SOFTWARE REQUIREMENTS SPECIFICATION

For

ShowMe: Related Work

Prepared By: -

Group 6

Software Factory I

Department of Software Engineering

Arizona State University

Academic Year: 2017-2018

Version: 1.1

Preface

This document is designed for people who will be testing or using this application for their own benefits. It describes about the user requirements and functional requirements that was used to design and build the application. Each chapter tells about the functionality of this application and understand the purpose of it. If you are a researcher or a student, then it should help you in your research work, express your views on different papers and benefit from other user's ratings and comments.

Revision History

Date	Revision	Description	Author
11/20/2017	1.0	Initial Version	Abhishek Dutta
11/25/2017	1.1	Final Version	Abhishek Dutta

Table of Contents

1. Introduction

1.1 Purpose

The main objective of this document is to illustrate the functionality and requirements of the ShowMe:ShowMe Related Work application. This application is mainly focused on providing an interface to the researchers for navigating through various research papers which may be related or inter related by defining the relations among them. The application is intended mainly for researchers in various fields who can go through various papers needed for their work. Students can also benefit from this application.

1.2 Scope

The application allows the user to search for a paper by the paper name or its author which is shown in the form of a graphical notation: nodes and edges. The nodes represent the research papers and the edges represent the relation between two papers. Researchers can click on a node to view the research paper. Also, they can navigate to other research papers that are related to the current one by navigating through the edges and clicking on other nodes.

It allows them to upvote/downvote an edge to show the relevance of the other paper in relation to the first paper. They can also comment on a link in case they want to make some notes about the paper or edge. Users need to login using google to reveal their identity before they update anything.

1.3 Definitions, Acronyms, and Abbreviations

SRS	Software Requirements Specification
UI	User Interface
OAuth	Open Authentication
MVC	Model View Controller

Table 1. Acronyms and Terminology

1.4 References

Books

- Software Engineering: Ninth Edition by Ian Sommerville
- The Personal Software Process (PSP SM), November 2000 by Watts S. Humphrey

2. Overall Description

2.1 Product Perspective

This is a web based application implemented in three tier architecture using MVC architecture. The application provides the researchers a platform to read through various research papers with an interactive functionality to comment and vote on the relations of the papers. User needs to authenticate themselves before they can vote or comment on any part. There is another product called "Google Scholar" that is currently available in the market which provides the web links for various papers but it lacks any kind of interactive functionality with the user.

2.2 Product Architecture

The web application is built on a three-layered architecture. In the top tier we have our client-side application which will interact with user. In the middle tier we have our middleware where all the business logic and rules are defined. Bottom tier consists of the database where all the data and relations among data are stored. Below is the system architecture diagram.

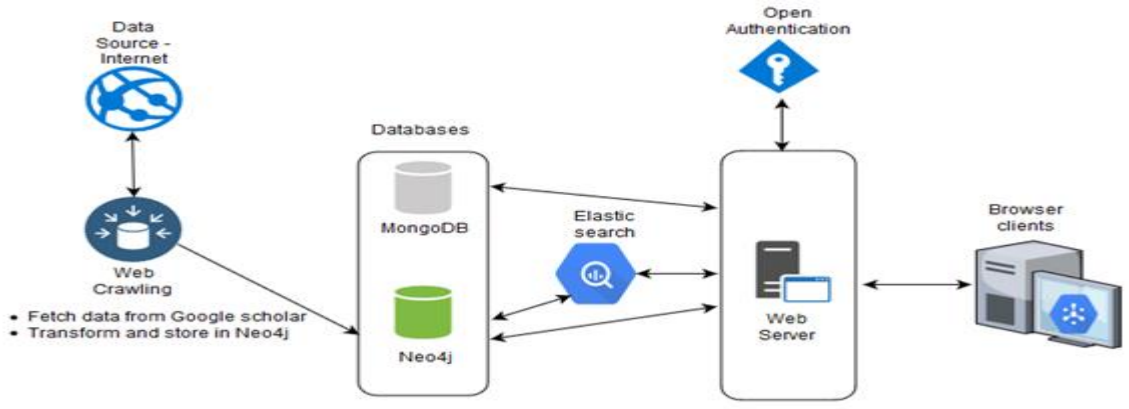

Figure 1. System Architecture

Each of the components are described below individually: -

Internet Data Source: These signify the websites which provide relevant information for the research papers, such as, the paper's DOI, author name, URL to the paper, abstract, references, etc. These websites will act as the source of information for our project. This data can be collected either manually or by using the web crawlers. Since we will need a huge volume of data, we are relying on web crawlers. For the prototype, we have manually collected data for one hundred research papers form Artificial Intelligence and Computer Graphics. Once we have a running prototype, we will be scaling the data using web crawlers.

Web Crawling: Web crawlers are the programs which collect relevant information from websites. We will develop web crawlers using python libraries - beautiful soup and scrappy, since we have some experience with using these libraries. Crawlers will fetch all the key features of the papers like - paper's DOI, author name, URL to the paper, abstract, references, etc. These crawlers will be hosted on robust systems where we can keep them running for long hours.

MongoDB: MongoDB is a document store NoSQL database. Since, our data will not have a defined schema and it will be growing at an undefined rate, we chose to use a NoSQL database over a relational database. We are using MongoDB to store user authentication information along with the user comments and upvotes/downvotes for the relationship between two papers.

Neo4J: Neo4J is a NoSQL graph database. The essential components in our project which consists of the papers and the relationship between them, exhibits a graph relation and hence we decided to use this graph DB. We are using Neo4J to store information about papers along with the relationship between them. Papers are getting stored as nodes and the relation as the edges of a graph. Initially, we were planning to use only one database (Neo4J) for storing all the information but, on further research we found out that this approach would hamper the performance of the application. Storing the user comments, upvotes/downvotes and the user authentication information would make the Neo4J dB heavily loaded with data and slow down the performance. So, we decided to use Mongo DB for storing the user comments, upvotes/downvotes and user authentication information.

Elasticsearch: Elasticsearch is an open source search engine. We are using it to facilitate the searching of papers.

Open Authentication: Open Authentication (OAuth) allows users to use their existing account like google, GitHub, yahoo etc. for authentication rather than having to create a new account. As of now, we are allowing users to create an account on our app and we are using the same account for authentication purposes. Once we have a running prototype, we will further enhance it to accommodate Open Authentication.

Web Server: Web server will host the application.

Clients browser: Clients can access the application using their browser.

2.3 Product Functionality

The application provides a list of user interactive features for better experience for researchers thereby saving them a lot of time and manual effort. Below are the set of functionalities that the application offers to its end users: -

a) Users can read through different papers by clicking on the nodes.
b) Users can navigate back and forth to other papers through the links to different nodes.
c) Users can upvote/downvote the links between different papers depending on the relevance and quality.
d) Users can comment on the links to justify their vote or just to add a note for other users to benefit from it.
e) Users can search for a paper.
f) Users can search papers categorized based on field of study.

3. Specific Requirements

3.1 User Requirements

From a user's perspective, the application should be able to visualize relations among papers as graphs with each node representing a paper and a directed edge between two nodes if one paper cites another. Users should see the paper if the user clicks on the node and relation properties if they click on the edge. Furthermore, they should be able to make an edge weak or strong depending on the relevance and context.

3.2 Functional Requirements

The main functional requirements gathered from the sponsor were the application should be able to visualize the papers and their relations in graphical format in the UI side. It should allow the users to navigate through the papers and enhance it by making it more interactive. Users should have the option to vote and comment on any of the edges so that they can other users can benefit it at a later stage. The application should also have search functionalities to search for a specific paper.

3.4 Non-Functional Requirements

The non-functional requirements specified by the sponsor was mainly more focused towards the look and feel of the UI using different visualization libraries. Other requirements included that valid user profiles needs to be associated with whoever wants to vote or comment. Others can view the papers without authenticating themselves. Application should not maintain any user passwords in the Database.

3.5 Software Requirements

The application is built using following tools / Software. It is required to have these software configured on the system.

- MongoDB(3.4.10)
- Neo4J (3.4.0)
- NodeJS (8.9.1)
- Vue.JS (2.0)
- OAuth (2.0)
- Cytoscape

3.5 Hardware Requirements

A standard computer system should be able run this application easily. Any system with below minimum configurations are sufficient to host it.

- 4 GB RAM
- X86 or x64 bit Windows/Linux/iOS Operating system
- 50 GB Hard drive (Depends on the usage to store the data)
- 2.1 GHZ CPU