|  |  |  |
| --- | --- | --- |
| **Source Code Recommendation (SCoRe)** | | |
| Team Information | 1.Elliott Campbell  2.Tristin Cory  3.Billy Janson  4.Tyler Medlin  5.Conor Russell  6.John Zickafoose | |
| Document Type: Software Requirement Specification v2.0 | | *Last updated on*: 02-13-2019 |

**Document Approval**

This Software Requirements Specification has been accepted and approved by the following stakeholders:

|  |  |  |
| --- | --- | --- |
| **Printed Name** | **Title** | **Signature and Date** |
|  |  |  |
|  |  |  |
| R. IQBAL | Customer/Course Instructor |  |

**Table of Contents**

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 References 1

1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 Functional Requirements 3

3.2 Non-Functional Requirements 4

3.3 Design Constraints 6

3.4 External Interface Requirements 6

3.4.1 User Interfaces 6

3.4.2 Hardware Interfaces 7

3.4.3 Software Interfaces 7

3.4.4 Communications Interfaces 8

3.5 Logical Database Requirements 8

3.6 Other Requirements 8

4. Change Management Process 9

Appendices 9

# 1. Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

## 1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the “Source Code Recommendation” (SCoRe) software. It will detail all of the features to be in the final product. It will also explain hardware and software constraints, and how it would work with the different interfaces. The main purpose of this document is to be shown to the customer for approval, and then used as a blueprint for developing.

## 1.2 Scope

The Source Code Recommendation system will allow users to perform a requirement search in english text of a database of previously developed software project source code. By giving the input of software requirements, either partial or full, the system will perform a query of the DB. The query will then return source code and the related software requirement text in order of relevance to the provided input if any exist within the DB.

Software engineers will be able to retrieve code examples or snippets in a timely manner and increase productivity. The system could be implemented for personal use or at an enterprise level in order to assist engineers.

The system could be implemented over an internet connection or an intranet connection, depending on the security environment desired, to access the web-based GUI. All source code is maintained in the database, and that database is read to display textual search results to the user on the web-based GUI.

## 1.3 Definitions, Acronyms, and Abbreviations

* SCoRe **–** Source Code Recommendation
* GUI – Graphical User Interface
* UI – User Interface
* DB – Data Base
* Corpus – dataset utilized by the database
* NLTK – Natural Language Toolkit

## 1.4 References

Appendix 1.1 GUI Diagram

Appendix 1.2 Change Request Form

## 1.5 Overview

The remainder of this document includes three chapters and one appendix that detail the requirements and constraints surrounding the SCcRe. Section 2 provides an overview of the general system functionalities. Section 3 provides the requirement specifications in detailed terms and a description of the different system interfaces. Section 4 indentifies and describes the process that change managerment of the project, and when the project scope or requirements change. It also outlines who can submit changes and by what means. The appendix contains diagrams that can assist with visualization and understanding of the project.

# 2. General Description

This section will give an overview of the system. The general workings of the system and basic functionality will be introduced. The potenial users of the system will also be descibed in this section. The constraints and assumptions for the system are introduced here as well.

## 2.1 Product Perspective

Currently retrieving snippets of previously developed source code is a manual process or relies on an inaccurate search process. This project aims to improve the accuracy and efficiency of software development via a GUI-based web application. The application will utilize a database of previously developed source code mapped to keywords and an effective requirement search in english language text search to return search results in order of relevance to developers.

## 2.2 Product Functions

The system will consist of parts: a DB of source code and software requirements, a Python middleware to perform the query, and a web-based GUI to receive user input and display the results of the query.

The returned data will be listed in order of relevance to the input and display the source code snippet and the related software requirement specification description. The DB will accept new source code and requirements documentation that are mapped to specified keywords.

## 2.3 User Characteristics

The primary users of this software are software engineers and software development organizations to be use the product to increase efficiency by decreasing retrieval time for previously developed source code.

## 2.4 General Constraints

The dataset for the DB will consist of Python source code and the related software requirement specifications. For the scope of this project the system will only allow users to perform searches on Python source code.

The computer being used is another constraint of the system. The hardware that executes the search algorithms must have at least 2.4 GHz Intel core i3 or higher; the algorithm may not run at optimum speeds which could slow down the overall calculations. This could lead to low system performance or algorithm failure.

Flask Alchemy and Python must be installed on the computer in order for the product to function in its intended manner.

## 2.5 Assumptions and Dependencies

One assumption of the system is that the computer that is executing the algorithm has at least a 2.4 GHz Intel core i3 or higher to properly run the algorithm. If the system is being ran on less than this, the performance will be decreased as it will not have enough processing power to complete the search in the specified timeframe.

Another assumption of the system is that the computer will have Python and Flask Alchemy installed in order to function in its intended manner.

# 3. Specific Requirements

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

## 3.1 Functional Requirements

1. The system’s user interface will allow for text input

*Source:* Project topic Description

*Priority: 2*

*Introduction:* The system’s user interface will allow for text-based input to be used as search query

*Dependent on:* None

*Inputs:* Text

*Processing:* The text will be sent to the system’s back-end.

*Outputs:* Notification of submission success.

*Error Handling:* The system will notify the user of a failed submission.

1. The system will parse the text into a keyword search query

*Source:* Project topic Description

*Priority:* 2

*Introduction:* The system will parse the text into tokenized keywords to pass to the DB

*Dependent on:* **FR1**

*Inputs:* Text

*Processing:* Keyword recognition and extraction.

*Outputs:* Keywords from original text

*Error Handling:* None

1. The system will search the DB for source code matching the parsed keywords.

*Source:* Project topic Description

*Priority:* 1

*Introduction:* The system’s search engine will utilize a SQL query to find source code that matches the parsed keywords.

*Dependent on:* **FR2**

*Inputs:* Keywords

*Processing:* Keywords will be run through an SQL query

*Outputs:* Source code matching the keywords in JSON format

*Error Handling:* None

1. The system will display the recommended source code snippets to the user interface.

*Source:* Project topic Description

*Priority:* 2

*Introduction:* The system will display the recommended source code to the user interface.

*Dependent on:* **FR2**

*Inputs:* Source code matching the keywords in JSON format

*Processing:* Interpret the JSON file to display search results to user.

*Outputs:* The search results with code snippets, must haves & missing.

*Error Handling:* If the HTTP request fails to retrieve JSON, log information and determine if user interface is available for receiving object. Display related error message.

1. The system will be able to recommend multiple pieces of code per input.

*Source:* Customer Meeting - Razib Iqbal

*Priority:* 2

*Introduction:* The system will be capable of returning recommended code that meet the search keywords of the user’s requirement.

*Dependent on:* **FR3, FR4**

*Inputs:* Keywords

*Processing:* Refer to functional requirements **FR3, FR4**

*Outputs:* Multiple pieces of source code

*Error Handling:* If the system has too many suggestions, choose those that best match the query.

## 3.2 Non-Functional Requirements

1. The system must return results within 2 seconds of a search being requested.

*Source:* Customer Meeting - Razib Iqbal, Mazharul Islam

*Priority:* 2

*Introduction:* The DB will be queried and the output JSON file will be created within 2 seconds of a user search

*Dependent on:* **FR4**

*Inputs:* User text search

*Processing:* The DB will perform the query based on keywords

*Outputs:* JSON file

*Error Handling:* The system will display a time-out message.

1. The system’s user interface will sort the recommended source code results by number of matched keywords.

*Source:* Team Discussion

*Priority:* 2

*Introduction:* The user interface will display the returned recommendations in descending order of matched keywords

*Dependent on:* **FR3**

*Inputs:* None

*Processing:* User interface will interpret and display source code

*Outputs:* Source code snippets formatted for readability

*Error Handling:* Display appropriate error message if unable to parse JSON or if no source code is found.

## 3.3 Design Constraints

**DC1. Programming Language**

*Source:* Project Description

*Description:* Python and its public librarires are used for the implementation of The Source Code Recommendation product.

**DC2. Operating System**

Source: Project Description

Description: SCoRe is designed to be developed and worked on with Microsoft Windows 10 (32/64 bit), Linux, and OS X operating systems. Working on older OS versions and any other OS is not within project scope.

**DC3. Platform**

Source: Project Description

Description: SCoRe is designed to work on any Personal Computer. Mobile devices or smart phones are not within the project scope.

## 3.4 External Interface Requirements

### 3.4.1 User Interfaces

### UIR.1 The system will have a web-based graphical user interface to interact with the system.

*Source:* Team Discussion

*Priority:* 2

*Description:* A graphical user interface will be utilized to facilitate user searches of a connected database; functional components are diagrammed in Appendix 1.1

*Related to:* **FR1**

**UIR.2 The GUI will have a field that allows a user to enter a textual search.**

Source: Team Discussion

Priority: 2

Description: A box that accepts textual user input to be used for a search.

Related to: **FR1**

**UIR.3 The GUI will have a button that allows the textual search to be initiated.**

Source: Team Discussion

Priority: 2

Description: A button will be visible to the user that will initiate the search.

Related to: **FR1**

**UIR.4 The GUI will have a field to display the returned recommended source code to the user.**

Source: Team Discussion

Priority: 2

Description: This field will display the returned recommended source code to the user.

Related to: **FR4**

**3.4.2 Hardware Interfaces**

Not applicable

**3.4.3 Software Interfaces**

**SIR.1**  **SCoRe will utilize a Flask Alchemy framework**

*Source:* Team Discussion

*Priority:* 1

*Description:* Connection to the database will be handled by Flask Alchemy

*Related to:* **FR4**

**SIR2. The Python NLTK library will be used**

*Source:* Team Discussion

*Priority:* 1

*Description:* Tokenization of keywords from search input will utilize this library

*Related to:* **FR4**

**3.4.4 Communications Interfaces**

**CIR.1 The GUI will communicate via a network connection with the DB through the Flask server**

*Source:* Team Discussion

*Priority: 2*

*Description:* The GUI will communication via a network connection (LAN,WAN,etc) with the DB via the Flask server

*Related to:* **FR3**

**3.5 Logical Database Requirements**

**LDR.1 Flask Alchemy will be used to contain the DB.**

*Source:* Team Discussion

*Priority:* 1

*Description:* A Flask Alchemy database will be used to hold the database

*Related to:* **SIR1, FR3**

**LDR.2 The DB will hold the source code, keywords, and mapping metadata tables.**

Source: Team Discussion

Priority: 1

Description: The DB schema will represent the relational characteristics of the dataset.

Related to: **FR3**

**LDR.3 The DB will contain a manual mapped table.**

Source: Team Discussion

Priority: 1

Description: This DB table will contain source code and keywords mapped manually.

Related to: **FR3**

**LDR.4 The DB will contain an automatically mapped table.**

Source: Team Discussion

Priority: 1

Description: This DB table will contain source code and keywords mapped via semantic search.

Related to: **FR3**

# 4. Change Management Process

This section presents our teams process for change management. It is a step-by-step process for changing any aspect of our project’s initial design. This is to provide governance and oversight among all involved parties to ensure satisfaction with the proposed change and to provide record of such.

When a change request is presented either from a group meeting or by a stakeholder it will begin the process by submission of a change reqest form (Appendix 1.2). Once the form is submitted via email or Slack message, the request will be reviewed in a due diligence phase. In this phase the group will enter into a discussion around the feasibility of making the requested change and what requirements may be impacted by the request.

If the change request is accepted the team will revise the requirement or create a new one depending on the scope of the request. Once this is completed we will bring the requirement back to the stakeholder to confirm the teams interpretation of the change is congruent with their vision.

If the stakeholder is satisfied with the change, the final step is to update the SRS document to reflect the change and additonally update the related user stories. The change may potentially affect the time required and completion dates of user stories which will adjust the projects overall velocity.

# Appendices

Appendix 1.1

GUI Diagram



Appendix 1.2

[Change Request Form](https://livemissouristate-my.sharepoint.com/:w:/r/personal/ejc22_live_missouristate_edu/Documents/CSC-450/Forms%20and%20Diagrams/change-request-form.docx?d=w78f7d6965df54c4fadccb718ab0b754d&csf=1&e=QbDoO6)

(link to OneDrive Document)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Change Request** | | | |  | |  | | | | | | |
|  | | | |  | |  | | | | | | |
| Requirement ID (if applicable): | | |  | | | | Change Number: | | |  |  | |
| Requested by: |  | | | | | | | Date: |  | |  | |
|  | | | | | | | | | | | | |
| **Category** | | | | | | | | | | | | |
|  Requirements  Scope  Quality  Schedule | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **Detailed Description of Proposed Change** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **Justification for Proposed Change** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **Comments** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **Approval** | | | | | | | | | | | |
|  Approved  Rejected  Defer | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Justification** | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Comments** | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Signatures** | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Name** | | **Role** | | | **Signature** | | | | | **Date** | |
|  | |  | | |  | | | | |  | |
|  | |  | | |  | | | | |  | |
|  | |  | | |  | | | | |  | |
|  | |  | | |  | | | | |  | |