**TAXONOMY AND RANKING OF REUSABLE COMPONENTS FOR EFFECTIVE RETRIEVAL**

****

**Submitted to**

**the faculty of Engineering and Technology of**

**Kakatiya University, Warangal**

**in partial fulfilment of the requirements**

**to award**

**Bachelor of Technology**

**in**

**Computer Science & Engineering**

BY

**S.LAVANYA [12016T0952]**

**A.AKSHAY [12016T0957]**

**DEEKSHA.G [12016T0941]**

**A.KOUSHIK [12016T0954]**

**Under the Guidance of**

**Dr. P. NIRANJAN**

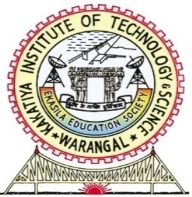
**Professor & Head, CSE**

**Department of Computer Science & Engineering**

**Kakatiya Institute of Technology & Science,**

**WARANGAL (TS)**

**2015 - 2016**

** KAKATIYA INSTITUTE OF TECHNOLOGY & SCIENCE**

**.** (Sponsored by Ekashila Education Society, Warangal)

(Affiliated to Kakatiya University, Warangal)(Approved by AICTE, New Delhi)

**Opp: Yerragattu Hillock, Vill: Bheemaram, Mandal: Hasanparthy, Warangal-506015, TS.**

**C E R T I F I C A T E**

This is to certify that **A.AKSHAY REDDY (12016T0957)** of IV/IV B.Tech. Computer Science and Engineering has satisfactorily completed the dissertation work entitled “**TAXONOMY AND RANKING OF REUSABLE COMPONENTS FOR EFFECTIVE RETRIEVAL”** in the partial fulfilment of the requirements of B.Tech. degree during the academic year 2015-16.

**Supervisor** **Head of the Department**

Dr. P. Niranjan Dr. P. Niranjan

**External Examiner**

**ACKNOWLEDGEMENT**

We express our deep sense of gratitude to our project guide, **Dr. P. Niranjan**, **Head Department of Computer Science and Engineering** under whose guidance and supervision this work has been accomplished. His keen interest, constant inspiration and constructive criticism have been of great help to us throughout the course of this project work.

We are extremely grateful to **Sri.** **S.Nagaraju**, Associate Professor and Project coordinator, **Department of Computer Science and Engineering** for his consistent support in conduction of our project reviews which paved the way for successful completion of the work.

Our sincere thanks to **Dr. P. Niranjan**, **Head, Department of Computer** **Science and Engineering** for providing all the facilities to carry out the project work and for his constant encouragement throughout this work.

We wish to thank **Dr. K. Guru Raj,** **Principal**, and **Dr. Y. Manohar, Director**, Kakatiya Institute of Technology & Science for providing all the required facilities.

We thank all **Faculty** members of the Department of Computer Science and Engineering for sharing invaluable knowledge with us. We extend our thanks to the **Technical staff** of the department for the valuable suggestions to technical problems.

We would like to place on record our heart full thanks to all those who have directly or indirectly helped us in the successful completion of this project work.

S.LAVANYA 12016T0952

A. AKSHAY 12016T0957

DEEKSHA.G 12016T0941

A .KOUSHIK 12016T0954

**TABLE OF CONTENTS**

**Topics**  **Page No.**

1. INTRODUCTION 1

1.1 Introduction 2

1.2 Background and Motivation of the Topic 3

2. LITERATURE SURVEY 6

2.1 Existing Software Component Classification Techniques 7

2.2 Advantages and Disadvantages of Existing System 8

2.3 Motivation for the Proposed Approach 9

2.4 General Architecture for Ranking Process 10

2.5 Description of the Proposed Approach 11

2.6 Proposed Rank Algorithm 12

2.7 Modules in the Proposed Approach 13

2.8 Dataset used and its Description 13

3. DESIGN 15

3.1 Software Requirement Specifications 16

3.3 Project Flowchart 21

3.4 Dataflow Diagrams 22

3.5 UML (Unified Modeling Language) 25

4. IMPLEMENTATION 35

4.1 Implementation 36

5. TESTING 38

6. RESULTS 42

6.1 Results 43

6.2 Code 53

7. CONCLUSION AND FUTURE SCOPE 67

8. BIBILIOGRAPHY 69

**ABSTRACT**

Software Reuse is the process of implementing or updating software systems using existing software reusable components. To improve the practicality of software reuse one has to have knowledge of its location and understanding of the software reusable component. Reusable components stored in a repository are useful in developing early prototypes with better quality. One of the most fundamental problems in software reusability is locating and retrieving software components from a large repository. To reuse a software component, you first have to find it. Retrieval of component should be less time consuming and efficient. Hence the software reuse repositories must be developed with efficient representation of software components which facilitate the developer to locate and retrieve the components very easily as per the requirements. Software reuse greatly reduces the effort, development time and costs of the software. It focuses on various challenges involved in software reusable component classification, retrieval methods and ranking algorithms. A web based software tool is proposed with a new integrated classification scheme, to classify and retrieve as well as build a comprehensive software reuse repository.

**LIST OF FIGURES**

Figure 2.1 System Architecture 10

Figure 3.1 Project Flow Chart 21

Figure 3.2 Level-0 Data Flow Diagram 23

Figure 3.3 Level-1 Data Flow Diagram 24

Figure 3.4 Use Case Diagram of the System 27

Figure 3.5 Class Diagram of the System 29

Figure 3.6 Sequence Diagram for the User 30

Figure 3.7 Sequence Diagram for the Admin 31

Figure 3.8 Collaboration Diagram for the User 32

Figure 3.9 Collaboration Diagram for the Admin 33

Figure 3.10 Activity Diagram 34

Figure 6.1 Login Page for the System 43

Figure 6.2 Administrator page for the System 44

Figure 6.3 User page for the System 45

Figure 6.4 Add new Component page for the System 46

Figure 6.5 Edit Component page for the System 47

Figure 6.6 Delete Component page for the System 48

Figure 6.7 View Database page for the System 49

Figure 6.8 User Search page for the System 50

Figure 6.9 Results Generated from the Search Page 51

Figure 6.10 User downloaded files page 52

**LIST OF TABLES**

Table1: users 13

Table2: attributes 14

Table3: att\_calculate 14

Table4: ratinglater 14

**CHAPTER-1**

**INTRODUCTION**

* 1. **Introduction**

Software Engineering is an application of systematic and disciplined approach to the design, development, operation and maintenance of software. Component based software engineering focuses on the development of applications based on software components, so that applications are easy to maintain and extend. A component is an existing piece of software written with the purpose of reuse that can be deployed independently with little or no modification. A software component can be code, procedure, module, subsystem, requirements, design or even documentation. Software reuse has been practised since the programming began. The software reuse plays an important role because developing software from scratch is a tedious, time consuming and a costly process. The major areas which are to be focussed in software reuse are good design of repository, classification, clustering, indexing, searching and retrieval mechanisms. It also requires sufficient and proper documentation of components, a flexible means of combining components, and a means of adapting components to specific needs. There are many different techniques to locate and retrieve components efficiently, e.g., keyword search, free text based retrieval, enumerated, faceted, integrated approach etc. The proposed system uses attribute based approach by which components are retrieved using ranking algorithms. This approach improves the chance of retrieving appropriate component from the repository.

**1.2 Background and Motivation of the Topic**

**Component:**

Component is a well-defined unit of software that has a published interface and can be used in conjunction with other component to form large units. Reuse deals with the ability to combine separate independent software components to form a larger unit of software. A software component can be any product of the software development process -- a unit of code, a design specification, a test case, etc. Once the developer is satisfied with the component he has retrieved from library, then it is added to current project under development.

**Software Reuse:**

Reusability is a measure of the ease with which one can use those previous concepts or objects in the new situations. Reuse is the use of previously acquired concepts or objects in a new situation, it involves encoding development information at different levels of abstraction, storing this representation for future reference, matching of new and old situations, duplication of already developed objects and actions, and their adaptation to suit new requirements. Source code is most commonly reused; thus many people misconceive software reuse as the reuse of source code alone. Source code and design reuse have become popular with (object-oriented) class libraries, applications frameworks and design patterns. Software reuse and software components have a major influence on the structure of the software systems as well as on the way we build them.

**Component Based Software Engineering (CBSE):**

The goal of the component-based software engineering is to increase the productivity, quality and time-to market in software development. One important paradigm is to build software systems from standard components rather than developing each time. This requires thinking in terms of system families rather than single systems. CBSE uses software Engineering principles for designing and construction of software systems same as object oriented programming. It focuses on reusing and adapting existing components as opposed to just coding in a particular style. CBSE encourages the composition of software systems, as opposed to programming them.

Several technologies were developed to build the components and assembling applications from sets of those components. Object oriented and component technologies are examples-especially standards such as CORBA. The business and organizational within which applications are developed, deployed and maintained are changed. There is an increasing need to communicate with legacy systems, as well as constantly updating current systems. This need for new functionality in current applications requires technology that will allow easy additions. CBSE facilitates software systems to be more easily assembled, and less costly to build. The software systems built using CBSE are not only simple but also adaptable and updateable. CBSE allows use of predictable architectural patterns and standard software architecture leading to a higher quality end result.

**The Benefits of Software Reuse:**

As discussed previously, software reuse has a positive impact on software quality, as well as on costs and productivity. Software reuse results in improvements in quality, productivity and reliability.

**Quality:** Error fixes accumulate from reuse to reuse. This yields higher quality for reused components that is developed and reused only once.

**Productivity:** A productivity gain is achieved due to less code that has developed. This results in less testing efforts and also saves analyses and design labour, yielding overall savings in cost.

**Reliability:** Using well-tested components increase the reliability of a software system. Moreover, the use of a component in several systems increases the chance of errors being detected and strengthens confidence in that component.

**Redundant Work and Developing Time:**  Developing every system from scratch means redundant developments of many parts like requirement specifications, use cases, architecture, etc. This can be avoided when these parts are available as reusable components and can be shared, resulting in less development and less associated time and costs.

**Documentation:** Although documentation is very important for the maintenance of a system, it is often neglected. Reusing software components reduces the amount of documentation to be written.

**Maintenance Costs:** Fewer defects can be expected when proven quality components have been used and less maintainability of the system.

**Software Reuse Industry Examples:**

* Reuse benefits have been reported in various industrial settings.
* An empirical study from a NASA software production environment has shown that modules used without modifications (revision) had less interaction with other modules, simpler interfaces, less interaction with human users, and higher ratios of commentary compared to newly developed or modified (revised) modules. In this study 25 software projects (ranging from 3,000 to 112,000 source code lines) were considered. An average of 32% of software had been reused or modified from previous systems.
* At Motorola software reuse is considered a candidate technology for initiative and goals to improve productivity and quality.
* At Hewlett-Packard a reuse assessment of two reuse programs had indicated higher quality (reduction in defeat density ranging from 24% to 76%) and a 40% to 57% increase in productivity.
* IBM has formed the Reuse Technology Support Centre, involving close to 30 of their sites worldwide. Their best programs report savings in the millions of dollars, with reuse accounting for 20% to 30% of the software.
* Other industry examples are cited by Braun.
* Raytheon Missile Systems has reported an average of 60% reuse and a 50% increase in net productivity in new developments (1979).
* NEC Software Engineering Laboratory has reported a 6.7 to 1 productivity improvements and 2.8 to 1 quality improvements (1987).
* Fujitsu has experienced an improvement from 20% of projects on schedule to 70% on schedule (1987).
* GTE has reported 14% reuse and savings of $1.5 million, with projected figures of 50% reuse and savings of $10 million (1987).
* Softech, Inc. has reported an increase in productivity to 10 to 20 times the industry average (1987).
* University Defence Systems has reported 60% reuse in a system of 700,000 lines of Ada code (1991).
* Celsius Technology has experienced a 250% increase in productivity, projecting an additional increase of about 300% (1992). Experience reports cited by Miliet al. also document increase in productivity and quality.

**Disadvantages of Reuse:**

* Reuse often requires cleaning or transport, which have environmental costs.
* Some items, such as Freon appliances or infant auto seats, could be hazardous or less energy efficient as they continue to be used.

**CHAPTER-2**

**LITERATURE SURVEY**

**2.1 Existing Software Component Classification Techniques**

The following classification techniques had been employed to construct reuse repository, namely,

• Free Text classification

• Enumerated classification

• Attribute value classification

• Faceted classification

1. **Free Text Classification:**

Free text retrieval performs searches using the text contained within documents. The retrieval system is typically based upon a keyword search. All of the document indexes are searched to try to find an appropriate entry for the required keyword. The major drawback with this method is the ambiguous nature of the keywords used. Another disadvantage is that a search my result in many irrelevant components. A typical example for free text retrieval is the ‘grep’ utility used by the UNIX manual system. This type of classification generates large overheads in the time taken to index the material, and the time taken to make a query. All the relevant text in each of the documents relating to the component is index, which must then be searched from beginning to end when a query is made.

1. **Enumerated Classification:**

Enumerated classification uses a set of mutually exclusive classes, which are all within a hierarchy of a single dimension. The classification scheme will allow a user to find more than one item that is classified within the same section / subsection assuming that if more than one exists. For example, there may be more than one book concerning a given subject, each written by a different author. This type of classification schemes is one dimensional, and will not allow flexible classification of components into more than one place. As such enumerated classification itself does not provide a good classification scheme for reusable software components.

1. **Attribute Value Classification:**

The attribute value classification scheme uses a set of attributes to classify a component. For example, a book has many attributes such as the author, the publisher, its ISBN number and its classification code in the Dewey Decimal system. These are only example of possible attributes. Depending upon who wants a information about a book, the attributes could be concerned with the number of pages, size of the paper used, the type of print face, the publishing date, etc. Clearly the attributes relating to a book can be: Multidimensional, the book can be classified in different places using different attributes. Bulky, all possible variations for attributes could run into many tens, which may not be known at the time of classification.

1. **Faceted Classification:**

Faceted classification schemes are attracting the most attention within the software reuse community. Like the attribute classification method, various facets classify components however there are usually a lot fewer facets than there are potential attributes. Ruben Prieto-Diaz has proposed a faceted scheme that uses six facets. The functional facets are Function, Objects and Medium. The environmental facets are System type, Functional area, Settings. In Faceted classification parts are described by a set of terms, or facets. In this way facets are similar to attribute value method. However, with facets the choice of values is limited. This eliminates the problem of ambiguity in deciding the best value for a term or attribute.

**2.2 Advantages and Disadvantages of the Existing System**

1. **Free Text Classification:**

Advantages: This is flexible for less number of components. Disadvantages: It is ambiguous as it uses free text as the search key for classification. It also uses more number of indexes for classification.

1. **Enumerated Classification:**

Advantages: This is faster method. The classification of the components is due to improved search techniques.

Disadvantages: This technique can be applied only for few number of components and it is difficult to expand.

1. **Attribute Value Classification:**

Advantages: It can be applied to huge components at a time. It is very accurate and efficient technique.

Disadvantages: It is slowest of all the methods. The obtained components are difficult to classify.

1. **Faceted Classification:**

Advantages: It can be applied to many number of components and easily expandable. It is more flexible to use and classification of components.

Disadvantages: It is very slow process.

**2.3 Motivation for the Proposed Approach**

The objective of the present investigation is to categorise the reusable components in a repository and retrieve the software reusable components from a reuse repository. One among the major purposes of developing software components is to amplify a component library. The library stores reusable assets, thus serves as a primary reuse distribution mechanism. The software reuse library allows the storage, control, distribution and maintenance of reusable software assets. This reuse library forms a vital part of the software engineering environment. For the use of new engineering work tasks, this software allows software engineers to find, browse and access reusable assets as per their requirements. The charge of the asset management is typically divided into four areas: library management, library population, library operations and library maintenance. The factors which affect the success of the software reuse program are the classification used in creation of software reuse repository, which supports the software engineers and other users in the process of developing the new software. The reuse repository tool of optimum quality is essential to have a wide variety of high quality components. An efficient classification scheme must be used to arrange the components for easier retrieval of best software components. The user must find and locate appropriate software components for an effective reuse of software. Then the user has to access these components scrupulously and rapidly, if  required should be able to modify them. The software component reuse repository is associated with two major operations i.e., classification of components in efficient manner and retrieval of best components that match with the maximum requirements provided by the user. The reuse repository was previously constructed by using four dissimilar classification techniques. They include free text classification, enumerated classification, attribute value classification and faceted classification scheme.

**2.4 General Architecture for Ranking Process**

**USER INTERFACE**

**BEST COMPONENTS**

**EXISTING COMPONENTS**

**SEARCH ATTRIBUTES**

**COMPONENTS WITH RANK**

**REUSE REPOSITORY**

**RANKING PROCESS**

**SEARCH PROCESS**

**INSERT PROCESS**

**DATABASE INTERFACE**

**DATABASE**

**Fig.2.1 SYSTEM ARCHITECTURE**

**2.5 Description of the Proposed System**

In the proposed approach, some of the attributes for classification and retrieval of software components from the reuse repository are contemplated. Some of the key issues considered in this proposed work include ranking the components based on the download count, rating given by the user and the number of lines of code. The cornerstone of the proposed work is mainly on the development of a software tool with rich graphical user interface in order to categorize and retrieve the software reusable components and rank them for relevance. Further with a facility to select the best fit software reusable component from among the pertinent ones.

**Component description:** Before storing the components in the software repository, they have to be well defined. The following attributes are being considered to classify a software component.

1. Name of the component
2. Return type
3. Number of arguments
4. Programming language
5. Operating system
6. Time complexity
7. Space complexity
8. Domain

**Rank process:** The goal of this process is to rank the software reusable components. Hence the user or developer can easily choose their desired software components from the component repository. The software reusable components are designed to generate rank in this process according to their relevance of the user’s specified criteria.

Ranking is given on the basis of the following factors, the number of times  a software component is downloaded, the rating given by the users and the number of lines of code of software reusable component.

The download count of a component will be incremented when that component is downloaded from the repository. With this download count, given user rating and the number of lines in a code component rank is assessed. The rank is calculated by assigning 60% weightage for download count, 30% weightage for user rating and 10% weightage for the number of lines of code.

**2.6 Proposed Rank Algorithm**

**Input:** Component download count, user feedback rating, number of lines of code.

**Output:** Components with calculated rank.

**Begin**

1. Let the software component be **i.**
2. Let the download count of **i** th component as di .
3. Let **n** be the number of users of the software components.
4. Let the user **j**  gives the feedback of component **i** indicated by **rij**

Where 1 <= **j** <= **n** and 1 <=**rij** <=5.

1. Let **l** be the number of lines of code of a particular software component.
2. Calculate the rank of **ith** component represented by **Ranki** as

**Ranki** = **0.6 \* di + 0.3 \* - 0.1 \* l**

Where 0.6 is the component download weightage, 0.3 is the user feedback weightage and 0.1 is the number of lines used in the code weightage.

1. Display the components as per the descending order of the ranking.

**Example:**

For example the download count of a component is 4, the ratings given for those components are 3,4,2,4 and the number of lines of code is 8.

The rank then is calculated as follows

Rank = 0.6 \* 4 + 0.3 \* (3+4+2+4) – 0.1 \* 8 = 5.5

**2.7 Modules in the Proposed Approach**

1. Populating the database.
2. Classification of components.
3. Retrieval of relevant components.
4. Apply Ranking Algorithm to the obtained components.
5. Obtain best component from among the relevant components.

**Steps in Proposed Approach:**

In the proposed system components are represented by using attribute value classification scheme. The retrieval of software components from repository is a two-step process.

1. In the first step all the relevant components are retrieved by using proposed attribute keyword classification scheme.
2. In the next step most relevant components are selected by using proposed ranking algorithm.

**2.8 Dataset used and its Description**

In the approach pre-assumed dataset with attribute values like component name, return type, number of arguments, programming language, operating system, time complexity, space complexity, domain are used. The dataset mainly comprises of 80 components and can add few more components in due course of execution of the project.

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Description** |
| Utype | VARCHAR(50) | It stores the type of the user |
| Name | VARCHAR(50) | It stores the name of the user |
| Password | VARCHAR(50) | It stores the password of respective user name |

**Tables used in the Project:**

**2.8.1 TABLE NAME: users**

**2.8.2 TABLE NAME: attributes**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Description** |
| Ccode | VARCHAR(50) | It stores the unique id of particular component |
| Cname | VARCHAR(50) | It stores the component name |
| Rtype | VARCHAR(50) | It stores the return type |
| Acount | VARCHAR(50) | It stores the argument count |
| Plang | VARCHAR(50) | It stores the programming language |
| OS | VARCHAR(50) | It stores the operating system |
| Tcomp | VARCHAR(50) | It stores the time complexity |
| Scomp | VARCHAR(50) | It stores the space complexity |
| Domain | VARCHAR(50) | It stores to which domain the component belongs to |
| Cfile | VARCHAR(50) | It stores the file |

**2.8.3 TABLE NAME: att\_calculate**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Description** |
| Ccode | VARCHAR(50) | Stores unique id of component |
| Dwcnt | Int(10) | Stores the download count |
| Rating | Float | Stores the rating value given by the user |
| Rank | Float | Stores the rank of the component |

**2.8.4 TABLE NAME: ratinglater**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Description** |
| Username | VARCHAR(50) | Stores the name of the user |
| Ccode | VARCHAR(50) | Store the unique id of the component |
| Cname | VARCHAR(50) | Store the name of the component |
| Status | VARCHAR(50) | Store the status of the component |
| Rating | Float | Stores the rating value |

**CHAPTER-3**

**DESIGN**

**3.1 Software Requirements Specifications**

Software Requirement Specifications (SRS) is a document that clearly and precisely specifies each and every requirement for the software product as well as the external interfaces to hardware and firmware. Each requirement should be defined so that it can be verified by a method such as inspection, demonstration, analysis and testing. There are a number of desirable properties that a SRS should possess. In particular, the requirements documents should be:

* Correct
* Complete
* Consistent
* Functional
* Verifiable
* Easily changed

**Hardware Requirement Specifications:**

Processor : Intel core i5, 64-bit, 2.4 GHz

Memory : 4 GB RAM

Hard Disk : 500GB

**Software Requirement Specifications:**

Programming Language : Java, Jsp, HTML

Database : MySQL

Operating System : Windows 7

**Functional Requirements:**

Functional requirements specify which output file should be produced from the given file they describe the relationship between the input and output of the system, for each functional requirement a detailed description of all data inputs and their sources. The range of valid inputs must be specified.

**Non Functional Requirements:**

Describe user-visible aspects of the system that are not directly related with the functional behavior of the system. Non functional Requirements include quantitative constraints, such as response time or accuracy.

**Software Description**

**HTML**

**Hyper Text Markup Language**, commonly abbreviated as **HTML**, is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology used to create web pages,[[1]](https://en.wikipedia.org/wiki/HTML#cite_note-1) as well as to create user interfaces for mobile and web applications. Web browsers can read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically and, before the advent of Cascading Style Sheets (CSS), included cues for the presentation or appearance of the document (web page), making it a markup language, rather than a programming language. HTML elements form the building blocks of HTML pages. HTML allows images and other objects to be embedded and it 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. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <img /> and <input /> introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages. HTML markup can also refer the browser to Cascading Style Sheets (CSS) to define the look and layout of text and other material.

**CSS: Cascading Style Sheets**

**Cascading Style Sheets** (**CSS**) is a style sheet language used for describing the presentation of a document written in a markup language.[[1]](https://en.wikipedia.org/wiki/Cascading_Style_Sheets#cite_note-1) Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications.CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts.[[3]](https://en.wikipedia.org/wiki/Cascading_Style_Sheets#cite_note-3) This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

**JSP**

Java Server Pages (JSP) is a technology for developing web pages that support dynamic content which helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>. A Java Server Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands. Using JSP, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically. JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages and sharing information between requests, pages etc.

**Advantages of JSP**

Following is the list of other advantages of using JSP over other technologies:

* Active Server Pages (ASP): The advantages of JSP are twofold. First, the dynamic part is written in Java, not Visual Basic or other MS specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and non-Microsoft Web servers.
* Pure Servlets: It is more convenient to write (and to modify!) regular HTML than to have plenty of println statements that generate the HTML.
* Server-Side Includes (SSI): SSI is really only intended for simple inclusions, not for "real" programs that use form data, make database connections, and the like.
* JavaScript: JavaScript can generate HTML dynamically on the client but can hardly interact with the web server to perform complex tasks like database access and image processing etc.
* Static HTML: Regular HTML, of course, cannot contain dynamic information

**MySQL Connector Java**

MySQL provides connectivity for client applications developed in the Java programming language with MySQL Connector/J, a driver that implements the Java Database Connectivity (JDBC) API. MySQL Connector/J is a JDBC Type 4 driver. Different versions are available that are compatible with the JDBC 3.0 and JDBC 4.0 specifications. The Type 4 designation means that the driver is a pure Java implementation of the MySQL protocol and does not rely on the MySQL client libraries.

**Database**

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds. Other kinds of data stores can be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those types of systems. So nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys.

A Relational Database Management System (RDBMS) is software that:

* Enables you to implement a database with tables, columns and indexes.
* Guarantees the Referential Integrity between rows of various tables.
* Updates the indexes automatically.
* Interprets an SQL query and combines information from various tables.

**RDBMS Terminology:**

* Database: A database is a collection of tables, with related data.
* Table: A table is a matrix with data. A table in a database looks like a simple spreadsheet.
* Column: One before we proceed to explain MySQL database system, let's revise few definitions related to database. Column (data element) contains data of one and the same kind, for example the column postcode.
* Row: A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.
* Redundancy: Storing data twice, redundantly to make the system faster.
* Primary Key: A primary key is unique. A key value can not occur twice in one table. With a key, you can find at most one row.
* Foreign Key: A foreign key is the linking pin between two tables.
* Compound Key: A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.
* Index: An index in a database resembles an index at the back of a book.
* Referential Integrity: Referential Integrity makes sure that a foreign key value always points to an existing row.

**Advantages of Database System**

**a. Data Sharing**

The data for different applications or subsystems is present in the same location. So, data that is common among different applications need not be stored repeatedly, as was the case in the file system based environment.

**b. Data Independence**

Data and programs are independent of each other, so change is one has no or minimum effect on the other. Data and its structure are stored in the database where as application programs manipulating this data are stored separately, the change in one does not necessarily affect the other.

**c. Controlled Redundancy**

This means that we do not have to duplicate the data unnecessarily. We do duplicate data in the databases; however, this duplication is deliberate and controlled.

**d. Better Data Integrity**

This is an important feature which validates the data being entered in the database. Since the data is being placed at a central location and managed by DBMS, it provides a conducive environment to check or to ensure that the data being entered into the database is actually valid. Integrity of data is very important, since all the processing and the information produced in turn are based on this data. If the data entered is not valid, we cannot rely on the correctness of the results produced. The business decisions are made based on the information present in the database and invalid information leads to wrong decisions, and business in turn collapses. In the database environment, DBMS provides many features to ensure the integrity of data; hence we have more reliable data.

**3.2 Project Flow Chart**

DESIGNING THE DATABASE

INSERT THE COMPONENTS INTO THE REPOSITORY

SEARCH THE COMPONENT

APPLY RANKING ALGORITHM FOR THE RELEVANT COMPONENTS

DISPLAY THE COMPONENTS WITH RANK

DOWNLOAD COMPONENT

**Fig.3.1 Project Flow Chart**

**3.3 Data Flow Diagrams**

A data flow diagram is a graphical representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context level DFD first which shows the interaction between the system and outside entities. This context level DFD is then exploded to show more details of the system being modeled. The Data Flow diagram shows the flow of data or information. It can be partitioned into single processes or functions. Data Flow Diagrams can be grouped together or decomposed into multiple processes.

There can be physical DFD"s that represent the physical files and transactions, or they can be business DFD"s (aka logical, or conceptual).

**Components:** A data flow diagram illustrates the processes, data stores, and external entities in a business or other system and the connecting data flows.

User

Database input output

Three components of Data Flow Diagram:

Function

File/Database

Input/output

**Dataflow Diagram Notations**

**External Entities / Terminators:** These are outside of the system being modeled. Terminators represent where information comes from and where it goes. In designing a system, we have no idea about what these terminators do or how they do it.

**Processes:** These are the functions that modify the inputs m the process of generating the outputs.

**Data Stores:** Data stores represent a place in the process where data comes to rest. A DFD does not say anything about the relative timing of the processes, so a data store might be a place to accumulate data over a year for the annual accounting process. A data store is a repository for data. Data stores can be manual, digital, temporary.

**Data Flows:** Data flow are how data moves between terminators, processes, and data stores (those that cross the system boundary are known as input output descriptions).

**Duplication:** External entities and data stores can be duplicated in the system for more clarity, while processes cannot. External entities that have been replicated are marked by an asterisk (\) in the lower left part of the oval that represents that entity. Data stores have a double line on the left side of their box.

**Developing a DFD:**

1. The system makes a context level DFD, which shows the interaction (data flows) between the system (represented by one process) and the system environment (represented by terminators).
2. The system is decomposed in lower level DFD (Zero) into a set of processes, data stores, and the data flows between these processes and data stores.
3. Each process is then decomposed into an even lower level diagram containing its sub processes.
4. This approach then continues on the subsequent sub processes, until a necessary and sufficient level of detail is reached which is called the primitive process.

The corresponding DFD regarding project design are drawn below:

Component details Component Retrieved

**Fig.3.2** **Level-0 Data Flow diagram**

.

Figure 3.3 shows the first level data flow diagram of software reuse repository system. All the users should be login before accessing the repository system. Administrator also needs to be login in order to access/upload the components

User

Administrator

Registered Users

Administrator

User

Administrator

Database

**Fig.3.3** **Level-1 Data Flow diagram**

**3.4 UML (Unified Modeling Language)**

UML or Unified Modeling Language is a specification language that is used in the software engineering field. It can be defined as a general purpose language that uses a graphical designation which can create an abstract model. This abstract model can then be used in a system. This system is called the UML model. XML can be used to serialize the UML model. The Unified Modeling Language is commonly used to visualize and construct systems which are software intensive. Because software has become much more complex in recent years, developers are finding it more challenging to build complex applications within short time periods. It can also be used to build models for system engineering, business process, and organization structures. UML it is very important to differentiate between the system diagrams and the UML model. A system diagram is a portion of graphical symbol, and it is used to denote the system's model. The model will have what is called a semantics backplane, and this is a document that is used to deal with the diagrams and elements of model.

**Models**

There are three models, they are

1. Object Model: The Object model will be responsible for demonstrating the system structure and substructure, and it will do this by using relationships, attributes, and operations. Class diagrams are closely related to this model.

2. Functional Model: The functional model will demonstrate the system functionality from the view of the user. It will be directly connected to the use case diagrams.

3. Dynamic Model: The dynamic model will be responsible for demonstrating inner behaviour of a system. It is directly connected to state machine diagrams, sequence diagrams, and activity diagrams.

In the above process each model has specific roles that it must play for the proper function of system.

**UML elements:** For UML, a diagram is an element which must define which things should be modelled in a system.

They are

1. Use case Diagram

2. Class Diagram

3. Sequence Diagram

4. Collaboration Diagram

5. Activity Diagram

**Use Case Diagram**

A use case diagram describes a set of sequences in which each sequence indicates the relation with outside things. A use case involves the interaction of actor and system.

There exist 3 types of relationships-

1. Association

2. Dependency

3. Generalization.

Use case diagrams can contain

* + Actors - "things" outside the system
  + Use cases - system boundaries identifying what the system should do.

Use case diagram can be used during analysis to capture the system requirements and to understand how the system should work. During the design phase, you can use use-case diagrams to specify the behaviour of the system is implemented.

Use cases provide a means to,

* Capture system requirements.
* Communicate with end users and domain experts.
* Test the system.



**Fig.3.4** **Use Case Diagram**

**Class Diagram**

The class diagram contains icons representing classes, interfaces and their relationships.

Class: A class is a set of objects that share a common structure and common behaviour (the same attributes, operations and semantics). A class is a abstraction of real-world items. When these items exist in the real world, they are instance of the class and are referred to as objects.

A class icon is a 3-part box.

|  |
| --- |
| Class name |
| Attributes |
| Operations |

**Sequence Diagrams**

A sequence diagram is a graphical view of scenario that shows object interaction in a time based sequence what happens first what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces. A sequence diagram has two dimensions: vertical placement represents time and horizontal placement represents different objects.

Link: Objects interact through their links to other objects. A link is an instance of an association, analogous to an object being instance of a class. A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes.

**Collaboration diagrams**

A collaboration diagram is an interaction diagram that shows the order of messages that implement an operation or a transaction. Collaboration diagrams show objects, their links, and their messages. They can also contain simple class instances and class utility instances. Each collaboration diagram provides a view of the interactions or structural relationships that occur between objects and object-like entities in the current model. Collaboration diagrams contain icons representing objects. The “Create Collaboration Diagram” Command creates a collaboration diagram from information contained in the sequence diagram. The “Create Sequence Diagram” Command creates a sequence diagram from information contained in the interaction's collaboration diagram. The “Goto Sequence Diagram” and “Goto Collaboration Diagram” commands traverse between an interaction's two representations.



**Fig.3.5 Class Diagram**



**Fig. 3.6 Sequence Diagram for User**



**Fig.3.7 Sequence Diagram for Admin**



**Fig.3.8 Collaboration Diagram for User**



**Fig.3.8 Collaboration Diagram for Admin**

** Fig.3.10 Activity Diagram**

**CHAPTER-4**

**IMPLEMENTATION**

**4.1 Implementation**

The proposed approach was implemented using HTML, JSP and MySQL as explained in the design. The implementation of the proposed approach in the system is given in this chapter.

* + 1. **Login:**

Input: Username and Password. Output: If user is valid then login or else prompt to enter correct user credentials. Algorithmic Steps: 1. Send Username and Password for validation. 2. Check if username exists. 3. If username exists, check for password matching or else return no user exists. 4. Login to the respective page if valid credentials or else return password error.

**4.1.2 Add New Component:**

Input: Component Code, Component Name, Return type, programming language, operating system, time complexity, space complexity and Uploading file. Output: Component will be added to the repository. Algorithmic Steps: 1. Check if all the fields are given and file uploading is valid. 2. If valid upload to the repository or else prompt error.

**4.1.3 Edit Component:**

Input: Select a component that has to be edited. Output: Component will be updated in the repository. Algorithmic Steps: 1. Select a component which is to be edited. 2. Edit the required fields.

**4.1.4 Delete Component:**

Input: Select Component that has to be deleted. Output: Component will be deleted from the repository. Algorithmic Steps: 1. Check if the given component is present in the repository. 2. If present delete the component or else prompt error.

* + 1. **View Database:**

The user gets a complete view of the repository and its components.

* + 1. **User Search Page:**

Input: Attribute values Output: Component Algorithmic Steps: 1. Capture all necessary component attributes on which search is to be made. 2. Download the required component and rate it. 3. Apply Ranking algorithm for the rank of relevance.

**CHAPTER-5**

**TESTING**

**5.1 Introduction to Testing**

After finishing the development of any computer based system the next complicated time consuming process is system testing. During the time of testing only the development company can know that, how far the user requirements have been met out, and so on.

Following are the some of the testing methods applied to this effective project:

# Source Code Testing:

This examines the logic of the system. If we are getting the output that is required by the user, then we can say that the logic is perfect.

**Specification Testing:**

We can set with, what program should do and how it should perform under various condition. This testing is a comparative study of evolution of system performance and system requirements.

**Module Testing:**

In this the error will be found at each individual module, it encourages the programmer to find and rectify the errors without affecting the other modules.

**Unit Testing:**

Unit testing focuses on verifying the effort on the smallest unit of software-module. The local data structure is examined to ensure that the date stored temporarily maintains its integrity during all steps in the algorithm’s execution. Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.

**Integration Testing:**

Data can be tested across an interface. One module can have an inadvertent, adverse effect on the other. Integration testing is a systematic technique for constructing a program structure while conducting tests to uncover errors associated with interring.

**Validation Testing:**

It begins after the integration testing is successfully assembled. Validation succeeds when the software functions in a manner that can be reasonably accepted by the client. In this the majority of the validation is done during the data entry operation where there is a maximum possibility of entering wrong data. Other validation will be performed in all process where correct details and data should be entered to get the required results.

**Recovery Testing:**

Recovery Testing is a system that forces the software to fail in variety of ways and verifies that the recovery is properly performed. If recovery is automatic, re-initialization, and data recovery are each evaluated for correctness.

**Security Testing:**

Security testing attempts to verify that protection mechanism built into system will in fact protect it from improper penetration. The tester may attempt to acquire password through external clerical means, may attack the system with custom software design to break down any defences to others, and may purposely cause errors.

**Performance Testing:**

Performance Testing is used to test runtime performance of software within the context of an integrated system. Performance test are often coupled with stress testing and require both software instrumentation.

**Whitebox Testing:**

White-box testing (also known as clear box testing, glass box testing, transparent box testing,

and structural testing) is a method of testing software that tests internal structures or workings

of an application, as opposed to its functionality (i.e. black-box testing).

**Blackbox Testing:**

Black- box testing focuses on functional requirement of software. It enables to derive tests of input conditions that will fully exercise all functional requirements for a program. Black box testing attempts to find error in the following category:

* Incorrect or missing function
* Interface errors
* Errors in data structures or external database access and performance errors.

**Output Testing:**

After performing the validation testing, the next step is output testing of the proposed system since no system would be termed as useful until it does produce the required output in the specified format. Output format is considered in two ways, the screen format and the printer format.

**User Acceptance Testing:**

User Acceptance Testing is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system users at the time of developing and making changes whenever required.

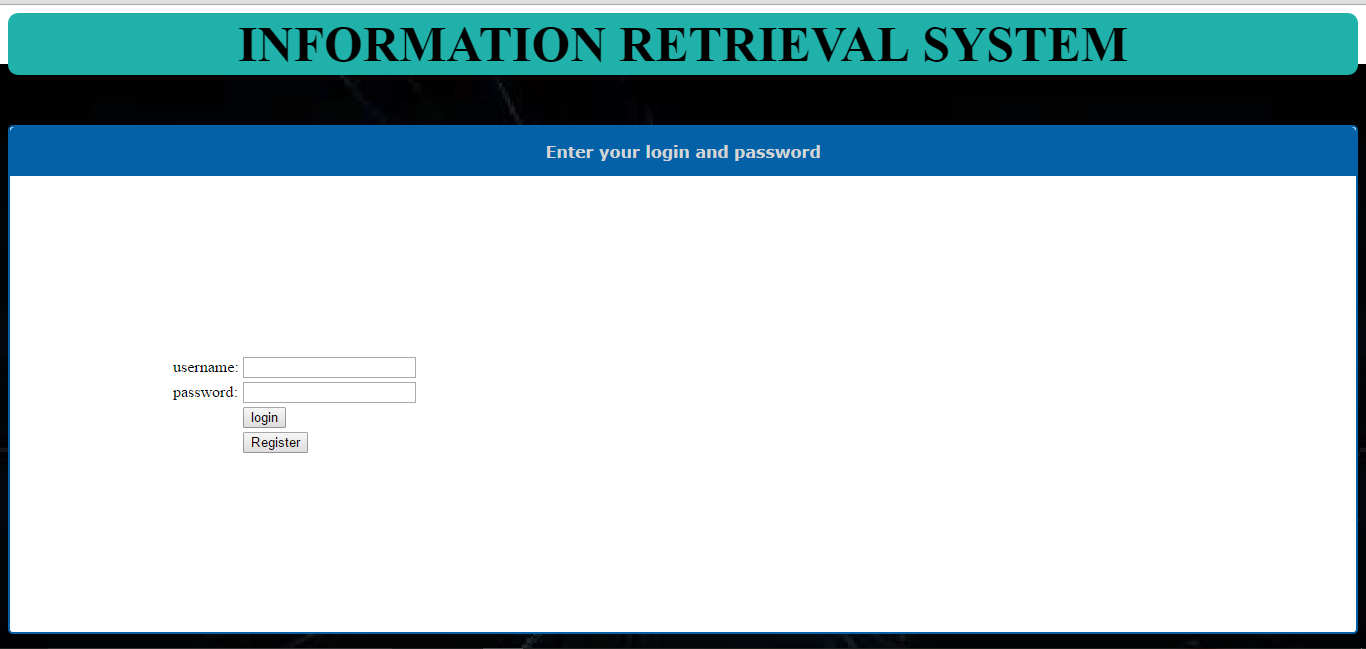
**CHAPTER-6**

**RESULTS**

**6.1 Results**

**Login Page:**

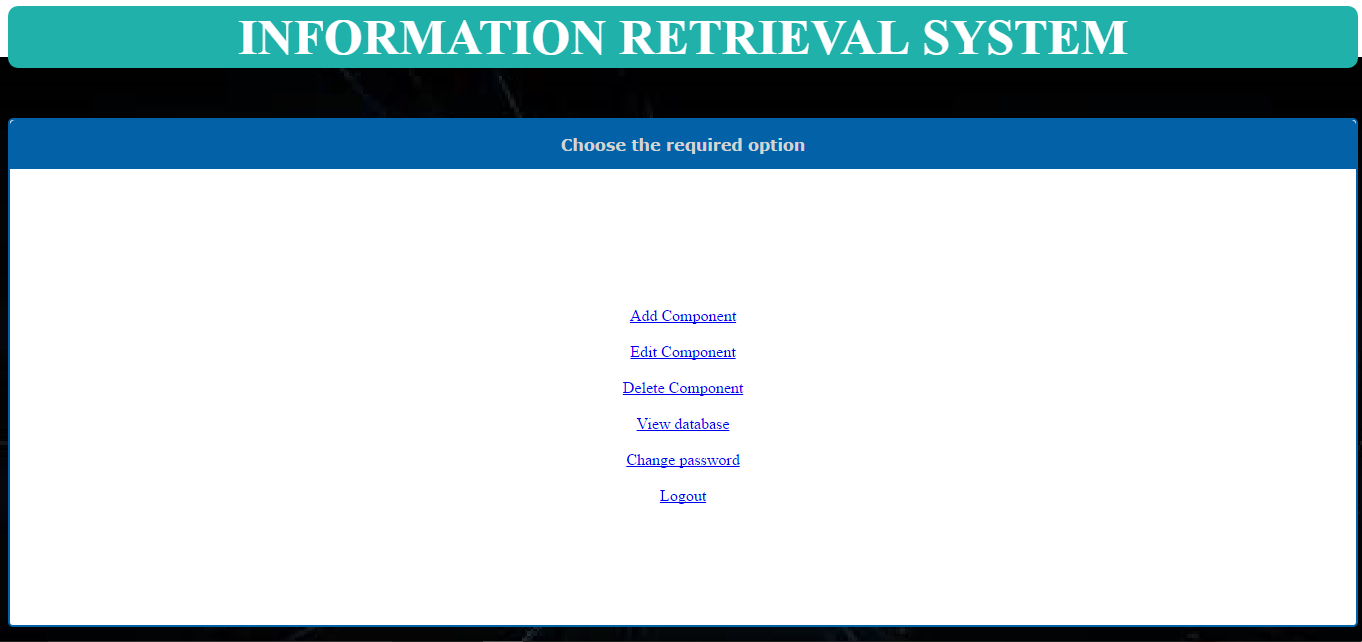
This is Login Page of the repository. Admin and Users with valid credentials can login through this page.

****

**Fig.6.1 Login Page**

**Administrator Page:**

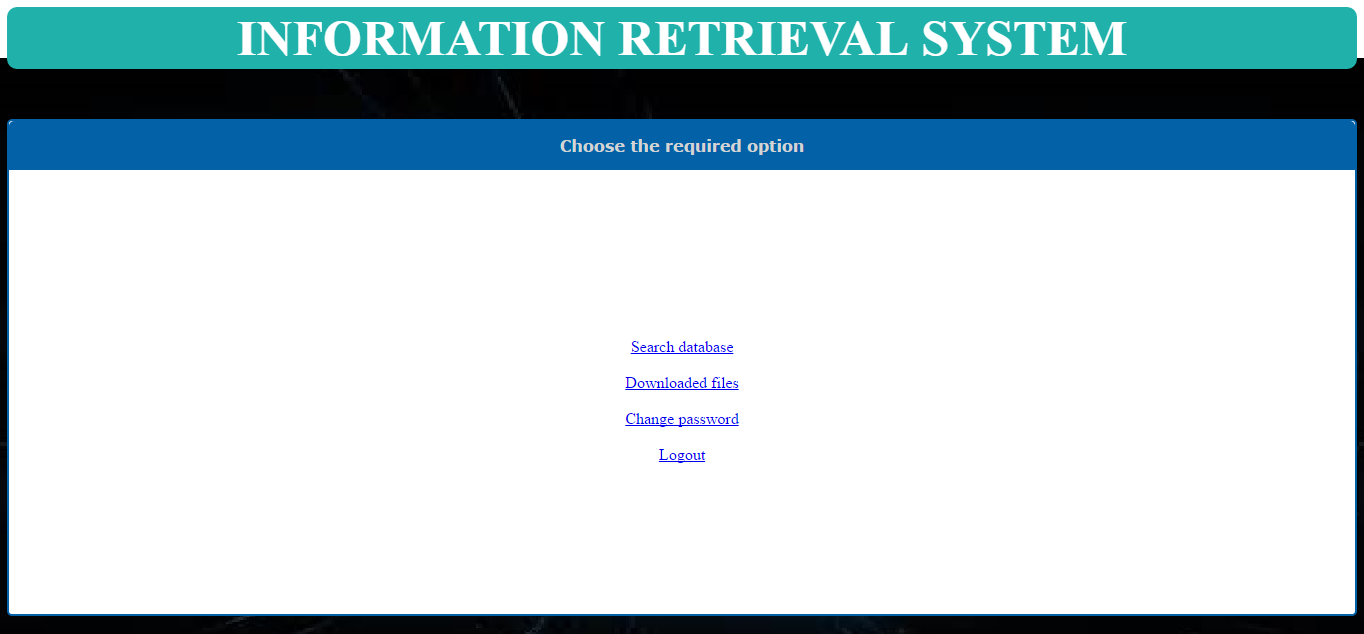
After successful login, the administrator is redirected to admin options page.

****

**Fig.6.2 Admin Page**

**User Page:**

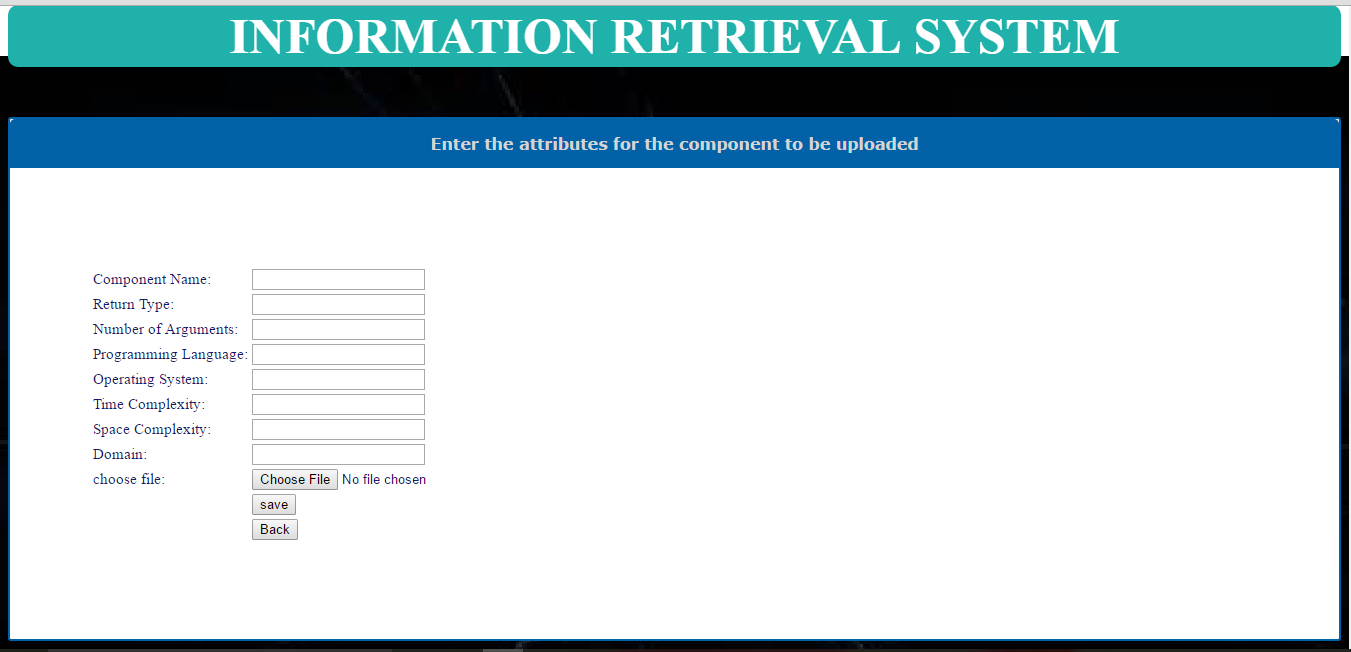
After successful login, the user is redirected to user options page.



**Fig.6.3 User Page**

**Add New Component**

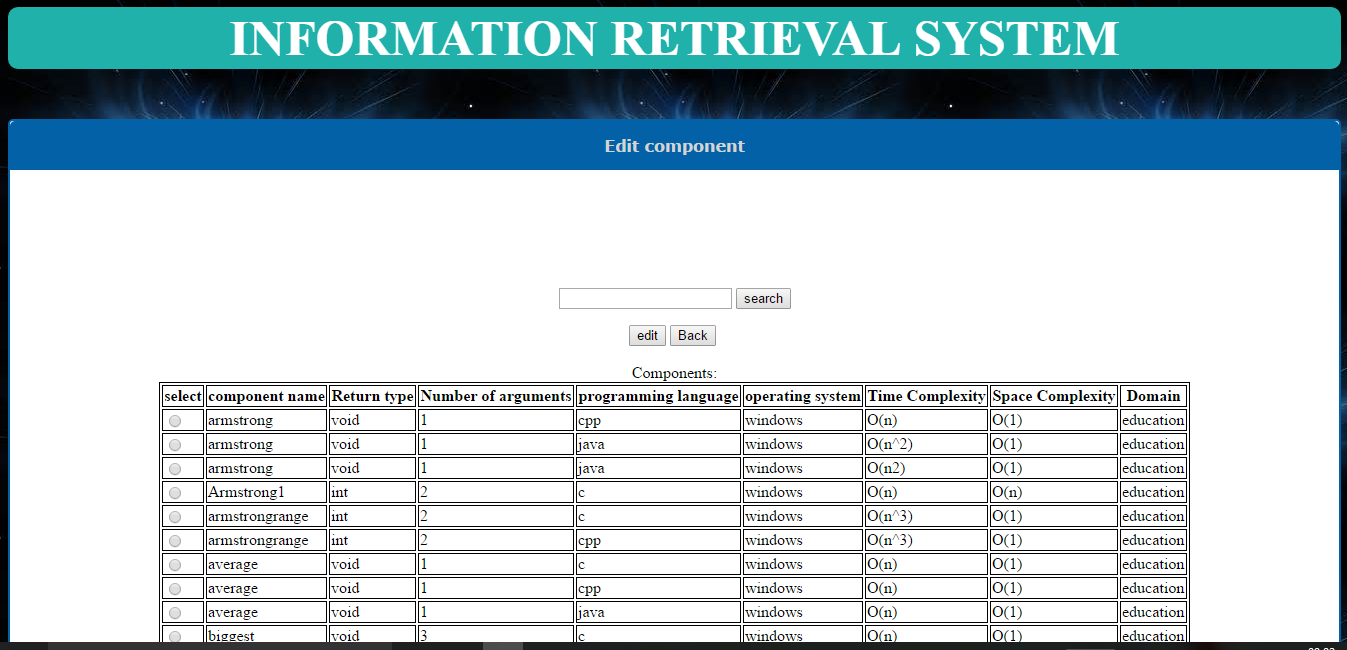
Add component page allows administrator to add new components to the repository. A file is to be uploaded and the attributes can be given using this page.



**Fig.6.4 Add New Component Page**

**Edit Component:**

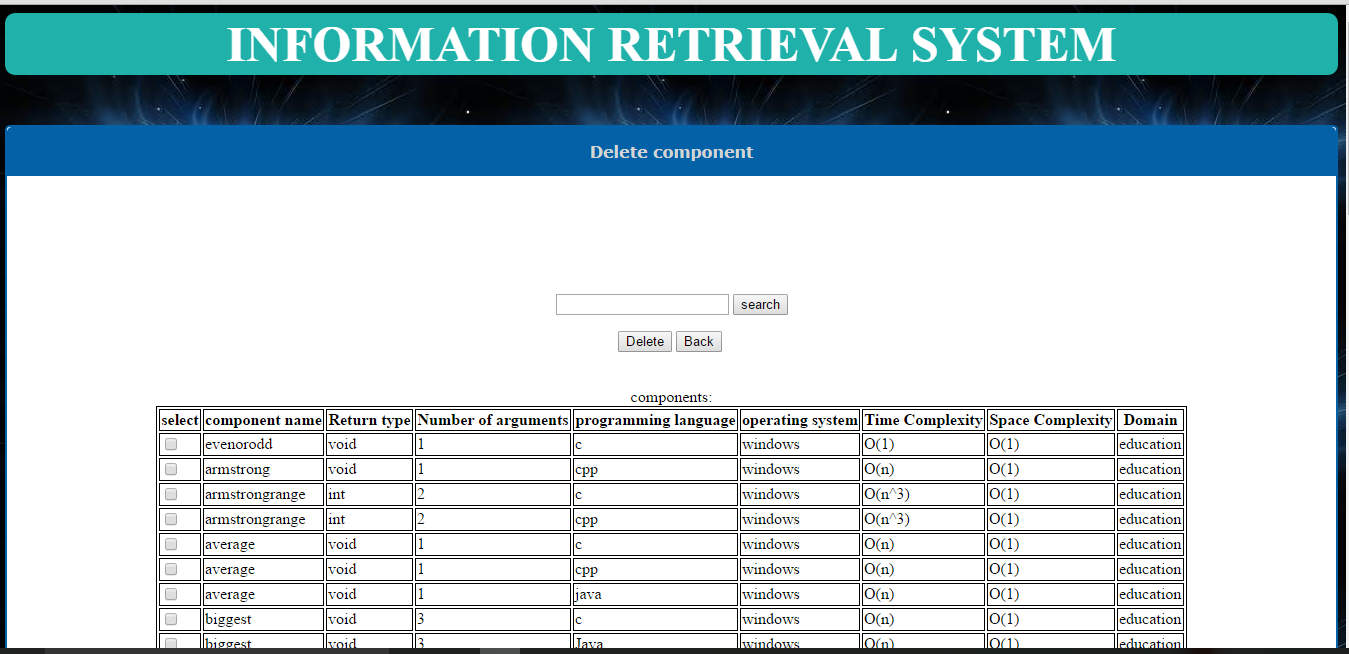
A component can be edited from the repository using this page.



**Fig.6.5 Edit Component Page**

**Delete Component:**

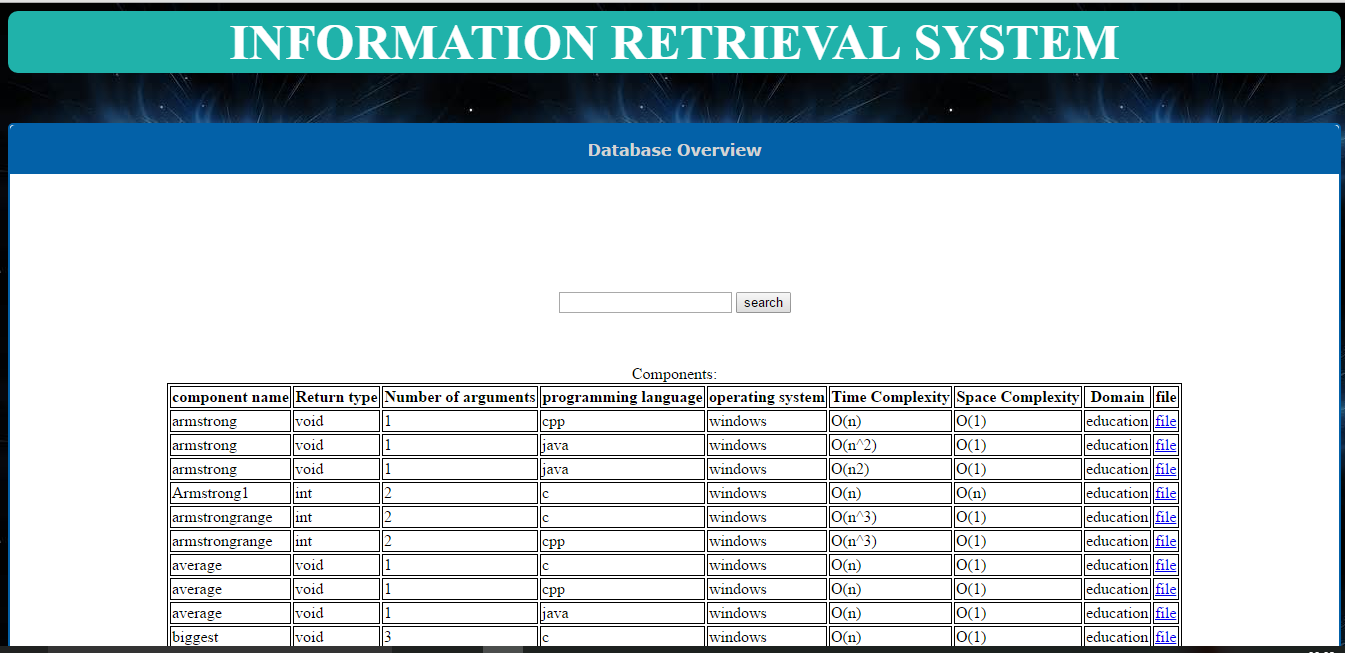
A component can be deleted from the repository using this page.



**Fig.6.6 Delete Component Page**

**View Database:**

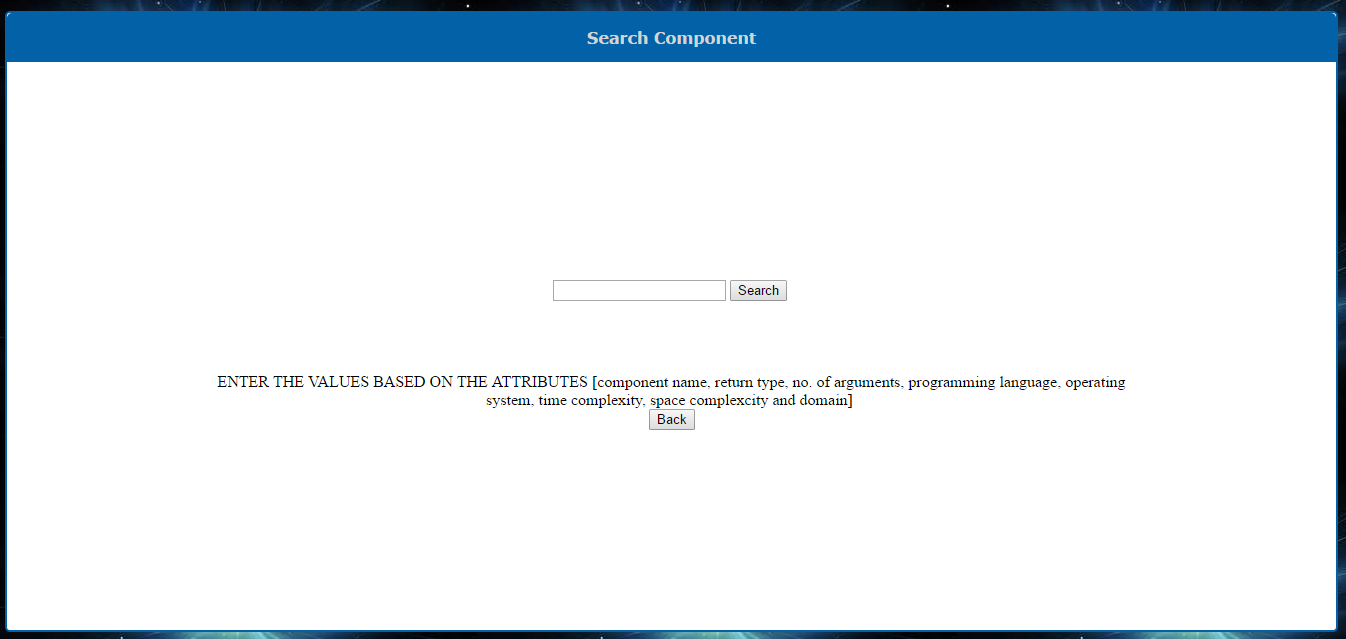
The components from the repository can be viewed from this page.



**Fig.6.7 View Database Page**

**User Search Page:**

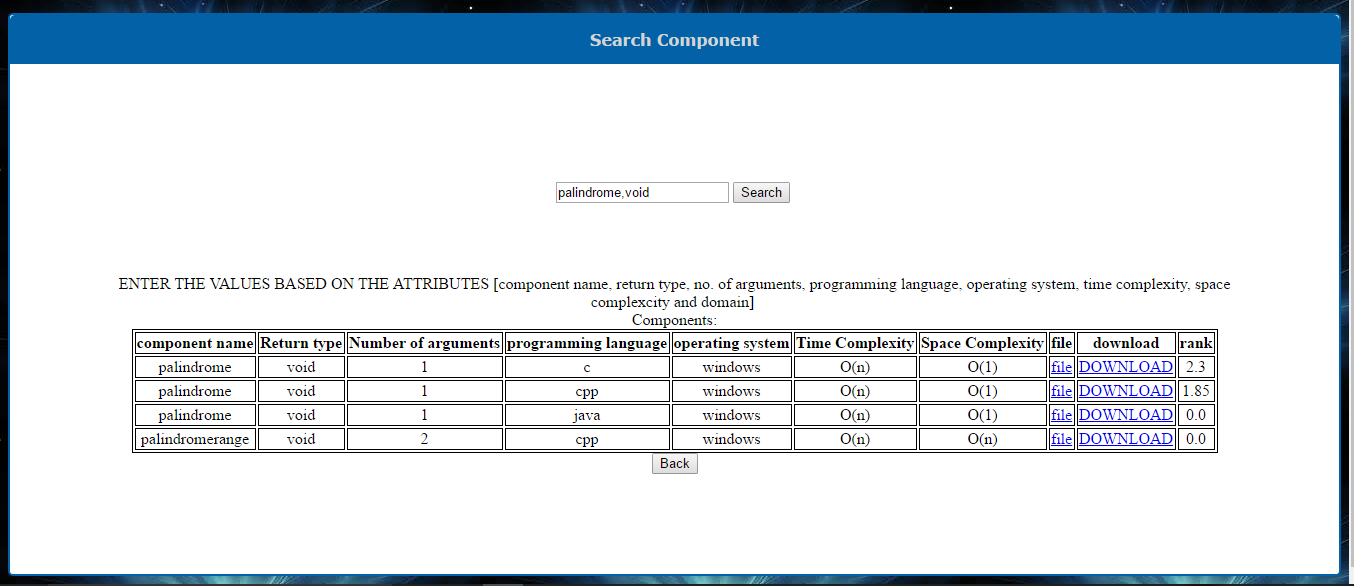
User can search the component and download the component using this page.

****

**Fig.6.8 User Search Page**

**Results Generated from the Search Page:**

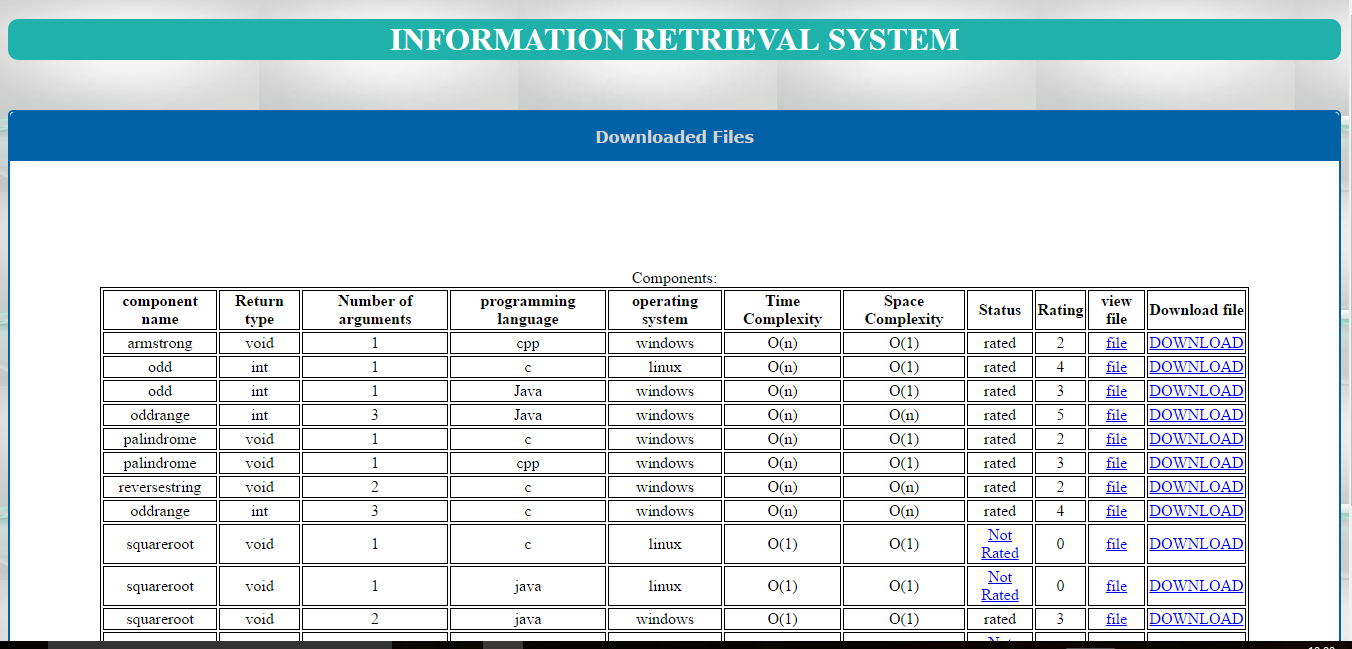
The components are obtained based on the attributes given by the user in the descending order of rank.

****

**Fig.6.9 Results Generated from the Search Page**

**User Downloaded Files Page**

User can rate the component and can download the component using this page.

****

**Fig.6.10 User Downloaded Files Page**

**6.2 Code**

**Login Page (Login1.jsp):**

<html>

<head>

<title> login </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

}

</style>

</head>

<body background="images/images.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em;font-weight:bold;">Enter your login and password</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 200px">

<form method="post" action="" name="" target="\_top">

<table>

<tr><td>username:</td><td><input type="text" name="login"></td></tr>

<tr><td>password:</td><td><input type="password" name="password"></td></tr>

<tr><td>&nbsp;</td><td><input type="submit" value="login" formaction="loginverify.jsp"></td></tr>

<tr><td>&nbsp;</td><td><input type="submit" value="Register" formaction="register.jsp"></td></tr>

</table>

</form>

</body>

</html>

**Administrator Page (aoptions.jsp):**

<html>

<head>

<title> Options </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

</style>

</head>

<body background="images/test.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em;font-weight:bold;">Choose the required option</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 120px">

<a href="addcmp.jsp">Add Component</a></br></br>

<a href="editcmp.jsp">Edit Component</a></br></br>

<a href="delcmp.jsp">Delete Component</a></br></br>

<a href="viewdb.jsp">View database</a></br></br>

<a href="chngpwd.jsp">Change password</a></br></br>

<a href="logout.jsp">Logout</a>

</body>

</html>

**User Page (uoptions.jsp):**

<html>

<head>

<title> Options </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

</style>

</head>

<body background="images/test.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em;font-weight:bold;">Choose the required option</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 150px">

<a href="searchdb.jsp">Search database</a></br></br>

<a href="downloadedfiles.jsp">Downloaded files</a></br></br>

<a href="chngpwd.jsp">Change password</a></br></br>

<a href="logout.jsp">Logout</a>

</body>

</html>

**Add New Component (addcmp.jsp):**

<html>

<head>

<title>Add component </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

}

table{font-family:times,serif;

font-size:15;

color:191970

}

</style>

</head>

<body background="images/test.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:3px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em;font-weight:bold;">Enter the attributes for the component to be uploaded</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 80px">

<form method="post" action="addcmp2.jsp" name="" target="\_top">

<table>

<tr><td>Component Name:</td>

<td><input type="text" name="cname" ></td>

</tr>

<tr><td>Return Type:</td>

<td><input type="text" name="rtype" >

</td>

</tr>

<tr><td>Number of Arguments:</td><td><input type="number" name="acount" ></td></tr>

<tr><td>Programming Language:</td>

<td><input type="text" name="plang" >

</td>

</tr>

<tr><td>Operating System:</td>

<td><input type="text" name="os">

</td>

</tr>

<tr><td>Time Complexity:</td>

<td><input type="text" name="tcomp">

</td>

</tr>

<tr><td>Space Complexity:</td>

<td><input type="text" name="scomp" >

</td>

</tr>

<tr><td>Domain:</td>

<td><input type="text" name="domain" >

</td>

</tr>

<tr><td>choose file:</td><td><input type="file" name="cfile" value="open"/>

<%-- <input type="submit" value="upload" > --%>

<%-- <progress></progress></td></tr> --%>

<tr><td>&nbsp;</td><td><input type="submit" value="save" formaction="addcmpdb.jsp"></td>

<tr><td>&nbsp;</td><td><input type="submit" value="Back" formaction="aoptions.jsp"></td></tr>

</table>

<%! String act; %>

<% String type=(String)session.getAttribute("ut");

if(type.equals("A"))

act="aoptions.jsp";

else if(type.equals("B"))

act="uoptions.jsp";

%>

<%-- <input type="submit" value="Back" formaction="<%=act%>"/> --%>

</form>

</body>

</html>

**Edit Component (editcmp.jsp):**

<%@page import="java.sql.\*"%>

<html>

<head>

<title> Edit component </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

}

table,th,td{ border:1px solid black}

</style>

</head>

<body background="images/abc.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em; font-weight:bold;">Edit component</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 100px">

<form action="sample.jsp">

<input type="text" name="search">

<input type="submit" value="search">

</form>

<%Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/components","root","Welcome1");%>

<form method="post" action="" name="" target="\_top">

<input type="submit" value="edit" formaction="editcmp1.jsp">

<input type="submit" value="Back" formaction="<%=act%>">

<br>

<br>

Components:

<table align=CENTER>

<tr><th>select</th><th>component name</th><th>Return type</th><th>Number of arguments</th><th>programming language</th><th>operating system</th><th>Time Complexity</th><th>Space Complexity</th><th>Domain</th></tr>

<%Statement stmt=con.createStatement();

ResultSet re=stmt.executeQuery("select cname,rtype,acount,plang,os,tcomp,scomp,ccode, domain from attributes order by cname");

while(re.next())

{

%>

<tr>

<td> <input type="radio" name="editc" value="<%=re.getString(8)%>" > </td>

<td> <%=re.getString(1)%></td>

<td> <%=re.getString(2)%></td>

<td> <%=re.getString(3)%></td>

<td> <%=re.getString(4)%></td>

<td> <%=re.getString(5)%></td>

<td> <%=re.getString(6)%></td>

<td> <%=re.getString(7)%></td>

<td> <%=re.getString(9)%></td>

<%

}

%>

</table>

<%! String act; %>

<% String type=(String)session.getAttribute("ut");

if(type.equals("A"))

act="aoptions.jsp";

else if(type.equals("B"))

act="uoptions.jsp";

%>

</form>

</body>

</html>

**Delete Component (delcmp.jsp):**

<%@page import="java.sql.\*"%>

<html>

<head>

<title> View database </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

}

table,th,td{ border:1px solid black}

</style>

</head>

<body background="images/abc.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em; font-weight:bold;">Delete component</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 100px">

<form action="sample1.jsp">

<input type="text" name="search">

<input type="submit" value="search">

</form>

<%

Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/components","root","Welcome1");%>

<form method="post" action="" name="" target="\_top">

<input type="submit" value="Delete" formaction="delcmpdb.jsp">

<input type="submit" value="Back" formaction="aoptions.jsp"/>

<br>

<br>

<br>

components:

<table align=CENTER>

<tr><th>select</th><th>component name</th><th>Return type</th><th>Number of arguments</th><th>programming language</th><th>operating system</th><th>Time Complexity</th><th>Space Complexity</th><th>Domain</th></tr>

<%

Statement stmt=con.createStatement();

ResultSet re=stmt.executeQuery("select cname,rtype,acount,plang,os,tcomp,scomp,ccode,domain from attributes");

int i=1;

while(re.next())

{

%>

<tr>

<td> <input type="checkbox" name="c<%=i%>" value="<%=re.getString(8)%>" > </td>

<td> <%=re.getString(1)%></td>

<td> <%=re.getString(2)%></td>

<td> <%=re.getString(3)%></td>

<td> <%=re.getString(4)%></td>

<td> <%=re.getString(5)%></td>

<td> <%=re.getString(6)%></td>

<td> <%=re.getString(7)%></td>

<td> <%=re.getString(9)%></td>

<%

i++;

}

%>

</table>

</form>

</body>

</html>

**View Database (viewdb.jsp):**

<%@page import="java.sql.\*"%>

<html>

<head>

<title> View database </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

}

table,th,td{ border:1px solid black}

</style>

</head>

<body background="images/abc.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em; font-weight:bold;">Database Overview</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 100px">

<form action="sample.jsp">

<input type="text" name="search">

<input type="submit" value="search">

</form>

<%Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/components","root","Welcome1");%>

<form method="post" action="" name="" target="\_top">

<!--

<input type="submit" value="Back" formaction="<%=act%>"> -->

<br>

<br>

Components:

<table align=CENTER>

<tr><th>component name</th><th>Return type</th><th>Number of arguments</th><th>programming language</th><th>operating system</th><th>Time Complexity</th><th>Space Complexity</th><th>Domain</th><th>file</th></tr>

<%Statement stmt=con.createStatement();

ResultSet re=stmt.executeQuery("select cname,rtype,acount,plang,os,tcomp,scomp,cfile,domain from attributes order by cname");

while(re.next())

{

%>

<tr>

<td> <%=re.getString(1)%></td>

<td> <%=re.getString(2)%></td>

<td> <%=re.getString(3)%></td>

<td> <%=re.getString(4)%></td>

<td> <%=re.getString(5)%></td>

<td> <%=re.getString(6)%></td>

<td> <%=re.getString(7)%></td>

<td> <%=re.getString(9)%></td>

<td> <a href="./sourcefiles/<%=re.getString(8)%>">file</a></td>

</tr>

<%

}

%>

</table>

<%! String act; %>

<% String type=(String)session.getAttribute("ut");

if(type.equals("A"))

act="aoptions.jsp";

else if(type.equals("B"))

act="uoptions.jsp";

%>

<input type="submit" value="Back" formaction="<%=act%>"/>

</form>

</body>

</html>

**User Search Page (searchdb.jsp)**

<%@page import="java.sql.\*"%>

<%@page import="java.util.\*"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"

pageEncoding="ISO-8859-1"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<title> View database </title>

<style type="text/css">

h1{font-family:Times new roman;

font-size:50;

fontweight:bold;

color:#ffffff;

background-color:#20B2AA;

border:thin groove;

border-radius:10px;

border: 2px solid #20B2AA

}

table,th,td{ border:1px solid black}

</style>

</head>

<body background="images/abc.jpg">

<div style="text-align:CENTER">

<h1>INFORMATION RETRIEVAL SYSTEM</h1>

<div style="width: auto; max-width:2024px; background-color: #FFffFF; border: 2px solid #0361A8; border-radius:5px; margin: 50px auto auto;">

<div style="background: #0361A8; border-radius: 5px 5px 0px 0px; padding: 15px;"><span style="font-family: verdana,arial; color: #D4D4D4; font-size: 1.00em; font-weight:bold;">Search Component</span></div>

&nbsp&nbsp&nbsp<div></div>

<div style="background: ; padding: 100px">

<%! String act,qry; %>

<%Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/components","root","Welcome1");

String vv="";

if(request.getParameter("val")!=null)

vv=request.getParameter("val");

%>

<form method="get" action="" name="" target="\_top">

<input type="text" name="val" value="<%=vv%>">

<input type="submit" name="search" value="Search">

<br>

<br>

<br>

<br>

<br>ENTER THE VALUES BASED ON THE ATTRIBUTES [component name, return type, no. of arguments, programming language, operating system, time complexity, space complexcity and domain]

<br>

<%

if(request.getParameter("search")!=null)

{

String val=request.getParameter("val");

String s[]=val.split(",") ;

String colnames[]={"cname","rtype","acount","plang","os","tcomp","scomp","domain"};

qry="select A.\*,ifnull(B.rank,0) rank from attributes A left join att\_calculate B on A.ccode=B.ccode where A.ccode in";

ArrayList<String> qal=new ArrayList<String>();

for(int i=s.length-1;i>=0;i--)

{

qry+=" (select ccode from attributes where (";

for(int j=0;j<colnames.length;j++)

{

qry+= colnames[j]+" like '%"+s[i]+"%'" ;

if(j!=colnames.length-1)

qry+=" or ";

else

{

if(i!=0)

qry+=") and ccode in ";

}

}

qal.add(qry);

}

for(int i=0;i<qal.size();i++)

{

qry+=" ) ";

}

qry+=") order by B.rank desc";

if(x==1)

return;

%>

Components:

<table align=CENTER>

<tr><th>component name</th><th>Return type</th><th>Number of arguments</th><th>programming language</th><th>operating system</th><th>Time Complexity</th><th>Space Complexity</th><th>file</th></th><th>download</th><th>rank</th></tr>

<%Statement stmt=con.createStatement();

ResultSet re=stmt.executeQuery(qry);

int c=0;

while(re.next())

{

c++;

String ccode=re.getString(1);

%>

<tr>

<td> <%=re.getString(2)%></td>

<td> <%=re.getString(3)%></td>

<td> <%=re.getString(4)%></td>

<td> <%=re.getString(5)%></td>

<td> <%=re.getString(6)%></td>

<td> <%=re.getString(7)%></td>

<td> <%=re.getString(8)%></td>

<td> <a href="./sourcefiles/<%=re.getString(9)%>">file</a></td>

<td> <a href="http://localhost:8080/IRS/downloadfile.jsp?file=<%=re.getString(9)%>&ccode=<%=ccode%>&cname=<%=re.getString("cname")%>&type=1"> DOWNLOAD</a></td>

<td><%=re.getFloat("rank")%></td>

</tr>

<%

}

if(c==0)

out.println("<h1>No matching results</h1>");

%>

</table>

<%

}

%>

<% String type=(String)session.getAttribute("ut");

if(type.equals("A"))

act="aoptions.jsp";

else if(type.equals("B"))

act="uoptions.jsp";

%>

<input type="submit" value="Back" formaction="<%=act%>"/>

</form>

</body>

</html>

**CONCLUSION AND FUTURE SCOPE**

**Conclusion of the Present Work**

Effective retrieval of the information from a repository is difficult and the time consuming. Software reuse is based on effective information retrieval. In the absence of a proper retrieval mechanism the importance of the software reuse reduces drastically.

Now-a-days it is very difficult to develop quality software with the focus on cost reduction. Software reuse seems to be one of the best solutions for software development Industries. Reuse is the process of developing new application using existing assets. The main problem in construction of any effective reuse repository is the representation of the components in the repository. User specifies the defined attributes for the component and these attributes are used for indexing and retrieval of components.

The success of the software reuse programs depends on the classification technique used in creation of software reuse repository. This supports the software engineers and other users in the process of developing the new software.

The main objectives of this proposed work is classification and retrieval of software components from reuse repository effectively. Here the attribute classification scheme is very flexible and easier to use. In this work Ranking Algorithm, approach for the selection of best fit component from all the relevant components is presented very effectively. The implemented system considers the mental perception of the user and classifies software reusable components in the repository along with the user knowledge or experience. In this proposed system, users can simply specify the attributes of classification scheme of the repository according to their keywords while using the system. The system becomes self-learning as, more people use the system and gives their knowledge to the system to increase the vocabulary and ultimately its capability to return components that are needed to solve their problems.

The new integrated classification scheme is realized in this work with utmost precision and accuracy for most efficient classification and retrieval of best suitable software reusable components.

**Future Scope of the Present Work**

Future work includes the retrieval of the software components with multimedia and intelligent classification of components for very effective retrieval of components. In order to be effective, the scheme should be enhanced to meet the interests of the people. The most surprising result for the author was the observation that developers are more likely to browse for personal assets than to search. More research is needed to understand the determining factors in this process. Work could also be done determining the effect of integrating the reuse environment more tightly with software developer’s development environment.

Genetic algorithms can also be combined with other soft computing techniques (Neural Networks, Ant Colony Optimization) to get more optimized result.

**BIBILIOGRAPHY**

[1] P. Niranjan, C V Guru Rao “A Mock up tool for software component Reuse Repository” IJSE, vol 1, no 2, April 2010.

[2] Ruben Prietzo-Diaz, “Implementing Faceted classification for software Reuse”, Communication of the ACM, VOL.3, NO.5 May 1991.

[3] Title: ”Reuse Repository” Ewa Stemposz, Alina Stasiecka, Kazimierz Subieta Polish—Japanese, Wersaw.

[4] William B.Frakes and Thomas. P.Pole, “An Empirical study of Representation methods for Results Software Components “, IEE Transaction on Software Engineering vol.20, no.8. 1994, pp.617-630.

[5] “Intelligent Classification and Retrieval of Software Components”, Andreas S. Andreou, Dimitrios G. Vogiatzis, George A. Papadopoulos*,* Department of Computer Science, University of Cyprus, 75 Kallipoleos Str, P.O.B. 537, CY-1678 Nicosia, Cyprus.

[6] Jeffery S. Poulin and Kathryn P. Yglesis “Experiences with Faceted Classification schema in a large Reusable Software Library (RSL)”,In the seventh Annual International Computer Software and Applications Conference(COMPSAC’93),1993,pp.90-90.

[7] Specification, Design and Implementation of a Reuse Repository, 31st annual international COMSAC 2007, IEEE Transactions on Software Engineering, 2007.

[8] Ruben Prietzo-Diaz, “Implementing Faceted classification for software Reuse”, 1900 IEEE, pp.300-304.

[9] R. Prietzo-Diaz and P.Freeman, “Classifying Software for Reuse”, IEEE Software 1987, Vol.4, No.1, ppp.6-16.

[10] Dr.C.V.Guru Rao, P.Niranjan “A Multilevel Representation of Repository for Software reuse “International Journal of Computer Science and Information Security”, Vol 9, No 9, September 2011.