The Recommendations System for Source Code Components Retrieval

[CODEGO]

**Project Proposal**

****

**Supervisor**

Mam Nazia Bibi [PHD Scholar]

**Submitted by**

1st AYESHA JAVED [30]

2nd AYESHA MALIK[12]

3rd MAHNOOR ALTAF[44]

4th KIREN RIASAT[21]

**Department of Computer Science,**

Bilquis post Graduate College for Women[BPGCW], RWP.

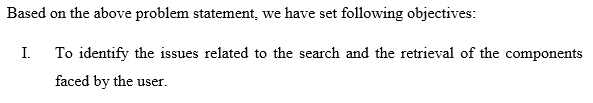
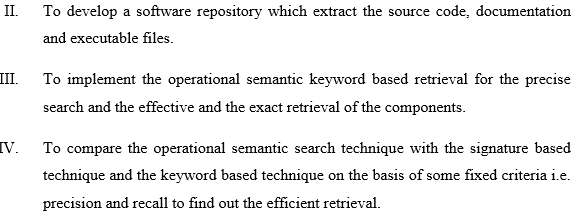
[1st October, 2019]

1. Introduction

Contemporary software development processes involve ﬁnding reusable software components from online repositories and integrating them to the source code, both to reduce development time and to ensure that the ﬁnal software project is of high quality. Although several systems have been designed to automate this procedure by recommending components that cover the desired functionality, the reusability of these components is usually not assessed by these systems. In this work, we present QualBoa, a recommendation system for source code components that covers both the functional and the quality aspects of software component reuse. Upon retrieving components, QualBoa provides a ranking that involves not only functional matching to the query, but also a reusability score based on conﬁgurable thresholds of source code metrics. The evaluation of QualBoa indicates that it can be eﬀective for recommending reusable source code.

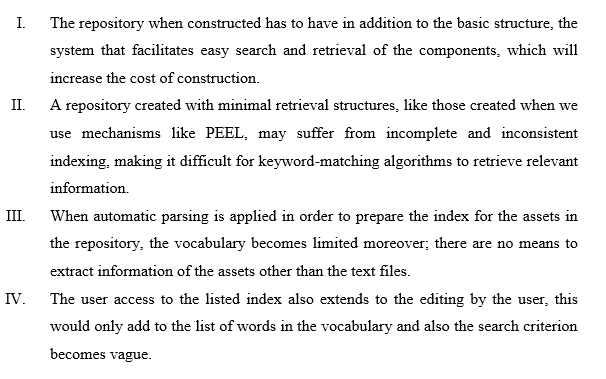
1. Objective

"*To design a real-time Reusability-aware Recommendations system for Source Code Components”.*

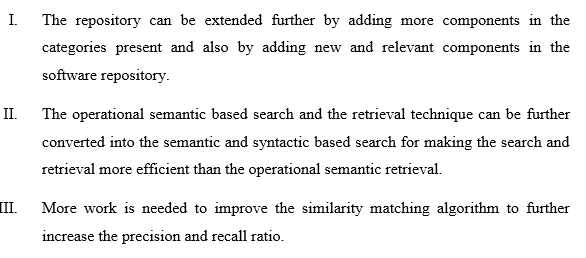
1. Problem Description

Lately, the rise of the open source community and the introduction of online source code repositories have provided numerous exploitation possibilities in the context of software reuse. Developers often rely on ﬁnding reusable source code components, both to reduce the time spent to develop them and to ensure that the resulting software is of high quality (in terms of reliability and functionality delivered).



**4.** Project Scope

In Code Recommendation system we proposed to do the Online source code repositories and question answering communities, such as GitHub or Stack Overﬂow, have facilitated the task of ﬁnding suitable (with respect to a developer query) source code and help the users to get correct and accurate **recommendations.**

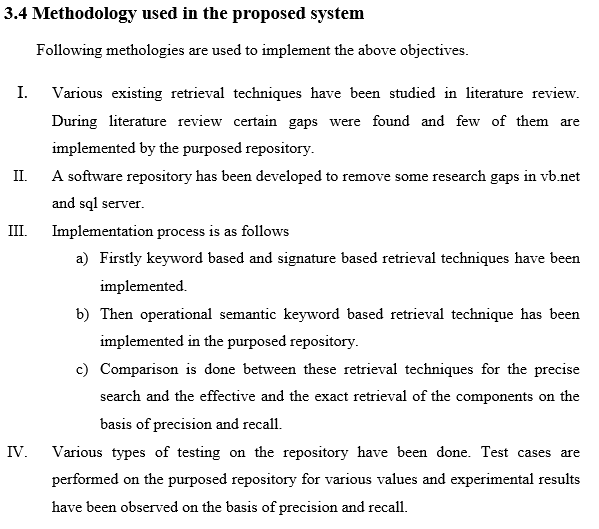


**5. Approach:-**

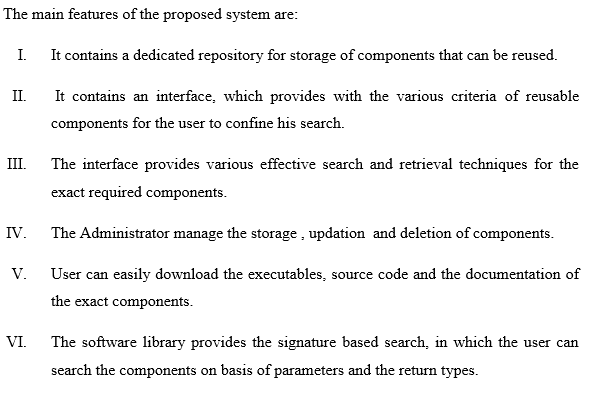
we first introduce the overall process of our approach.

* Crawling data from Github,Stackoverflow/web crawling
* Extracting data means source code according to user-query.
* Learning a Ranking Schema
* Ranking Candidate Code Examples for new Queries

**6. Methodologies**



**7.Features of Proposed Software Repository**



**8. System Diagram**



**9.**Solution Application Areas

Source code examples are used by developers to implement unfamiliar tasks by learning from existing solutions. To better support developers in finding existing solutions, code search engines are designed to locate and rank code examples relevant to user’s queries.

 Recommender systems are 60 a new type of emerging tools proposed to support developers in their tasks. A recommender system for software engineering is defined by Roubillard et al. as " a software application that provides information items estimated to be valuable for a software engineering task in a given context "Robillard et al. (2010).

[Recommendation] systems are software applications that aim to support users in their decision-making while interacting with large information spaces. They recommend items of interest to users based on preferences they have expressed, either explicitly or implicitly. The ever-expanding volume and increasing complexity of information […] has therefore made such systems essential tools for users in a variety of information seeking […] activities. [Recommendation] systems help overcome the information overload problem by exposing users to the most interesting items, and by offering novelty, surprise, and relevance.

**10.**Tools/Technology

* A **web crawler** (also known as a **web spider** or **web** robot) is a **program** or automated script which browses the World Wide **Web** in a methodical, automated manner
* Regression (predictive) Association Rule Discovery (descriptive) **Classification** (predictive) Clustering (descriptive) and there are many more
* **feature extraction techniques** include: Histogram of oriented gradients (HOG) Speeded-up robust **features** (SURF) Local binary patterns (LBP)

**Handy Coding Tools**

* GitHub.
* Twitter Bootstrap.
* Stack Overflow.

**11.**Expertise of the Team Members

# All the team member have the area of interest and basic knowledge of all the techniques that were used in our project and also have great collaboration to our team members.

AYESHA JAVED:-web crawling from different sources github,stackoverflow.

AYESHA MALIK:-Classification of Code examples.

KIREN RIASAT :-Ranking of the different source codes on the basis of the developers aspects of quality and efficiency based.

MAHNOOR ALTAF:-Tester for creating different test case of the Validity of the source code.

.

**12.** References

1. An exploratory study on Android Apps by Rabe Abdalkareem ,Emad Shihba ,Juergen Rilling
   1. Data driven Analysis of software Lab , Concordia University ,Monteal , Canada
2. Learning to rank code examples for code search engines Haoran Niu1·Iman Keivanloo1·Ying Zou1
3. A Semantic-Based Approach to Component Retrieval Vijayan Sugumaran Oakland University Veda C. Storey Georgia State University

