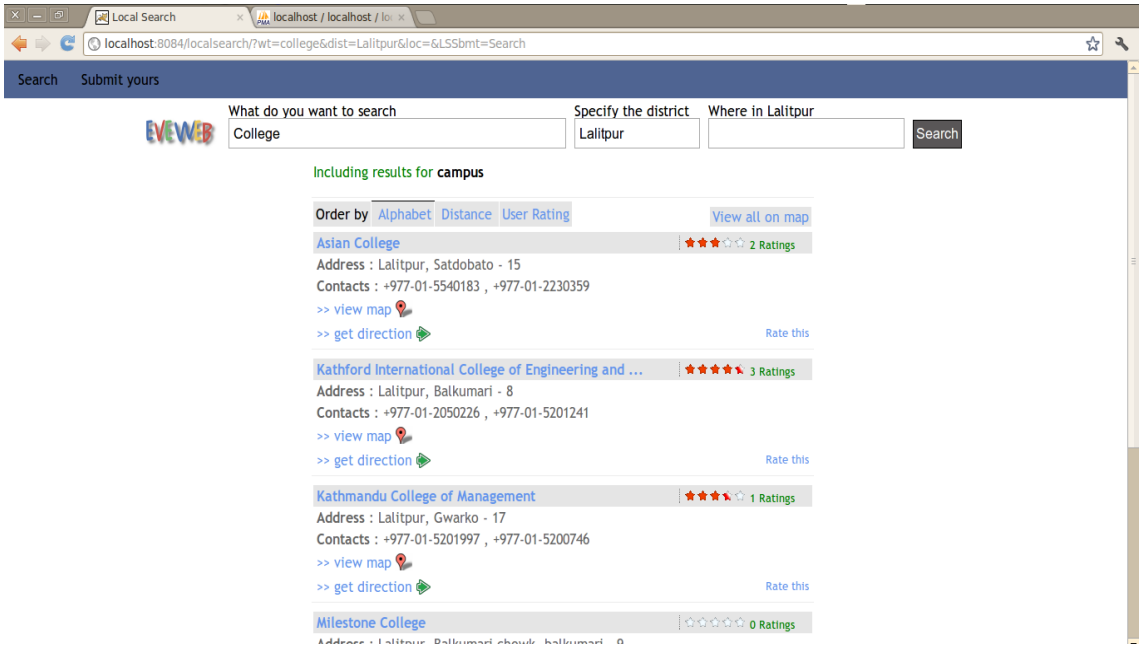


Fig 18 : Alphabetical Ranking

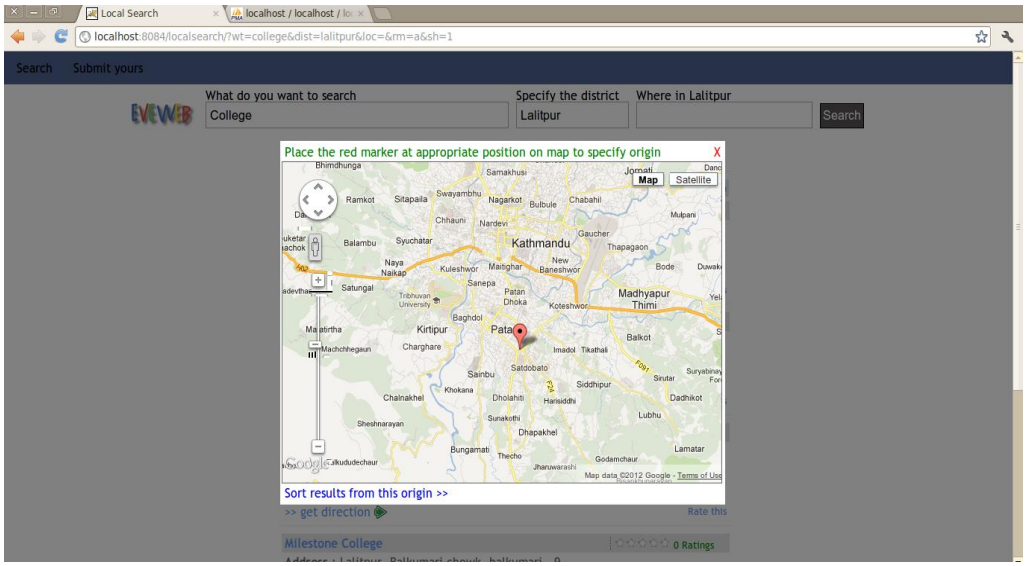


Fig 19: Distance ranking origin entry

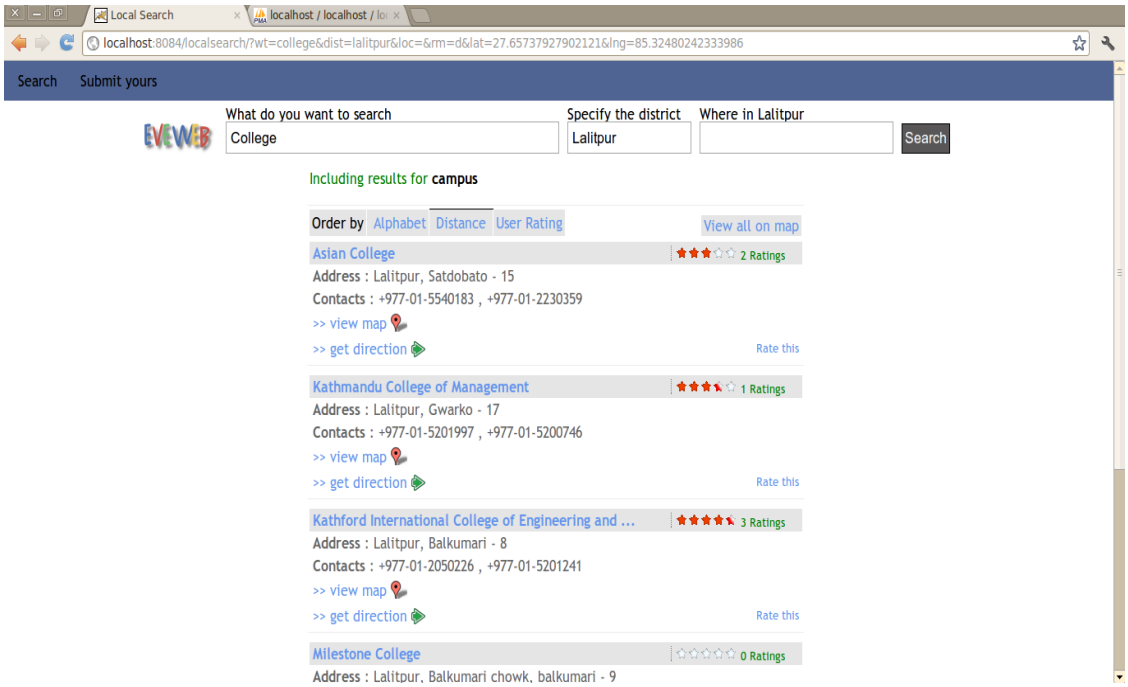


Fig 20 : Result for distance ranking

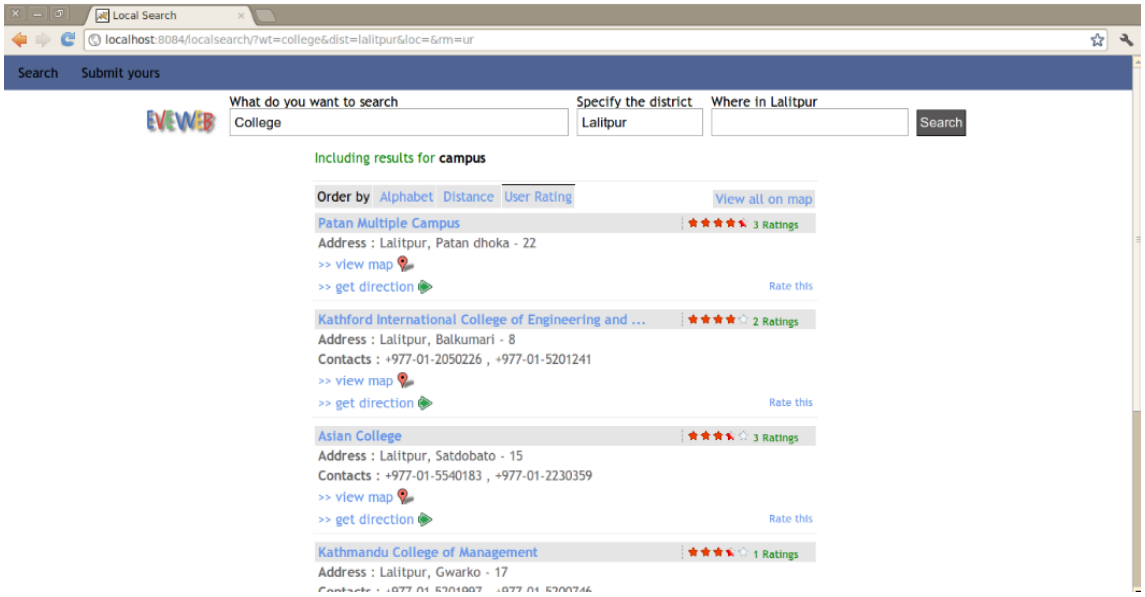


Fig 21 : Ranking based on user rating

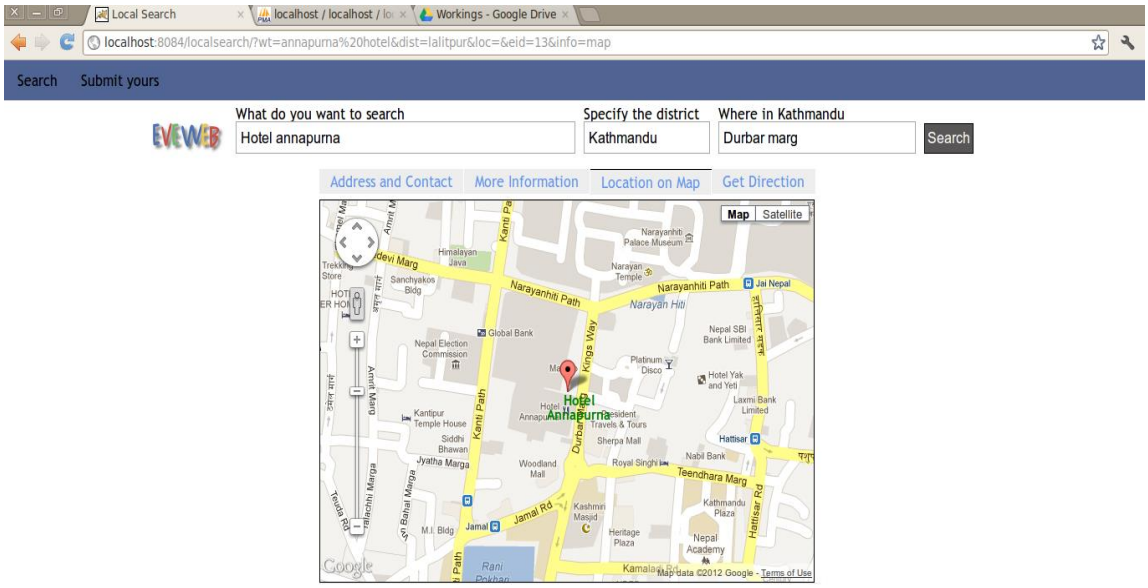


Fig 22 : Location on map

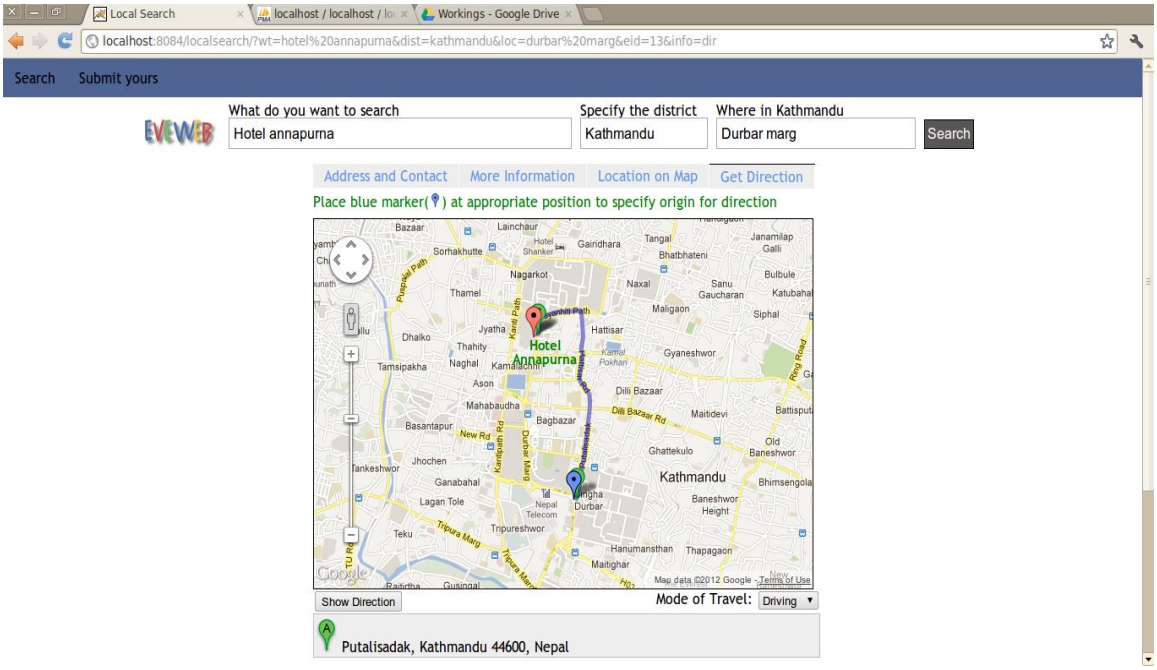
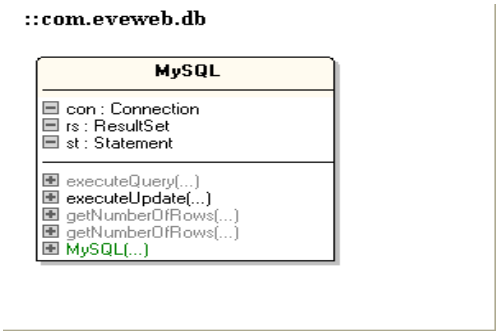
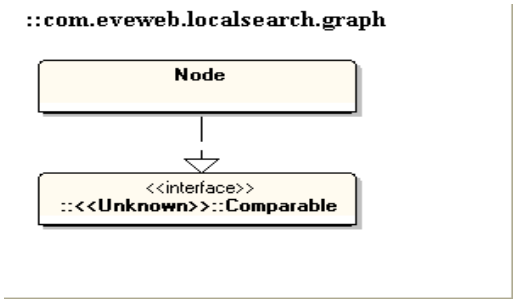


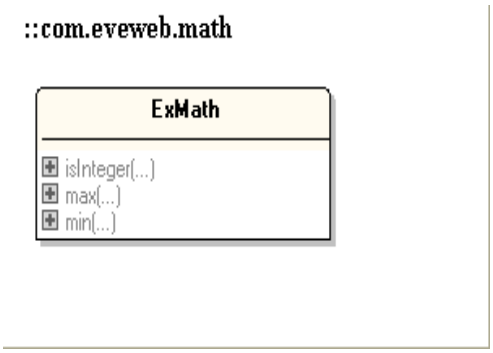
Fig 23: Direction display on map

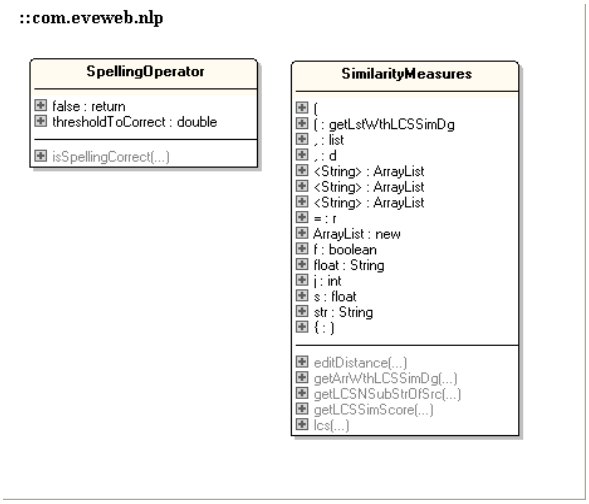
APPENDIX : PART B

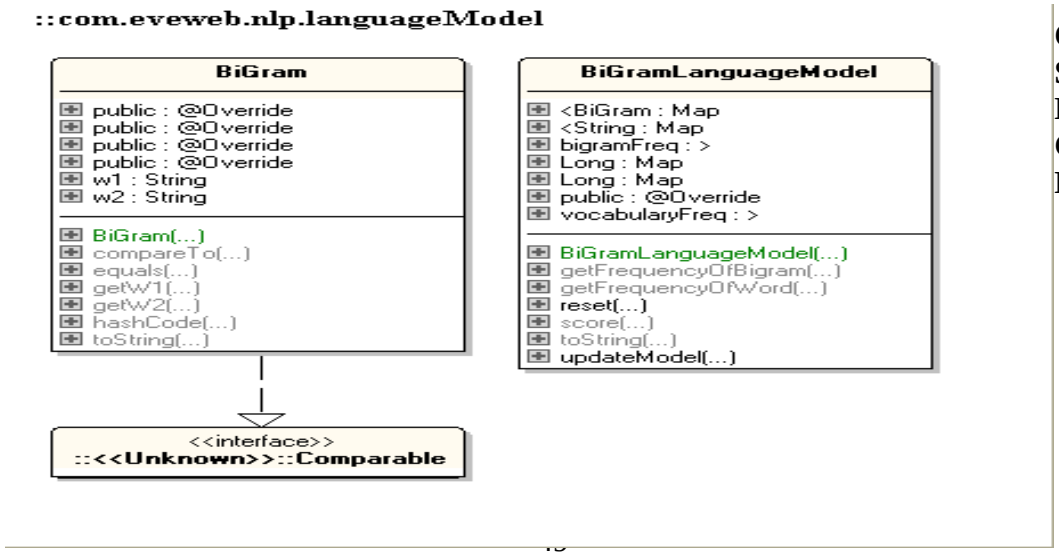
CLASS DIAGRAMS



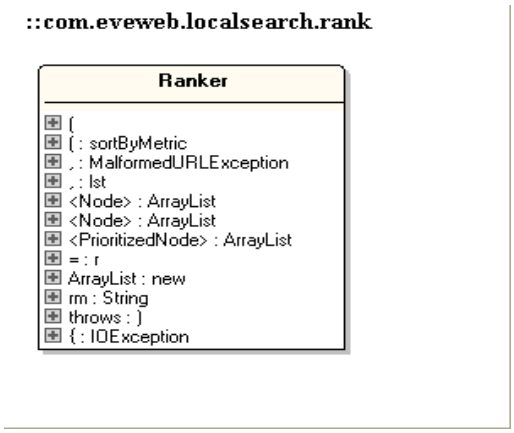


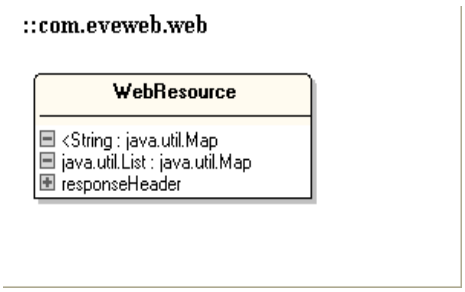


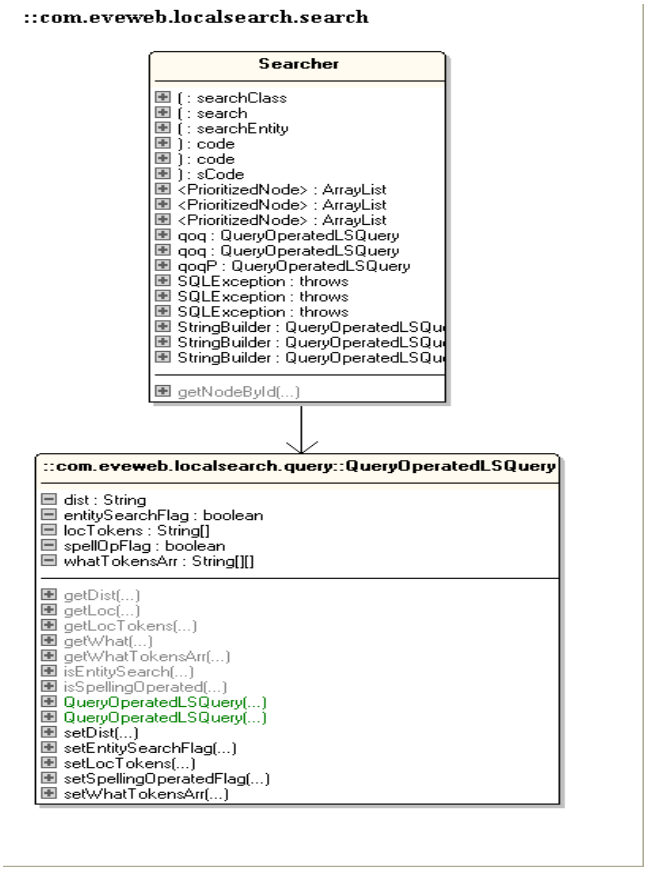




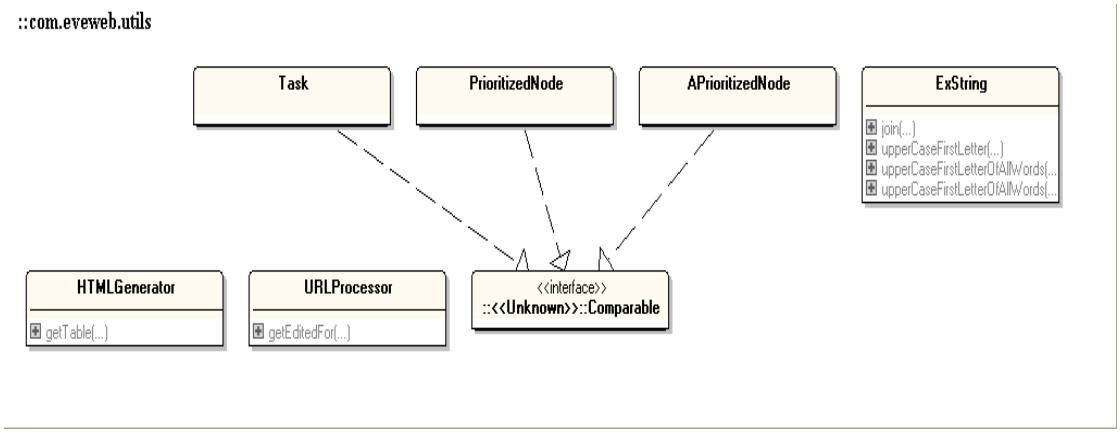
CLASS DIAGRAMS

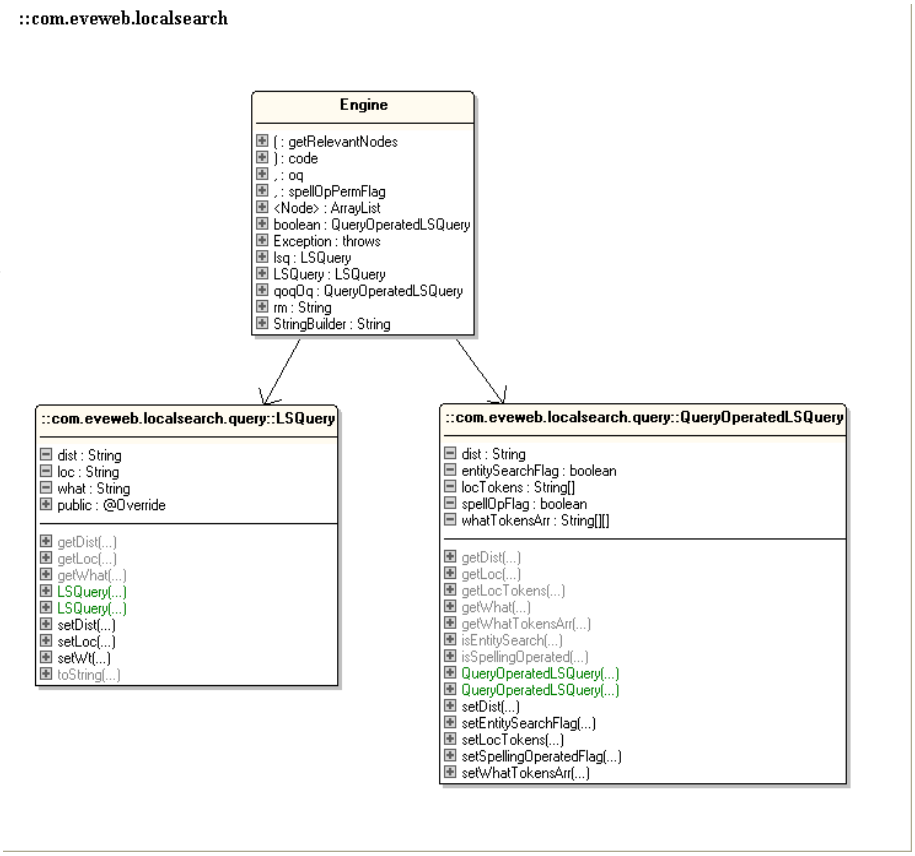

::com.eveweb.localsearch.rank
Ranker
(
(: sortByMetric
, : MalformedURLException
, : Ist
<Node> : ArrayList
<Node> : ArrayList
<PrioritizedNode> : ArrayList
= : r
ArrayList : new
rm : String
throws :)
{ : IOException


::com.eveweb.web
WebResource
<String : java.util.Map
java.util.List : java.util.Map
responseHeader


::com.eveweb.localsearch.search
Searcher
(: searchClass
(: search
(: searchEntity
) : code
) : code
) : sCode
<PrioritizedNode> : ArrayList
<PrioritizedNode> : ArrayList
<PrioritizedNode> : ArrayList
qoq : QueryOperatedLSQuery
qoq : QueryOperatedLSQuery
qoqP : QueryOperatedLSQuery
SQLException : throws
SQLException : throws
SQLException : throws
StringBuilder : QueryOperatedLSQu
StringBuilder : QueryOperatedLSQu
StringBuilder : QueryOperatedLSQu
getNodeById(...)
::com.eveweb.localsearch.query::QueryOperatedLSQuery
dist : String
entitySearchFlag : boolean
locTokens : String[]
spellOpFlag : boolean
whatTokensArr : String[][]
getDist(...)
getLoc(...)
getLocTokens(...)
getWhat(...)
getWhatTokensArr(...)
isEntitySearch(...)
isSpellingOperated(...)
QueryOperatedLSQuery(...)
QueryOperatedLSQuery(...)
setDist(...)
setEntitySearchFlag(...)
setLocTokens(...)
setSpellingOperatedFlag(...)
setWhatTokensArr(...)

CLASS DIAGRAMS





APPENDIX : PART C

CODE SNAPSHOTS

```
public static String getCorrectSpellingOf(String w, ArrayList<String> voc) {
    PriorityQueue<Task> pqLCS = new PriorityQueue<Task>();
    double score;
    for (String s : voc) {
        score = SimilarityMeasures.getLCSSimScore(w, s, true);
        if (score >= thresholdToCorrect) {
            pqLCS.add(new Task(s, score));
        }
    }
    if (!pqLCS.isEmpty()) {
        Task t;
        t = pqLCS.remove();
        if (pqLCS.isEmpty()) {
            return t.getTaskName();
        } else {
            double maxSim = t.getPriority();
            PriorityQueue<ATask> pqEditDistance = new PriorityQueue<ATask>();
            pqEditDistance.add(new ATask(t.getTaskName(), SimilarityMeasures.editDistance(w, t.getTaskName(), true)));
            Task tmp = pqLCS.remove();
            boolean conflictFlag = false;
            while (!pqLCS.isEmpty() && tmp.getPriority() == maxSim) {
                pqEditDistance.add(new ATask(tmp.getTaskName(), SimilarityMeasures.editDistance(w, tmp.getTaskName(), true)));
                tmp = pqLCS.remove();
                conflictFlag = true;
            }
            if (!conflictFlag) {
                return pqEditDistance.remove().getTaskName();
            } else {
                ATask tLCS = pqEditDistance.remove();
                double minEd = tLCS.getPriority();
                PriorityQueue<Task> pqLM = new PriorityQueue<Task>();
                pqLM.add(new Task(t.getTaskName(), ObjectCorpus.blm.score(t.getTaskName())));
                ATask tmpLM = pqEditDistance.remove();
                while (!pqEditDistance.isEmpty() && tmpLM.getPriority() == minEd) {
                    pqLM.add(new Task(tmpLM.getTaskName(), ObjectCorpus.blm.score(tmp.getTaskName())));
                    tmpLM = pqEditDistance.remove();
                }
                return pqLM.remove().getTaskName();
            }
        }
    }
    return w;
}
```

Fig 24 : Spelling correction

CODE SNAPSHOTS

```
     * This function updates language model with given sentence
     *
     * @param {String} sentence new sentence to language model
     */
    public void updateModel(String sentence) {
        if (sentence.trim().equals("")) {
            return;
        }
        int i;
        String toks[] = sentence.trim().split("(\\s)+");

        String w = toks[0];

        if (vocabularyFreq.containsKey(w)) {
            Long val = new Long(vocabularyFreq.get(w));
            vocabularyFreq.put(w, val + 1L);
        } else {
            vocabularyFreq.put(w, new Long(1L));
        }

        String w0;
        BiGram b;
        for (int j = 1; j < toks.length; j++) {
            w = toks[j].trim();
            if (vocabularyFreq.containsKey(w)) {
                Long val = new Long(vocabularyFreq.get(w));
                vocabularyFreq.put(w, val + 1L);
            } else {
                vocabularyFreq.put(w, new Long(1L));
            }
            w0 = toks[j - 1].trim();
            b = new BiGram(w0, w);
            if (bigramFreq.containsKey(b)) {
                Long val = new Long(bigramFreq.get(b));
                bigramFreq.put(b, val + 1L);
            } else {
                bigramFreq.put(b, new Long(1L));
            }
        }
    }
```

Fig 25 : Language Model Update

```
    /**
     * Takes a sentence as argument and returns the probability of the
     * sentence using system's language model.
     *
     * @return {double} score for given sentence in system's language model
     */
    public double score(String sentence) {
        String tokens[] = sentence.trim().split("(\\s)+");
        long sum = 0;
        for (long l : vocabularyFreq.values()) {
            sum += l;
        }
        double p = (getFrequencyOfWord(tokens[0]) + 1.0) / (vocabularyFreq.size() + sum);
        long c;
        for (int i = 1; i < tokens.length; i++) {
            c = getFrequencyOfBigram(new BiGram(tokens[i - 1], tokens[i]));
            p *= ((c + 1.0) / (getFrequencyOfWord(tokens[i - 1]) + vocabularyFreq.size()));
        }
        return p;
    }

    /**
     * This function resets the language model
     */
    public void reset() {
        bigramFreq = new HashMap<BiGram, Long>();
        vocabularyFreq = new HashMap<String, Long>();
```

Fig 26 : Language Model Score

CODE SNAPSHOTS

```
} else if (rm.matches("d:\\d+.\\d+:\\d+.\\d+:\\w+")) {

    PriorityQueue<APrioritizedNode> apqn = new PriorityQueue<APrioritizedNode>();
    String strArr[] = rm.split(":");
    String origin = strArr[1].trim() + "," + strArr[2].trim();
    String destinations;
    int rLen = r.size();
    if (rLen > 0) {
        destinations = r.get(0).getLat() + "," + r.get(0).getLng();
        for (int i = 1; i < rLen; i++) {
            destinations += "|" + r.get(i).getLat() + "," + r.get(i).getLng();
        }
        boolean loopFlag = true;
        String url = "http://maps.googleapis.com/maps/api/distancematrix/json?origins=" + origin + "&destinations=" + destinations + "&mode=" + strArr[3] + "&sensor=false";
        WebResource distMat = null;
        String json = "";
        while (loopFlag) {
            loopFlag = false;
            try {
                distMat = new WebResource(url);
                json = (String) distMat.getContent();
            }catch (Exception e) {
                int R = 6371;
                double olat = Double.parseDouble(strArr[1].trim()), olng = Double.parseDouble(strArr[2].trim());
                double dlat, dlng, d;
                apqn.clear();
                for (Node n : r) {
                    dlat = n.getLat();
                    dlng = n.getLng();
                    d = Math.acos(Math.sin(olat) * Math.sin(dlat)
                            + Math.cos(olat) * Math.cos(dlat)
                            * Math.cos(dlng - olng)) * R;
                    apqn.add(new APrioritizedNode(n, d));
                }
                while (!apqn.isEmpty()) {
                    t.add(apqn.remove().getNode());
                }
                return t;
            }
```

Fig 27 : Distance Ranking

```
            }
            Pattern pTxt = Pattern.compile("\"( )*text( )*\"( )*:( )*\"( )*[\\w. ]+( )*\"");
            Pattern pStatus = Pattern.compile("\"( )*status( )*\"( )*:( )*\"( )*[\\w. ]+( )*\"");

            Matcher mTxt = pTxt.matcher(json);
            Matcher mStatus = pStatus.matcher(json);
            String status="";
            double d;
            for (int i = 0; i < rLen; i++) {
                if (mStatus.find()) {
                    status = mStatus.group().split(":")[1].trim();
                }
                if (status.equals("\"OK\"")) {
                    d = Double.MAX_VALUE;
                    if (mTxt.find()) {
                        d = Double.parseDouble(mTxt.group().split(":")[1].trim().split(" ")[0].substring(1));
                    }
                    apqn.add(new APrioritizedNode(r.get(i), d));
                    if (mTxt.find());
                }
            }
        }
        while (!apqn.isEmpty()) {
            t.add(apqn.remove().getNode());
        }
```

Fig 28 : Extracting distance from distance matrix in json

CODE SNAPSHOTS

```
<script type="text/javascript"
        src="http://maps.googleapis.com/maps/api/js?key=AIzaSyD6FwI-furH9qyb_xl7connUTliOSOHKo4&sensor=false">
</script>
<script type="text/javascript" src="js/com/eveweb/geo/label.js" ></script>
<script type="text/javascript">
    (function(){
        gD= function(){}
        gD.directionsDisplay;
        gD.directionsService = new google.maps.DirectionsService();
        gD.dLat=<%=g.getLat()%>;
        gD.dLng=<%=g.getLng()%>;
        gD.oLat=gD.dLat+0.002222;
        gD.oLng=gD.dLng-0.012222;
        gD.initialize=function() {
            var latlng = new google.maps.LatLng(gD.dLat,gD.dLng);
            gD.directionsDisplay = new google.maps.DirectionsRenderer();
            var myOptions = {
                zoom: 14,
                center: latlng,
                mapTypeId: google.maps.MapTypeId.ROADMAP,
                mapTypeControl: false
            };
            var map = new google.maps.Map(document.getElementById("map_canvas"),myOptions);
            gD.directionsDisplay.setMap(map);

            gD.directionsDisplay.setPanel(document.getElementById("directionsPanelD"));
            var text='<%=g.getName()%>';
            var te=text.uCFOfAlWrdIgnr().split(' ');
            var i=0;
            var td='<div style="text-align:center;">'+te[0]+'</div>';

            for(i=1;i<te.length;i++){
                td+='<div style="text-align:center;">'+te[i]+'</div>';
            }
            var dMarker = new google.maps.Marker({
                position: latlng,
                map: map,
                desc: td
            });

            var oMarker = new google.maps.Marker({
```

Fig 29 : Direction display