A Major Project Final Report on

InNepal.net: A site search portal

Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelors of Engineering in Information Technology

Under Pokhara University

Submitted by:

Arjun Saud, 161505

Atul Kayastha, 161506

Sakar Bajimaya, 161528

Shovit Nepal, 161536

Under the supervision of

Mr. Subash Manandhar

Date: 24 July 2021

Department of Information Technology

NEPAL COLLEGE OF INFORMATION TECHNOLOGY

Balkumari, Lalitpur, Nepal

ACKNOWLEDGEMENT

First of all, the whole team of "**Innepal.net"** is extremely thankful for the valuable guidelines and feedbacks from our project supervisor **Mr. Subash Manandhar**. We are glad and feeling privileged to be able to present our process of transforming idea into a project that simulates the real-world problem-solving product. We would also like to extend our sincere thanks to **Nepal College of Information Technology** for providing us with an opportunity to perform a major project for the partial fulfillment of the requirements for the degree of B.E in Information Technology.

It is also our privilege to express our sense of gratitude and respect to our project coordinator Professor **Dr. Roshan Chitrakar** for his unparalleled knowledge of moral fiber and judgment along with his know-how-based method which was an immense support for the completion of our project.

We would also like to express our thanks to all the staffs under Department of Information Technology for helping us in the successful completion of this project. Lastly, we would like to thank our parents and beloved friends who helped and motivated us throughout the completion of our project.

ABSTRACT

With the increase in online presence of locals as well as national business in Nepal, the need for their quick access and easy reachability has become a necessary commodity. Internet penetration has increased and so is the data generated by Nepalese.

***InNepal.net** is a site search portal whose primary focus is to provide search results of requested queries based on the user's search term as well as their geographical location. The user enters desired search term into the search field and our search engine then looks into its index for relevant websites and displays them in the form of list. It aims to be a centralized site search engine system by providing a viable platform for both businesses and customers under a single entity. According to Google Analytics and various other trend analyzers such as Yahoo trends, the most searched keyword from consumers all over in Nepal has the letters "In" and "Nepal". We will be providing the localized search engine which gives the detailed information of their search queries in formatted as per their location aided by the preferences of collected data in their local region.*

Data is an important aspect for the implementation of this project. Taking the real time search data according to geographical locations, we can maximize the efficiency of the search engine which is in the core of this project. The search engine is made and implemented by using Elastic Search, Log stash and Kibana, with help of Beats for real time monitoring and data analyzing. API's can be generated from the search engine which could hence be provided to Nepalese business to help them in their websites to improve their efficiency by targeting the right consumer group from wide and specific locations as well.

Keywords: Search Engine, Google Analytics, Elastic Search, Log stash, Kibana, API.

Table of Contents

LIST OF FIGURES

1. INTRODUCTION

1.1 BACKGROUND

In context of Nepal, how often do we find the exact website and its related contents in the existing search portals? Apart from websites which has implemented digital marketing techniques such as Search Engine Optimization or page ranking upon their websites, it is often rare that we could find the contents at one search. According to Google Analytics and various other trend analyzers such as Yahoo trends, the most searched keyword from users in Nepal has the letters “in Nepal” on their keywords. We will be providing a localized search portal which gives the detailed information of their search queries in formatted as per their location aided by the preferences of collected data in their local region.

“Innepal.net” is a site search portal for Nepal that intends to provide a suitable platform to all the users by providing all the information related to their queries ending with “in Nepal” for instance “Top IT Colleges in Nepal”, “Bikes in Nepal”, “Smartphones under Rs 50,000 in Nepal” and similar. It provides all the users an ability to search requested queries based on their desired keywords as well as their geographical location. The user enters desired search term into the search field and our search portal then looks into its index for relevant websites and displays them in the form of list. We aim our system to be a centralized site search engine system by providing a viable platform for both businesses and customers under a single entity.

1.2 PROBLEM STATEMENT

The traditional search engine was able to produce the results for the user's requirements. But the question of efficient result is the biggest challenge for any search engine project. Along with that users are faced with diverse results. Few times only users are found to be happy with the search results given by search engines. Most of the time, search engines have failed to produce the required search result requested by the user. Most of the users not very much satisfied with the existing system.

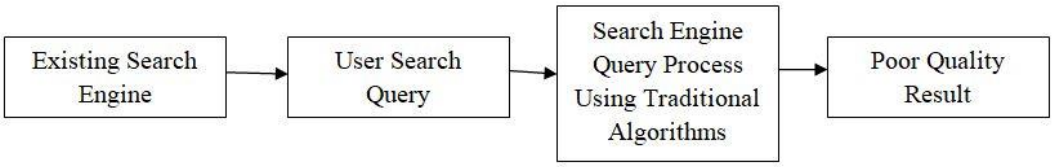

Figure 1: Existing System Architecture

The existing search engines in Nepal like **Bhetincha.com** shows all the results based on user queries. Also they are mostly based on traditional process of sorting and indexing. This makes working smaller websites of small or local businesses to fall behind. Through our system, we decided to use artificial intelligence to solve our problem more easily and precisely. Based on their keywords ending with "in Nepal" on their search queries such as "Mobile phones in Nepal" or "Cars in Nepal", users are able to get precise results based on their queries and their geographical location as well as businesses who wishes to make their websites appear at the top of our page. The limitations of the existing system are as follows:

A. For Nepalese customers/consumers:
   - Lack of generalized and customer centric platform where everything can be easily found under one roof. A platform where they can compare popular business and products around them.
   - People from one specific location should get personalized recommendations. For instance, users from Bhaktapur should get search results of their queries completely different from that of people searching from Kathmandu.

- Lack of centralized site search in Nepal.
- Time Consuming to find the localized results through the global search sites.
- Diversified results.

B. For Nepalese business:

- Healthy competition /ranking among themselves.
- Inefficient search integration used by major as well as minor business all over Nepal.
- Lack of proper machine learning models implemented on local search engines.

Thus our project solves the above issues in the following ways:

I. Provide a centralized and local site search system to promote local businesses.
II. Based on deep learning concepts for better results.
III. Results based on geographical location. For instance, users from Lalitpur will receive different search results than that of users from Bhaktapur searching the same queries.
IV. Use of Deep Learning Models to make local websites appear at the top based on site visit count.

1.3 PROJECT OBJECTIVES

The objective of this project is to search user's problem of getting results based on their location, as well as promote local websites to appear at the top. The portal does not always produce accurate results as it depends upon various factors like number of trains, number of pages sorted and more. The pages with the most visit shall appear at the top. Thus, the objectives of our project is highlighted as follows:

i. Create a centralized site search system that would promote local businesses of Nepal.
ii. Create a new method of Digital Marketing in Nepal.
iii. Improve the search response to users based on their geographical location.
iv. Introduce Deep Learning Model in a local search site engine in Nepal.
v. Create a site search engine and generate APIs for Nepalese markets.
vi. Provide a common platform to both business and consumers.
vii. Understanding search indexes.
viii. Identify the global trends and analyze the available site mapping systems.
ix. Creating API response.
x. To fill the void between the local business and customers.

1.4 SIGNIFICANCE OF THE STUDY

The significance of our system is that it allows us to use the already available websites and their contents as our dataset and use methods of sorting and indexing to display results to the users. Furthermore, the local businesses that are willing to display their websites appear at the top of our search engine shall directly interact with us and we can artificial intelligence models to display them accordingly based on user's location and number of their site visits.

1.5 SCOPE AND LIMITATIONS

The scope of the project is to identify user's queries. To meet the requirements, we use the simple and basic indexing-sorting approach.

The following could be our project's scope:

- Any users can easily interact with the simple system design.
- Any users can type his/her search queries and get their desired results.
- Users from all over the country could get precise and different results of their queries based on their searched terms.

The following could be our project's limitations:

- The limitation of this system is that it cannot 100% accurate in giving user's their best search results.
- Users may type his/her keywords incorrectly or even make grammatical mistakes while searching.

2. TEAM MEMBERS AND DIVIDED ROLES

NAME	ROLES	RESPONSIBILITIES
Arjun Saud	Backend Developer Database	• Development of backend API and testing. • Timely review of database data.
Atul Kayastha	Backend and UI Design	• Development of backend API and testing. • Creating user friendly web interface.
Sakar Bajimaya	Backend Developer Documentation	• Backend integration. • Development of documentation.
Shovit Nepal	Frontend and UI Design Documentation	• Creating user friendly web interface. • Preparing and development of documentation and report.

Table 1 Team members and divided roles

3. LITERATURE REVIEW

This section consists of the literature study on Search Engine System. Our project is looking forward to define all the possible services so that there is an intelligent search portal for required and general services for all the users as well as business owners. We have found various similar products that have already been developed in the global as well as local market, but our project stands among the rest.

3.1 Defining Search Engine

A search engine, also known as a search service, is a computer software that helps in the discovery of data stored on a computer system such as the World Wide Web, a corporate or proprietary network, or a personal computer. The search engine allows users to request information that meets particular criteria (usually content that contains a specific word or phrase) and returns a list of references that fulfill those requirements. Search engines use regularly updated indexes to operate quickly and efficiently.

Search Engine usually refers to a Web search engine, which searches for information on the public Web. However, there are other kinds of search engines like enterprise search engines, which search on intranets, personal search engines, which search individual personal computers, and mobile search engines. Some search engines also mine data available in newsgroups, large databases, or open directories. Unlike Web directories, which are maintained by human editors, search engines operate algorithmically. Most websites which call themselves search engines are actually front ends to search engines owned by other companies.

3.2 Types of Search Engines:

3.2.1 Crawler based:

Crawler-based search engines such as Google, compile their listings automatically. They "crawl" or "spider" the web, and people search through their listings. These listings are what make up the search engine's index or catalog. We can think of the index as a massive electronic filing cabinet containing a copy of every web page the spider finds. Because spiders scour the web on a regular basis, any changes we make to a web site may affect our search engine ranking. It is also important to remember that

it may take a while for a spidered page to be added to the index. Until that happens, it is not available to those searching with the search engine.

3.2.2 Directories based:

Directories such as Open Directory depend on human editors to compile their listings. Webmasters submit an address, title, and a brief description of their site, and then editors review the submission. Unless one signs up for a paid inclusion program, it may take months for the web site to be reviewed. Even then, there's no guarantee that the website will be accepted. After a website makes it into a directory however, it is generally very difficult to change its search engine ranking. So before we submit to a directory, we must spend some time working on our titles and descriptions. Moreover, the webpages have solid well-written content.

3.2.3 Mixed Results or Hybrid Search:

Some search engines offer both crawler-based results and human compiled listings. These hybrid search engines will typically favor one type of listing over the other however. Yahoo for example, usually displays human-powered listings. However, since it draws secondary results from Google, it may also display crawler-based results for more obscure queries. Many search engines today combine a spider engine with a directory service. The directory normally contains pages that have already been reviewed and accessed.

3.2.4 Pay-per-click:

More recently, search engines have been offering very cost effective programs to ensure that the ads appear when a visitor enters in one of the keywords. This new trend is to charge people on a Cost per Click model (CPC). The listings are comprised entirely of advertisers who have paid to be there. With services such as Yahoo SM, Google AdWords, and FindWhat, bids determine search engine ranking. To get top ranking, an advertiser just has to outbid the competition.

3.2.5 Meta crawlers or Meta Search Engines:

Metasearch Engines search, accumulate and screen the results of multiple Primary Search Engines (i.e. they are search engines that search search engines). Unlike search engines, metacrawlers don't crawl the web themselves to build listings. Instead, they

allow searches to be sent to several search engines all at once. The results are then blended together onto one page.

3.3 Existing Systems

3.3.1 Google Search

Google Search Engine is a proprietary search engine owned by American multinational technology company called Google. Invented by Sergey Brin and Larry Page, the company achieved better results for many searches with an algorithm called Page Rank. This iterative algorithm ranks web pages based on the number and Page rank of other web sites and pages that link there, on the premise that good or desirable pages are linked to more than others. It is the most popular search engine among the people.

3.3.2 Yahoo! Search

Yahoo! Search is another search service owned by Yahoo that sent queries to a searchable index of pages supplemented with its directory of websites, particularly stored on its Yahoo directory. It was the first popular search engine on the web, despite not being a true Web crawler search engine. But in 2003, it became its own web crawler-based search engine. Yahoo Search indexed and cached the common HTML page formats, as well as several of the more popular file-types, such as PDF, Excel spreadsheets, PowerPoint, Word documents, RSS/XML and plain text files. For some of these supported file-types, Yahoo Search provided cached links on their search results allowing for viewing of these file-types in standard HTML.

3.3.3 Microsoft Bing

Launched originally as MSN Search by Microsoft back in 1998, it consists of a search engine, index and web crawler. It has its own web crawler, the index of which is updated weekly and sometimes daily. Currently it also uses a Prediction engine that predicts some major political elections and reality show events.

3.3.4 Ask.com

Formerly known as Ask Jeeves, is an Internet search engine that is used to find relevant web pages. It is based on specific keywords and also has steps to verify the site on its own engine. As a business owner, we can benefit from the success of our website ranking. To submit our site to Ask.com, we must have a sitemap in XML format

uploaded to our Web server. We can then submit the sitemap URL directly to Ask.com using the Ask.com submission URL. It stands apart from the popular search engine Google in such a way that Google's Page Rank algorithm ranks its search results by popularity, whereas Ask has something it calls "Expert Rank". Essentially it is an automated search algorithm that orders the searched results using topic communities and that editorial functions create "Smart Answers". That means, the top results in searches are determined by expertise and not by popularity.

3.3.5 DuckDuckGo

DuckDuckGo is an internet search engine that emphasizes protecting user's privacy and avoiding filter bubble of personalized search results. It shows all users the same search results for a given search term by not profiling its users. The results are a compilation of over 400 sources, including all before mentioned search engines, as well as its own web crawler called DuckDuckBot. It also uses data from crowdsourced sites, including Wikipedia to populate knowledge boxes to the right of the results. It is a privacy focused search engine that does not store IP Addresses, does not log user information and uses cookies only when required. It also has its own web browser called DuckDuckGo Browser, which is also focused on privacy of the users.

3.4 What sets Innepal.net apart from other systems?

As we can see the existing system and search engines has their own system of displaying the results based on user queries. Our system "Innepal.net" would provide users with variety of options like geolocation-based site searching, keyword-based site searching and many more. Furthermore, our system uses the combination of Elasticsearch, Logstash, and Kibana, referred to as the "**Elastic Stack**" (formerly the "ELK stack") as a service. Logstash provides an input stream to Elasticsearch for storage and search, and Kibana accesses the data for visualizations such as dashboards. Elastic also provides "Beats" packages which can be configured to provide pre-made Kibana visualizations and dashboards about various database and application technologies.

3.5 Elasticsearch

Elasticsearch is a real-time distributed and open source full-text search and analytics engine. It is an Open Source developed in Java and used by many big organizations

around the world. It is basically a NoSQL Database, which is developed in Java programming language. It is a real-time, distributed, and analysis engine that is designed for storing logs. It is a highly scalable document storage engine. Similar to the MongoDB, it stores the data in document format. It enables the users to execute the advanced queries to perform detailed analysis and store all data centrally. The database is licensed under the Apache version 2.0 and based on Apache Lucene search engine. It is built-in RESTful APIs that help in fulfilling the request and responding to the request. It is an essential part of Elastic Stack or we can also say that it is a heart of Elastic Stack.

It is used in Single Page Application (SPA) projects. Many large organizations across the world use it. It supports full-text search that is completely document-based instead of schemas and tables. There are some more other search-based engines available, but they all are based on tables and schemas.

A typical Elasticsearch document looks like this:

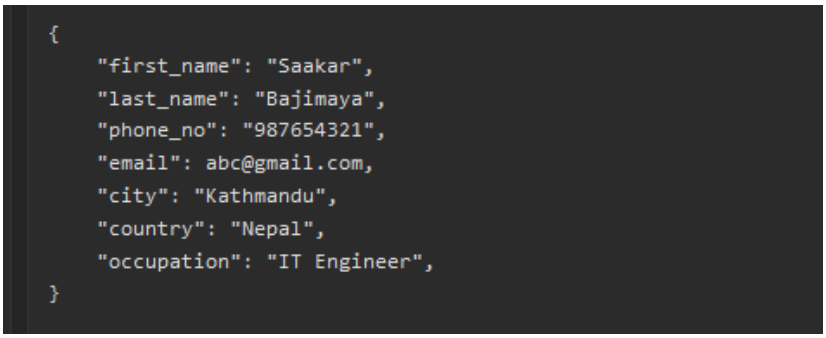

Figure 2 Snapshot of Elasticsearch Document

There are some other reasons for using Elasticsearch are as follows:

- It allows us to perform and combine various types of searches, like structured as well as unstructured. It also helps in working upon the data, which is based on geography as well as on matrix.
- We can retrieve the result from the data which we import in any way we want. It is all based on structured query sets.
- It allows the users to ask the query anyway they want.
- It provides aggregations that help us to explore trends and patterns in our data.

- It takes care of both query and analysis on data.
- Its database helps to complete the search query based on the previous searches automatically.
- Auto-completing a search box based on partially typed words based on previously issued searches while accounting for misspellings.
- Storing a large quantity of semi-structured (JSON) data in a distributed fashion, with a specified level of redundancy across a cluster of machines.

3.6 Logstash

Logstash is a light-weight, open-source, server-side data processing pipeline that allows us to collect data from a variety of sources, transform it on the fly, and send it to wer desired destination. It is most often used as a data pipeline for Elasticsearch, an open-source analytics and search engine. Because of its tight integration with Elasticsearch, powerful log processing capabilities, and over 200 pre-built open-source plugins that can help we easily index our data, Logstash is a popular choice for loading data into Elasticsearch.

The following are some of the benefits of using Logstash:

- **Easily load unstructured data:** It allows us to easily ingest unstructured data from a variety of data sources including system logs, website logs, and application server logs.
- **Prebuilt filters:** It allows pre-built filters, so we can readily transform common data types, index them in Elasticsearch, and start querying without having to build custom data transformation pipelines.
- **Flexible Plugin Architecture:** With already available 200 plugins on GitHub, it is likely that someone has already built the plugin that we need in our data pipeline. If not, we can also create one for ourselves.

3.7 Kibana

Kibana is a data visualization dashboard used on Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data. It is used for log and time-series analytics, application monitoring, and operational

intelligence use cases. It offers powerful and easy-to-use features such as histograms, line graphs, pie charts, heat maps, and built-in geospatial support. Also, it provides tight integration with Elasticsearch, a popular analytics and search engine, which makes Kibana the default choice for visualizing data stored in Elasticsearch.

The following are the key benefits of using Kibana:

- **Interactive Charts:** Kibana offers intuitive charts and reports that we can use to interactively navigate through large amounts of log data. We can dynamically drag time windows, zoom in and out of specific data subsets, and drill down on reports to extract actionable insights from wer data.
- **Mapping Support:** Kibana comes with powerful geospatial capabilities so we can seamlessly layer in geographical information on top of wer data and visualize results on maps.
- **Pre- built Aggregations and Filters:** Using Kibana's pre-built aggregations and filters, we can run a variety of analytics like histograms, top-N queries, and trends with just a few clicks.
- **Easily Accessible Dashboards:** We can easily set up dashboards and reports and share them with others. All we need is a browser to view and explore the data.

3.8 Lucene

Apache Lucene is a high performance, full-featured Information Retrieval library, written in Java. Elasticsearch uses Lucene internally to build its state of the art distributed search and analytics capabilities. Now, we've a scenario to implement the search feature to an application. We've tackled getting the data indexed, but now it's time to expose the full-text searching to the end users. It's hard to imagine that adding search could be any simpler than it is with Lucene. Obtaining search results requires only a few lines of code.

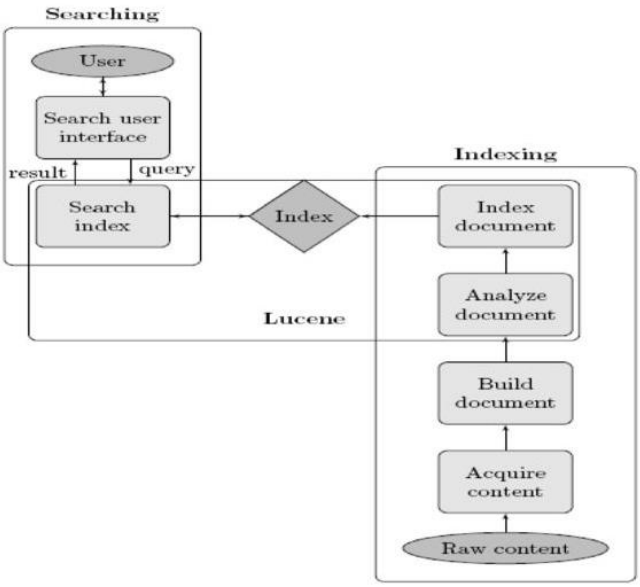

Figure 3 Typical components of a search application

Lucene provides easy and highly efficient access to those search results, too, freeing us to focus on our application logic and User Interface around those results. When we search with Lucene, we'll have a choice of either programmatically constructing our query or using Lucene's *QueryParser* to translate text entered by the user into the equivalent Query.

The first approach gives us ultimate power, in that our application can expose whatever interface it wants, and our logic translates interactions from that interface into a Query. But the second approach is easy to use, and offers a standard search syntax that all users are familiar with.

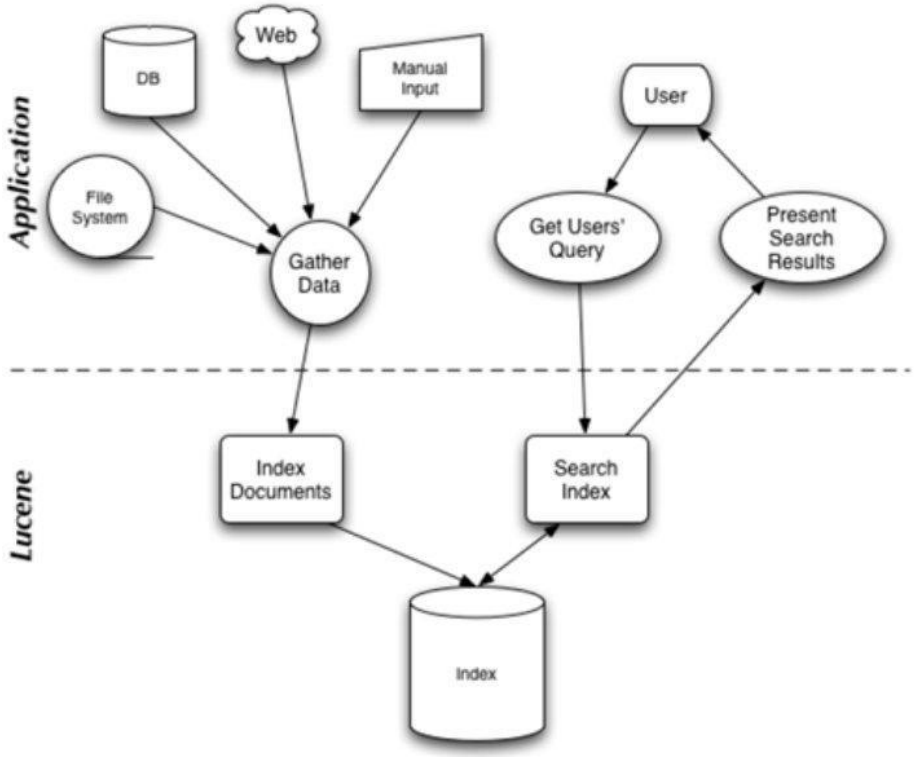

Figure 4 Lucene Architecture Implementation

For instance, let's take the simplest search of all: Searching for all documents that contain a single term. Example: Searching for a specific term.

IndexSearcher is the central class used to search for documents in an index. It has several overloaded search methods. We can search for a specific term using the most commonly used search method. A term is a string value that's paired with its containing field name- in our example, subject. Lucene provides several built-in *Querytypes*, *TermQuerybeing* the most basic. Lucene's search methods require a Query object. Parsing a query expression is the act of turning a user-entered textual query such as "mock OR joint" into an appropriate Query object instance; in this case, the Query object would be an instance of Boolean-Query with two optional clauses, one for each term.

3.9 Indexing

To search large amount of text quickly, one must first index that text and convert it into a format that will let one search it rapidly, eliminating the slow and sequential scanning process. This conversion process is call Indexing and it output is called an Index. Indexing is the initial part of all search applications. Its goal is to process the original

data into a highly efficient cross-reference lookup in order to facilitate rapid searching. The job is simple when the content is already textual in nature and its location is known.

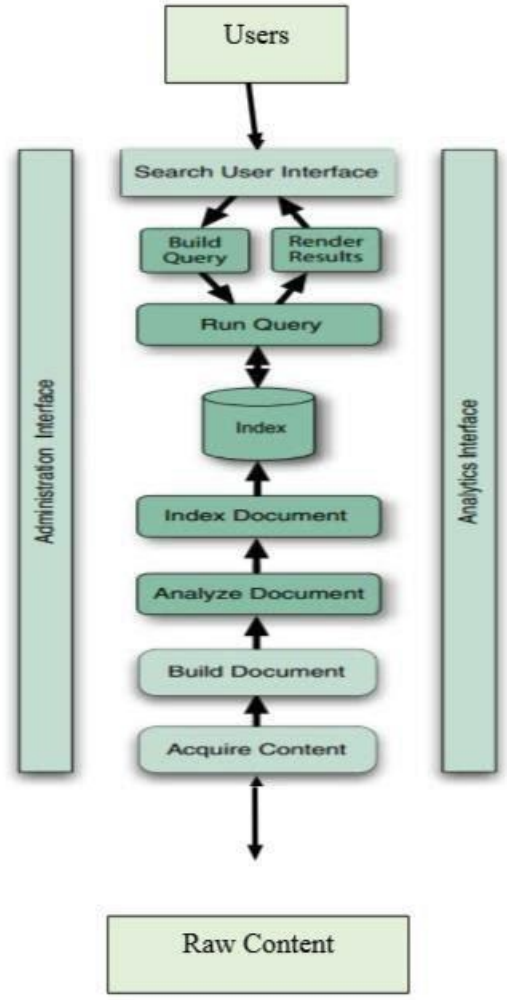

Figure 5 Components of a Search Application

The following steps are involved in Indexing:

- **Acquiring the content:** This process gathers and scopes the content that needs to be indexed. This process involves using a crawler or spider, gathers and scopes the content that needs to be indexed. It may be trivial, for example, if we're indexing a set of XML files that resides in a specific directory in the file system or if all our content resides in a well-organized database. Alternatively, it may be horribly complex and messy if the content is scattered in all sorts of places. It needs to be incremental if the content set is large.

- **Build Documents:** Once we have the raw content that needs to be indexed, we must translate the content into the units (usually called documents) used by the search engine. The document typically consists of several separately named fields with values, such as title, body, abstract, author, and URL. A common part of building the document is to inject boosts to individual documents and fields that are deemed more or less important.
- **Document Analysis:** The textual fields in a document cannot be indexed directly. Rather, the text has to be broken into a series of individual atomic elements called tokens. This happens during the document analysis step. Each token corresponds roughly to a word in the language, and the analyzer determines how the textual fields in the document are divided into a series of tokens.
- **Index the document:** The final step is to index the document. During the indexing step, the document is added to the index.

3.9.1 Creating an index (IndexWriter Class)

The first step in implementing full-text searching with Lucene is to build an index. To create an index, the first thing that need to do is to create an IndexWriter object. The IndexWriter object is used to create the index and to add new index entries (i.e. Documents) to this index. We can create an IndexWriter as follows:

IndexWriter indexWriter = new IndexWriter("index-directory", new StandardAnalyzer(), true);

3.9.2 Parsing the Documents (Analyzer Class)

The job of Analyzer is to "parse" each field of our data into an indexable "tokens" or keywords. Several types of analyzers are provided out of the box. Some of them are: *StandardAnalyzer*, which is a sophisticated general-purpose analyzer, *WhitespaceAnalyzer*, which is a very simple analyzer that just separates tokens using white space, *StopAnalyzer* removes common English words that are not usually useful for indexing, *SnowballAnalyzer* is an interesting experimental analyzer that works on word roots (a search on rain should also return entries with raining, rained, and so on).

3.9.3 Adding a Document/object to Index (Document Class)

To index an object, we use the Lucene Document class, to which we add the fields that we want indexed. It can be created as follows:

```
Document doc = new Document();

doc.add(new Field("description", hotel.getDescription(), Field.Store.YES, Field.Index.TOKENIZED));
```

3.10 Searching

Searching or Querying is the process of looking up words in an index to find documents where they appear. The components for searching the index are as follows:

1. **Search User Interface:** The search user interface is what users actually sees, in the web browser, desktop application, or mobile device, when they interact with the search application. It is the most important part of any search application.
2. **Build Query:** When we manage to entice a user to use the search application, s/he issues a search request, often as the result of an HTML form or Ajax request submitted by a browser to the server. We must then translate the request into the search engine's Query object. This is called build query step. The user query can be simple or complex.
3. **Run Query/Search Query:** Search Query is the process of consulting the search index and retrieving the documents matching the Query, sorted in the requested sort order. This component covers the complex inner workings of the search engine.
4. **Render Results:** Once we have the raw set of documents that match the query, sorted in the right order, we then render them to the user in an intuitive, consumable manner. The UI should also offer a clear path for follow-on searches or actions, such as clicking to the next page, refining the search, or finding documents similar to one of the matches, so that the user never hits a dead end.

3.11 Analyzers

Analyzers are (generally) composed of a single Tokenizer and zero or more Token Filters. A set of Char Filters can be associated with an analyzer to process the characters prior to other analysis steps. The analysis module allows one to register Token Filters, Tokenizers and Analyzers under logical names that can then be referenced either in mapping definitions or in certain APIs. The Analysis module automatically registers (if not explicitly defined) built in analyzers, token filters, and tokenizers. The types of Analyzers in general are broken down into a Tokenizer with zero or more Token Filter applied to it. The analysis module allows one to register Token Filters, Tokenizers and Analyzers under logical names which can then be referenced either in mapping definitions or in certain APIs.

Following are a list of analyzer types:

- Char Filter: Char filters allow one to filter out the stream of text before it gets tokenized (used within an Analyzer).
- Tokenizer: Tokenizers act as the first stage of the analysis process (used within an Analyzer).
- Token Filter: Token filters act as additional stages of the analysis process (used within an Analyzer).
- Default Analyzers: An analyzer is registered under a logical name. It can then be referenced from mapping definitions or certain APIs. When none are defined, defaults are used. There is an option to define which analyzers will be used by default when none can be derived. The default logical name allows one to configure an analyzer that will be used both for indexing and for searching APIs. The default index logical name can be used to configure a default analyzer that will be used just when indexing, and the default search can be used to configure a default analyzer that will be used just when searching.

In Lucene, analyzers can also be classified as:

- Whitespace Analyzer, as the name implies, splits text into tokens on whitespace characters and makes no other effort to normalize the tokens. It doesn't lower-case each token.

- Simple Analyzer first splits tokens at non-letter characters, then lowercases each token. Also this analyzer quietly discards numeric characters but keeps all other characters.
- Stop Analyzer is the same as Simple Analyzer, except it removes common words. By default, it removes common words specific to the English language (the, a, etc.), though we can pass in our own set.
- Standard Analyzer is Lucene's most sophisticated core analyzer. It has quite a bit of logic to identify certain kinds of tokens, such as company names, email addresses, and hostnames. It also lowercases each token and removes stop words and punctuation.

4. METHODOLOGY

4.1 Software Development Lifecycle

The framework we followed in developing the project is in agile model, in which the whole requirement is divided into various builds. AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development lifecycle of the project. Both development and testing activities are concurrent.

The agile software development emphasizes on four core values.

- Individual and team interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

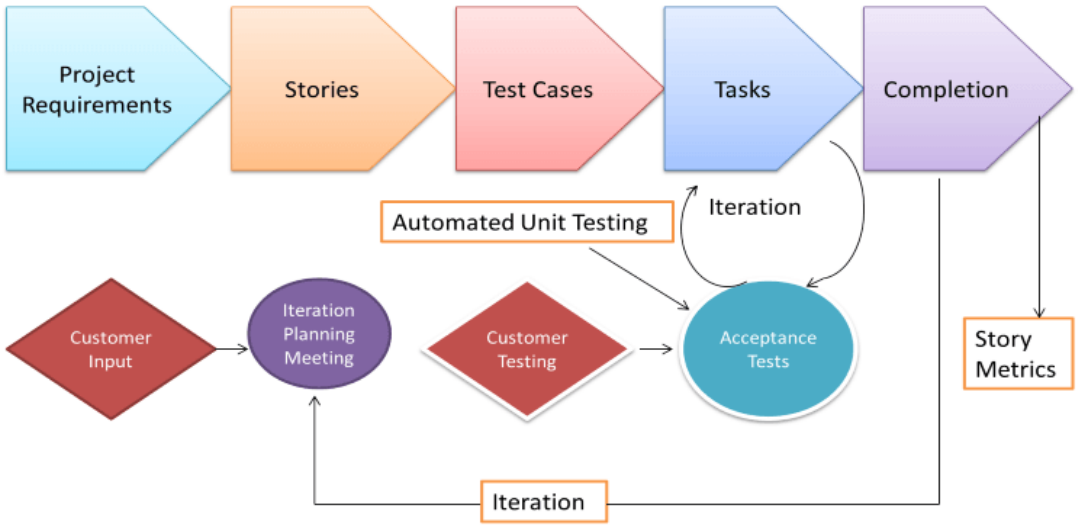

Figure 6 Agile Model

4.2 Object Oriented Analysis

Analysis is an important part of system development. Object Oriented Analysis adds more object oriented features to the standard analysis procedure. It provides flexibility, encapsulation, abstraction and most importantly reusability. Various Unified Modeling Language(UML) diagrams are used for carrying out object oriented analysis.

4.2.1 Use case Diagrams

Use Case Diagram is the basic analysis diagram. It is the first analysis diagram.it gives interaction of the system with entities outside the system. These entities are called actors. Actors are the users of the system or other systems who give input to the system and take output from the system. The various scenarios in the system are called use cases. They are denoted by oval. The complete system has been explained using a basic use case diagram which shows the interaction between the main actors and a set of use case scenarios. The detailed use case diagrams following the basic use case diagram gives the detailed interaction of each actor with the system.

The main actors are:

- **Users**

  They submit a query in the system which is processed by the system. They also obtain results from the system which is the list of websites containing the required animation.
- **Website Developer**

  They are required to register his website with the search engine in order to enable the crawler to locate the site.
- **Administrator (webmaster)**

  They specify certain rules with the website that are taken into consideration by the crawler while crawling a particular website. These information includes various webpages access constraints.

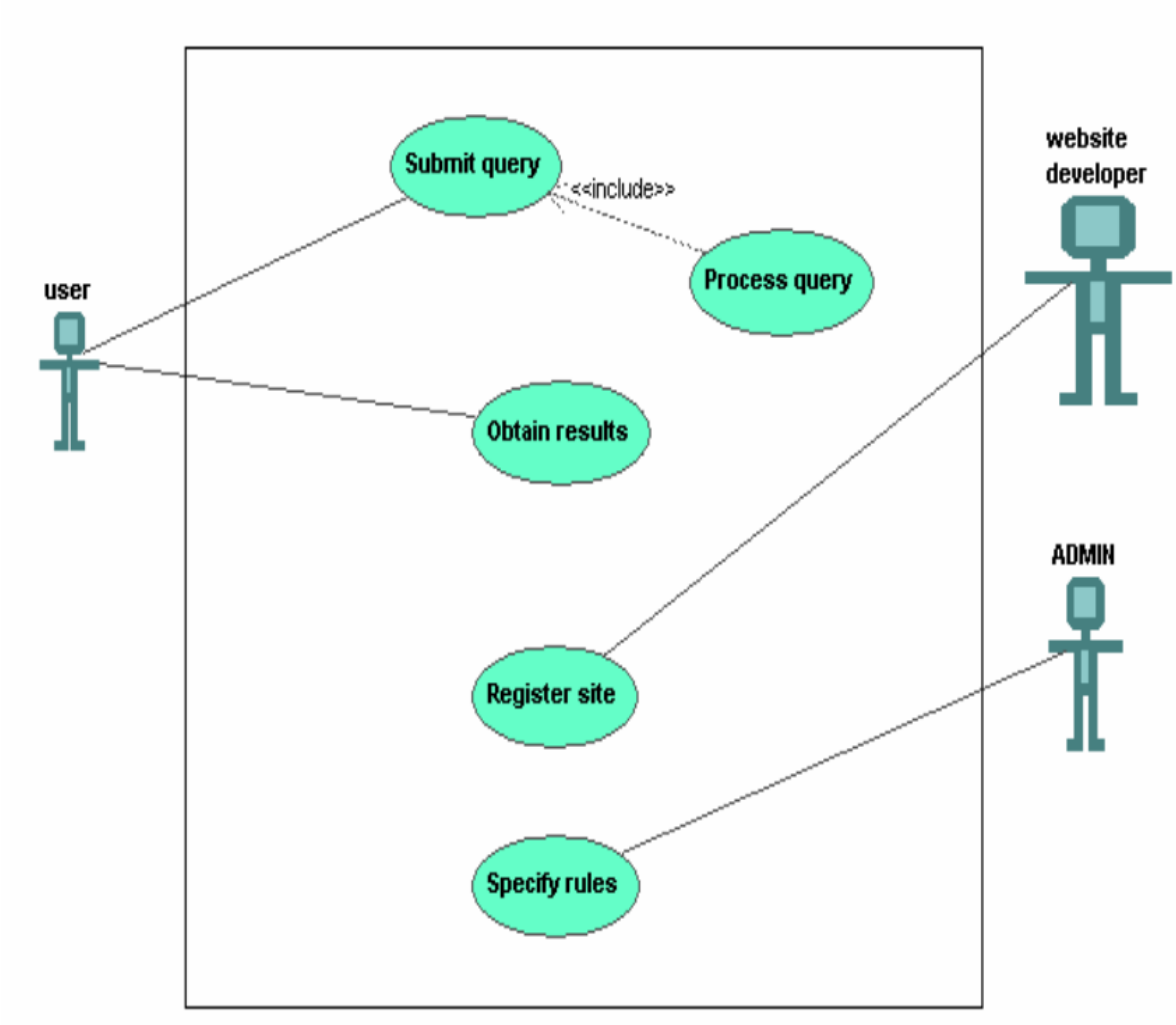

Figure 7 Basic Use case diagram

User submits query in the system which is processed by the system. The user also obtains results from the system which is the list of websites containing the required information.

The website developer is required to register his website with the search engine in order to enable the crawler to locate the site.

The website administrator specifies certain rules with the website with are taken into consideration by the crawler while crawling a particular website. These information includes various webpage access constraints.

4.2.1.1 Use Case view for User:

The following is the detailed use case view for showing the user's interaction with the system. User is the actor which submits query in the system to search information for a particular topic. User can choose the mode of submission. The query can be processed

either in normal mode or in advanced mode. Advanced mode allows using proximity search as well as specific file search. Submit query use case also includes validation. Validation checks for spellings and suggestions regarding similar queries. The user is provided with the results after processing the query.

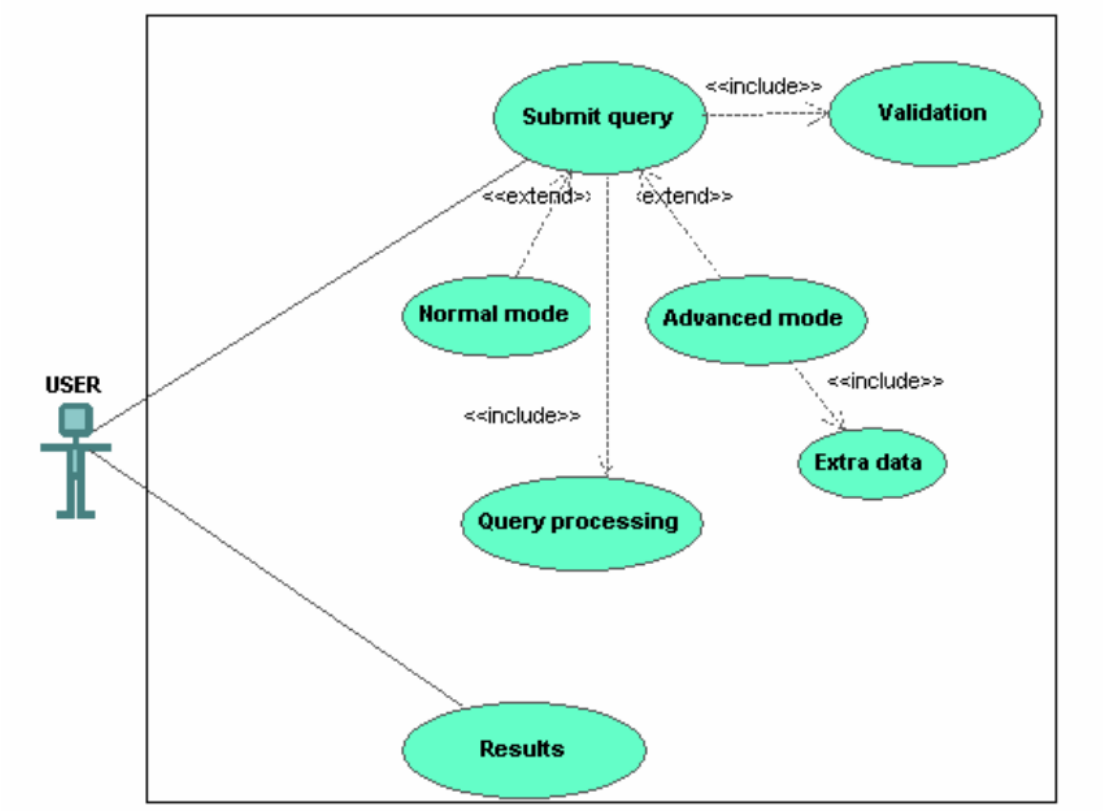

Figure 8 User use case view

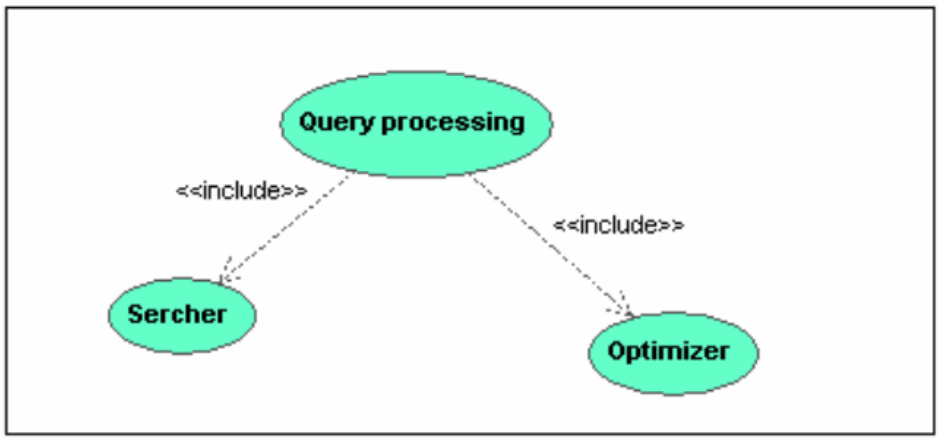

Figure 9 User use case view for Query Processing

4.2.1.2 Use case view for Website Developer:

The different aspect of the system is with respect to the Website Developer. The website developer has to register his website with the search engine so that it is crawled and shows up in the result of the query. The search engine has a special tool called extractor which adds the URL to the URL server. One of the methods to add the URL is through registration by the website developer. The crawler then accepts a URL from the server. It checks for the specified rules for the website and then scans and downloads the page. The system also includes a deadlink checker program which periodically checks for deadlinks in a webpage. If a deadlink is observed constantly then the particular website developer is informed.

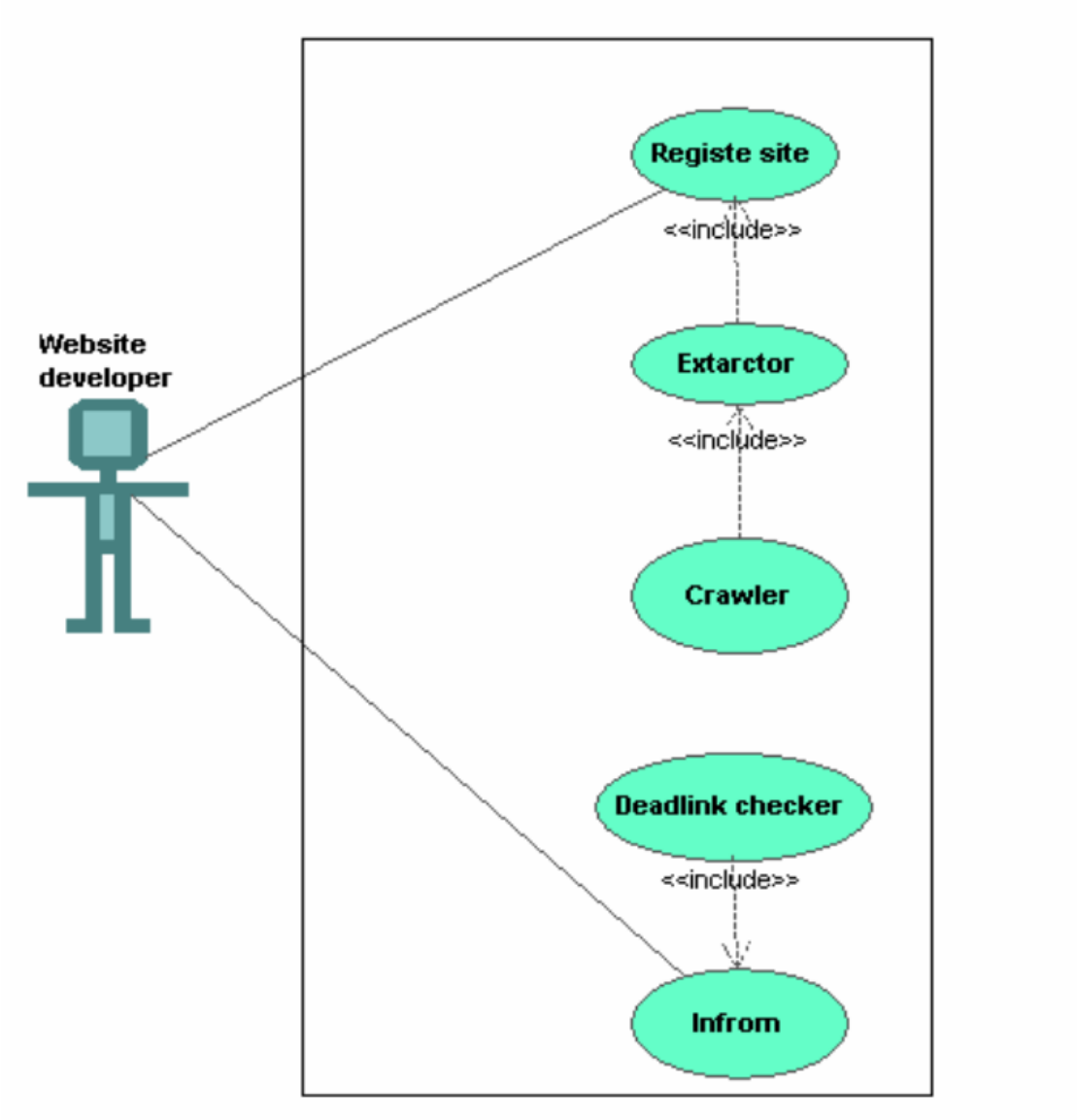

Figure 10 Use case view for Website Developer

4.2.1.3 Use Case view for Administrator (webmaster)

The web master or administrator is the actor which specifies various rules for a particular website which includes access information. One such file is **"robot.txt"**. This file is first scanned by the crawler and various permissions are checked before downloading a website.

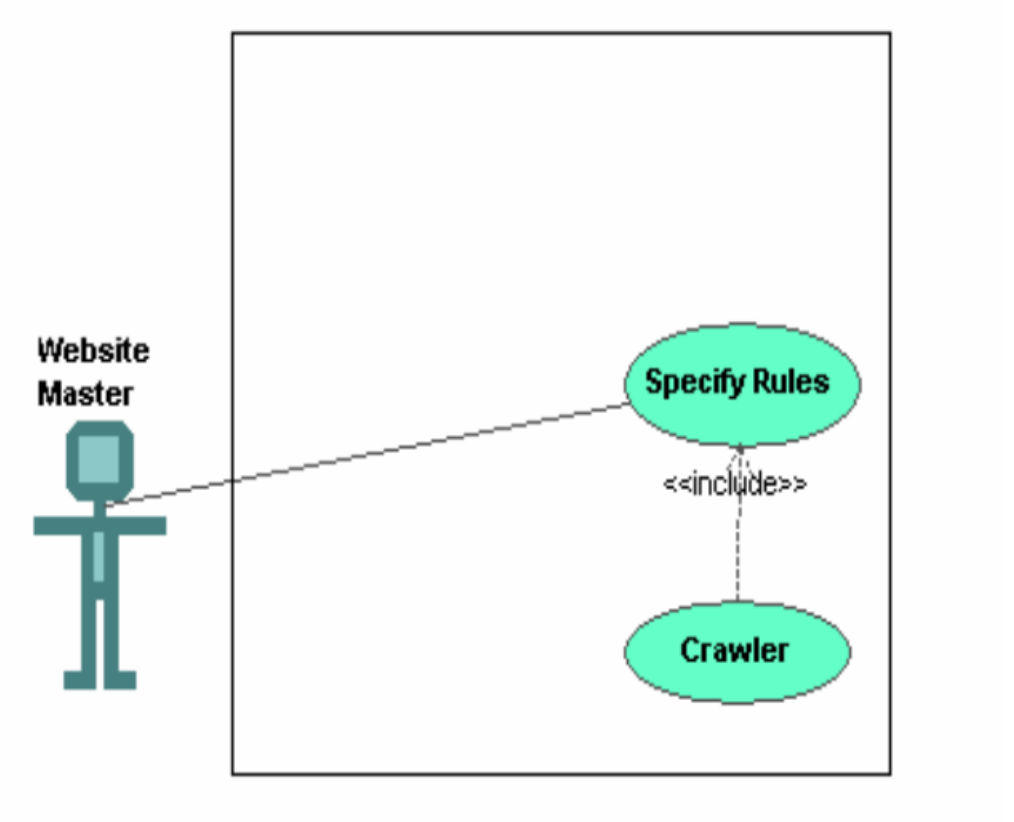

Figure 11 Use Case view for Administrator (webmaster)

4.2 System Design

In order to solve the problems in the existing system, the proposed system uses the latest algorithms such as search engine optimization technique, page ranking, indexing and web crawling. This will provide the best search results to the users. The unique and distinct search result is displayed by the proposed search engine to the user's query.

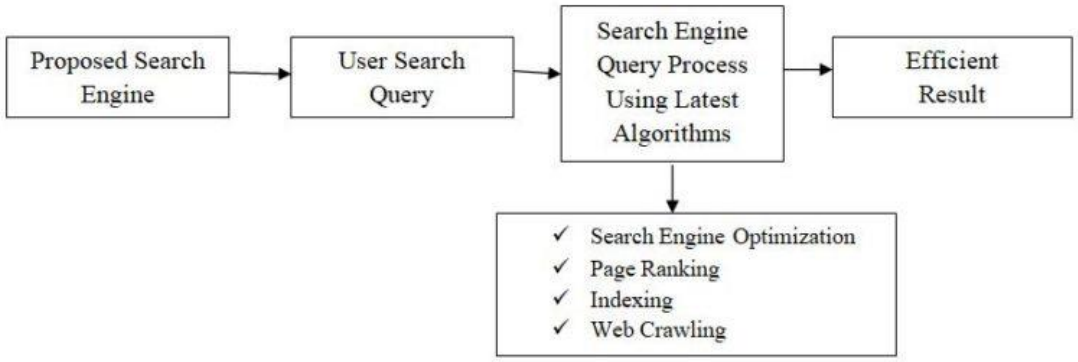

Figure 12 Proposed System Architecture

A number of web links are indexed, whenever a keyword is searched by the user using this Search Engine application. The analysis is performed for the data in the indexed links. This task is achieved by page rank algorithm. Finally, the web page containing the top match to the key word is showed to the user in the first link of the result page.

The following modules are used:

1. **Search:** User search for the particular query in the search engine. This user query is saved for further processing.

2. **Indexing:** After a user searches for the particular keyword, database matching those keywords is identified. Then number of web links relevant to this query are indexed.

3. **Analyzing:** Then to retrieve the best possible match, this search engine application uses page ranking, web crawling and search engine optimization.

4. **Result Retrieval:** After analyzing, the best matching result is retrieved and it is displayed in the user interface window to the user.

This would provide us with the following advantages:

- It provides the best search result for the user by using the latest tools, algorithms and technologies.
- It uses the features such as search engine optimization, web crawling, indexing, and page ranking etc. to extract the exact and fast result.
- This can be utilized from any geographical location, as long as there is availability of the internet service.

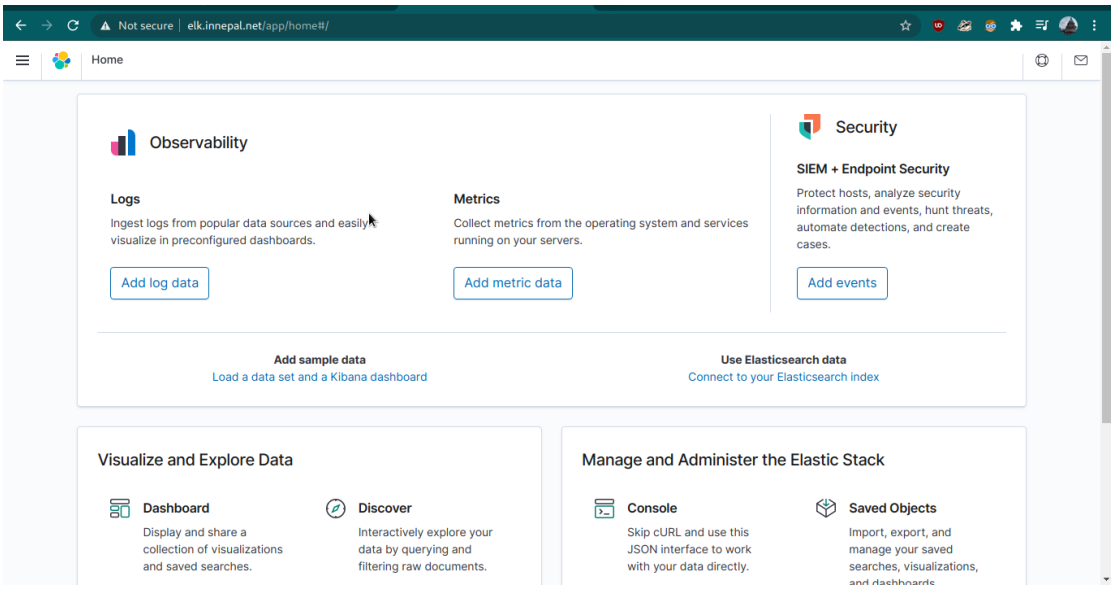

Figure 13 ELK Hosted at innepal.net

4.3 Technologies Used

The following are the technologies used in our project:

- HTML5, CSS, JavaScript, React JS for creating lawet and user interface for our site.
- ELK (combined Elasticsearch, Logstash and Kibana).
- Django REST Framework for building up our rest API.
- NodeJS for backend development of our system.
- Python and JavaScript as our main programming language.
- Docker Virtualization for server deployment.

4.4 Tools Used

The following are the tools that we used in our system:

Tools	Purpose
Atom	IDE Code Editor
Microsoft Word and Microsoft Powerpoint	To create our proposal report and slides for our demo presentation.
Draw.io	To draw our designs and diagrams.
Mozilla Firefox and Google Chrome	For testing our site and to launch our hosted site.
Docker Container	To deploy our server.
PIP and NPM	Package installer for Python and NodeJS.
Git and Github	For version control and team collaboration.

Table 2 Tools used in our system

4.5 Task and Time Schedule

Task/Time	**Oct2020-Dec2020**	**Feb2021-Mar2021**	**Mar2021-April2021**	**April2021-May2021**	**July 2021**
Research					
Data Collection					
Search implementation					
Optimization and Production					
Deployment					

Table 3 Gantt Chart for task and time schedule

5. CONCLUSION AND FUTURE EXTENSION

"Innepal.net" is a site search portal that provides search results of requested queries based on the user's search term. The user enters desired search term into the search field and our search engine then looks into its index for relevant websites and displays them in the form of list. It aims to be a centralized site search engine system by providing a viable platform for both businesses and customers under a single entity. The gist of our project is to provide a convenient way for business owners as well are general users to have a better search experience based on their queries.

Our system is not a replacement to current available technology but a complementary service to those technologies. We had first analyzed the current systems and listed out its problem statement and design our system to conquer those problems. After designing the system, we created prototypes and conducted tests. We had used agile method to develop the overall system, creating and testing individual model and taking the feedbacks.

With an intent to extend our project further, we are looking forward to implement the better version of the following features into the system more precisely.

- Deploying our search portal into the Nepali market to the general public as a centralized and localized search engine.
- Activities such as different digital marketing strategies like Search Engine Optimization (SEO), Page Ranking and others may be implemented as well.
- As a means of promoting local businesses, different advertisement campaigns could be conducted through our portal.
- Implement a data collection and analysis framework to study the information based on user's search queries.
- Data Visualization may be implemented as well.

6. BIBLIOGRAPHY/REFERENCES

[1] Artyom Churilin, "Choosing an open-source log management system for small business," Master's Thesis, Faculty of Information Technology, Tallinn University of Technology, Tallinn, Estonia.

[2] Christopher (2015, April 15). Visualizing data with Elasticsearch, Logstash and Kibana[Online]. Available: http://blog.webkid.io/visualizedatasets-with-elk/

[3] Prakash, T., Kakkar, M., & Patel, K. (2016). Geo-identification of web users through logs using ELK stack. 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence).

[4] L.K. Joshila Grace, V. Maheswari, and Dhinaharan Nagamalai" Web Log Data Analysis and Mining" in Proc CCSIT-2011, Springer CCIS, Vol 133, pp 459-469, 2011.

[5] Anders Aarvik (2014, April 04). A bit on Elasticsearch + Logstash + Kibana (The ELK stack) [Online]. Available: http://aarvik.dk/a-bit-onelasticsearch-logstash-kibana-the-elk-stack/

[6] Christopher (2015, April 15). Visualizing data with Elasticsearch, Logstash and Kibana [Online]. Available: http://blog.webkid.io/visualizedatasets-with-elk/

[7] Building custom search architecture for wer site using ELK: Elasticsearch (2019, Oct 10)

[8] [Online]Available: https://yes-soft.de/building-customised-search-architecture-for-wer-site-using-elk-elasticsearch-ishtar/