ENG 3050 – Software Engineering Design I

Winter 2015

Artificial Intelligence Literature Searching

Assignment 1 – Software Requirement

Group #3

Contributing Members:

John Simko, Brandon Stanley, Shahood Mirza

Date Submitted: 1/19/2015

Approval

This document has been read and approved by the following team members responsible for its implementation:

Print Name	Signature	Comments
John Simko		

Print Name	Signature	Comments
Brandon Stanley		

Print Name	Signature	Comments
Shahood Mirza		

Revision History

Date	Author	Comments
Jan 15th	John Simko	Created Requirements document, title page, Revisions matrix.
Jan 15th	Brandon Stanley	Added Functional Requirements and CASE Tools
Jan 18th	John Simko	Added GUI mock-ups. Added references
Jan 19th	Brandon Stanley	Added Use Case diagram
Jan 19th	Shahood Mirza	Project Scope/Objectives, basic non-functional requirements
Jan 19th	John Simko	Reorganized information. Started Domain Analysis
Jan 19th	Brandon Stanley	Added to Domain Analysis, added to non-functional requirements, minor font modifications.
Jan 19th	Shahood Mirza	Added content to basic non-functional requirements headings
Jan 22nd	John Simko	Fixed title page, reorganized according to assessment. Notated further changes to be made (glossary, use case description, functional req's description)
Jan 26th	Brandon Stanley	Remade Use Case Diagram, added use case descriptions, added clarification to functional requirements, added keyword definitions, modified table of contents.
Jan 29th	Brandon Stanley	Changed CASE documentation tool from Sandcastle to Doxygen
Feb 08th	Brandon Stanley	Removed browse use case and added next/previous page use case
Feb 09th	Brandon Stanley	Modified use case to include add and remove login info

Table of Contents

1. Introduction

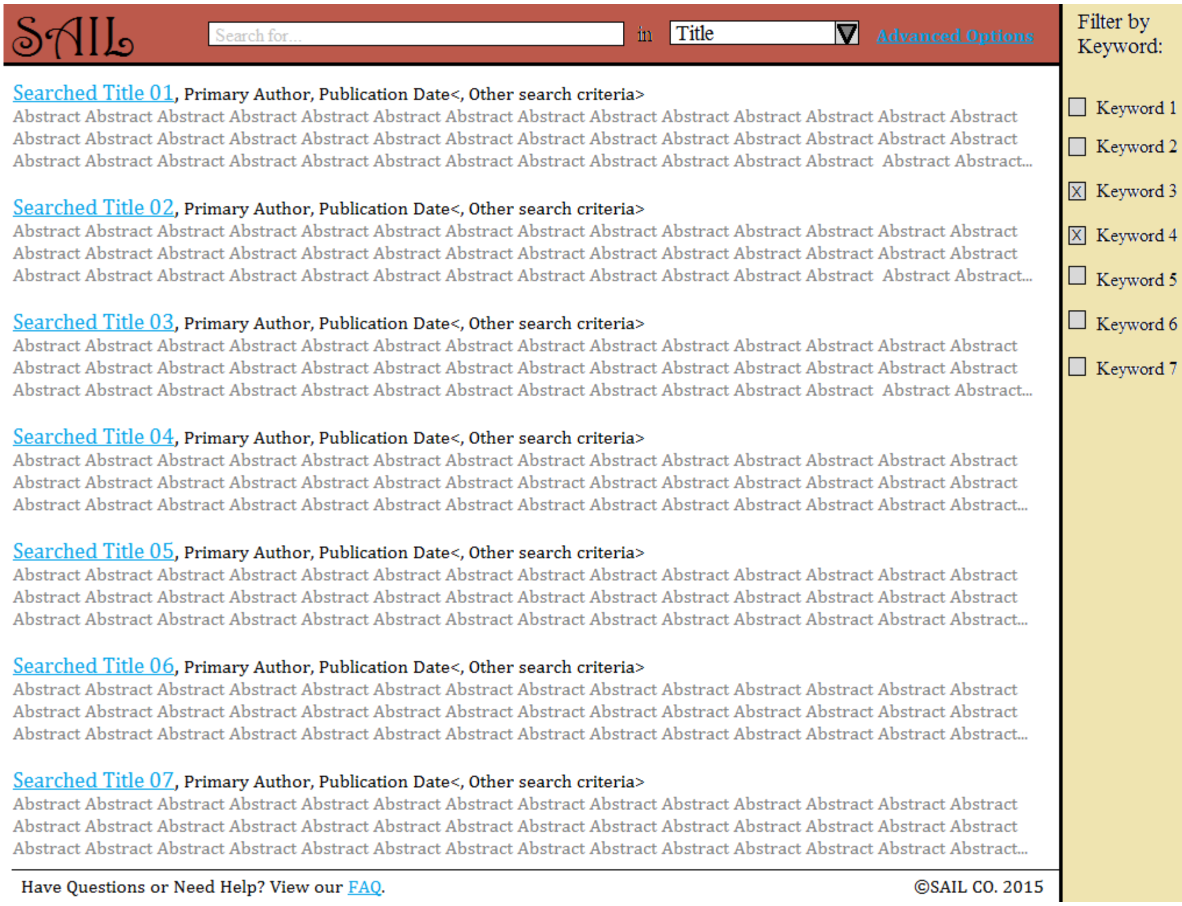
1.1 Objective

To design a user-friendly database of articles and resources based on artificial intelligence that will allow users to locate documents based on custom criteria.

1.2 Scope

The software will be designed to provide convenient access to a large database of resources. Certain features will be allowed through various access permissions which will allow the user to either manage the software or browse through the available information. After logging in, each user will be able to search, sort/filter and browse the database according to their needs. Administrators will have permissions to add new user accounts and will have access to add, remove and edit resources from the database.

1.3 User Interface



2. Domain Analysis

2.1 General Knowledge of Domain

- Each resource has a Title, Publication Year, and Author.
- Searches can be done on each of the above attributes.
- Most resources include an abstract, which briefly describes itself.
- Many resources have a more specific publication date.
- Most resources have Author supplied keywords

2.2 The Environment

- All users have a personal computer.
- Users are typically running Windows based OS on their PC.

2.3 Customers and Users

- AI research specialists.
- Corporations.
- Hobbyists.
- System administrator.

2.4 Competing Software

- IEEEXplore
- Google Scholar

2.5 References

- http://ieeexplore.ieee.org/Xplore/home.jsp
- https://scholar.google.ca/
- http://library.lakeheadu.ca/

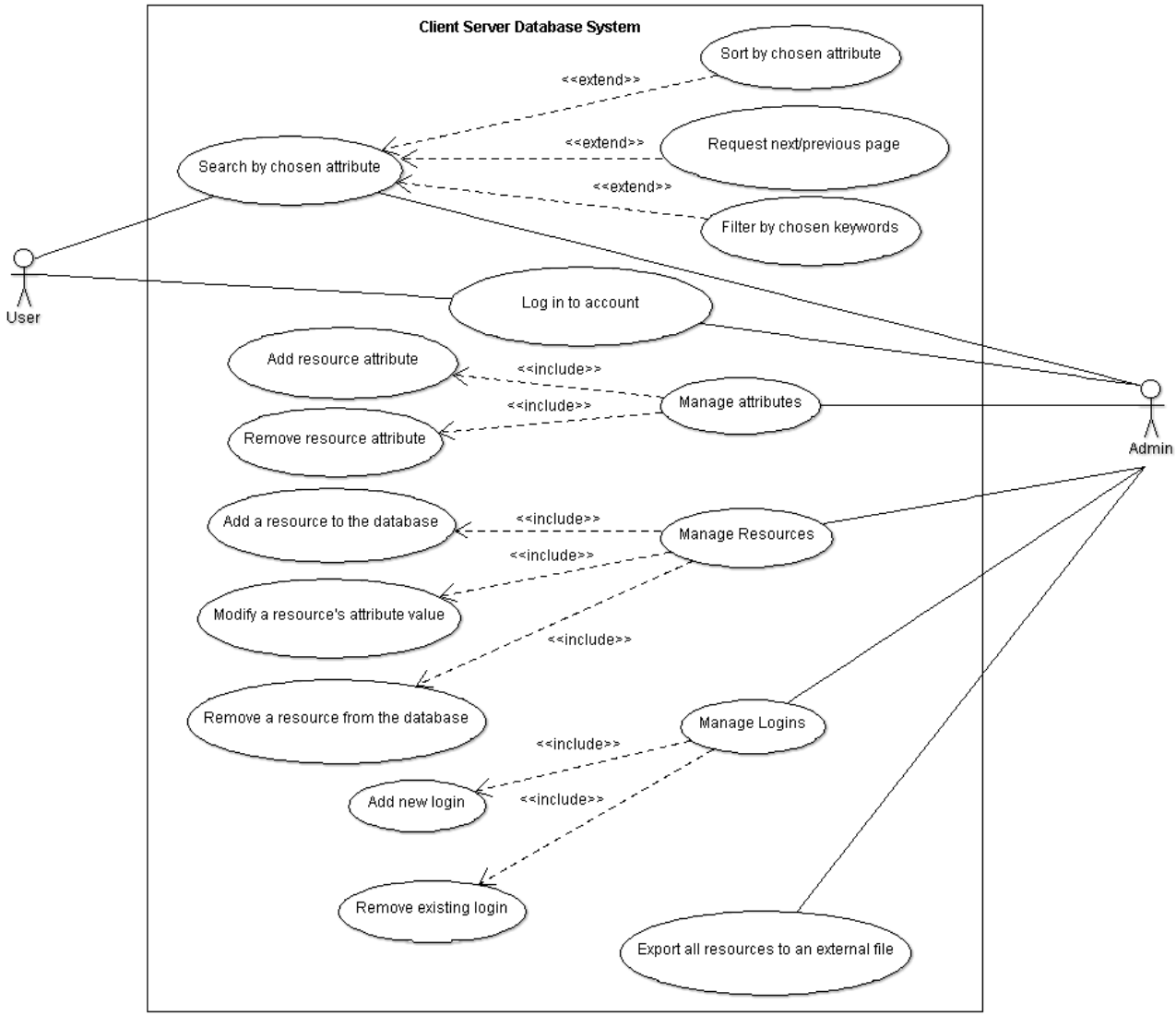
3. Specific Requirements

3.1 Functional Requirements

- User/Admin must be able to search for resources by the author, title of work, publication company, year of publication, abstract, keywords, and any admin added attributes.
- User/Admin must be able to sort search results by author, title of work, publication company, and year published, and any admin added attributes.
- User/Admin must be able to filter out search results that do not contain user/admin selected keywords.
- User/Admin must log in using a username and password to use the software.
- Admin must be able to add a new resource, modify attributes of an existing resource, and delete existing resources.
- Admin must have the ability to add a new attribute and to delete existing attributes common to any resource
- Admin must be able to export all resources and relevant data to an arbitrary file

3.2 Use Case Model

3.2.1 Use Case Diagram



3.2.2 Use Case Description

Search by chosen attribute:

A user/admin can search for any attribute that belongs to the resources. Attributes include Title, author, publication company, date of publication, abstract, key words, and any admin added attributes. The server responds with a page of resources, which is a small subset of the search results.

Sort by chosen attribute:

A user/admin can sort all search results by any given attribute in either ascending or descending order. Attributes include title, author, publication company, date of publication, and any admin added attributes.

Filter by chosen keywords:

A user/admin can filter all search results by the keywords that appear in the current set of search results. This hides all results that are missing the selected keywords.

Request next/previous page:

A user/admin can request the server for the next/previous page of resources.

Log in to account:

A user/admin must log in to an account before they are able to access any resources that exist on the server.

Add resource attribute:

An admin is able to add an attribute to all resources in the database.

Remove resource attribute:

An admin is able to remove an attribute from all resources in the database.

Add resource to the database:

An admin is able to add a resource to the list of resources in the database.

Modify a resource in the database:

An admin is able to modify the attributes of any resource that exists in the database.

Remove a resource in the database:

An admin is able to remove any resource that exists within the database.

Add new login

An admin is able to add a new user or admin login account.

Remove existing login

An admin is able to remove an existing user or admin account.

Export all resources to an external file:

An admin is able to export all the resources in the database to an arbitrary external file.

3.3 Non-Functional Requirements

3.3.1 Performance

Most server queries will not involve large amounts of data as users will primarily search using specific keywords. As a result, performance should not be hindered by general software usage. In order to alleviate any excessive requests, recently searched items will be locally cached for user convenience.

A multithreaded setup will be implemented. This will allow efficient use of resources while multiple users are simultaneously logged into the system.

3.3.2 Reliability

The software is designed to be highly robust. Features will be made user-friendly and easy to learn. Duplicate data will be automatically managed to make efficient usage of available system resources. In the case that users enter erroneous data, notification messages and/or feedback will be provided for assistance. The product will incorporate checks to ensure correct execution and will be thoroughly tested for reliability.

3.3.3 Backup

If a data recovery is required, a constant backup of resources will be kept in order to restore updated functions. The database administrator will have access to export all the data currently stored at any given time.

3.3.4 Availability

Apart from scheduled maintenance hours, servers will be monitored and kept online for users to browse and search through. Communication links between client and server will be implemented using socket programming via TCP sockets. The server will be made available to multiple concurrent connections.

Availability constraint – due to the use of Lakehead University's network, the client-server connection will rely on the uptime of the University's system. As a result, we cannot guarantee connectivity during certain network outages.

3.3.5 Extensibility

The database and functions will be extensible in case of future enhancements. General search/filter functions will be dynamic and will be able to function if other engineering topics are added to the database resources. Basic functions and UI elements will allow for scalability in terms of adding larger amounts of content to the database. Through the use of Object-Oriented design methodologies, many functions will be designed with reusability and extensibility in mind.

3.3.6 Security/Privacy

Database administrators will have sole privileges to manage user access. This will prevent unauthorized access to the database from unregistered users. Each user will have unique credentials and will be required to login to the database to utilize its functions. User passwords will be hashed and stored in a database to prevent unauthorized theft and maintain privacy.

The client-server connection will be implemented using TCP with SSL to establish an encrypted link between endpoints. Ultimately, this will provide an extra layer of privacy for users during any data transfer such as logging in or completing search queries.

3.3.7 Maintainability

Software will be coded to ensure seamless testing and future maintenance. Documentation will be done both internally and externally throughout the phases of the project. Consistent formatting and standards will be used allowing contiguous modifications as necessary.

Class diagrams, use case diagrams and software design methodologies will be utilized to ensure maintainability of the product and compliance with industry design standards.

3.3.8 Safety

Since the software does not control any physical components, it does not pose any safety concerns for clients using its features.

Please refer to section 3.3.6 (Security/Privacy) in regards to privacy and confidentiality of user information.

4. CASE Tools

ArgoUML

UML Diagram design tool. Used to design all diagrams for documentation.

Subversion

Source control tool. Manages revisions of code done by project members.

PostgreSQL

Database used to store all data for server. This database is read from on server startup, and stored in local memory.

Visual Studio

IDE used for all code development for this project.

Doxygen

Automated document generation tool. Generates documents from code comments.

5. Glossary

Artificial Intelligence:

the theory and development of computer systems able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

Artificial Intelligence Literature:

literature pertaining to the subject of artificial intelligence, including standards, articles, papers, and journal publications.

Hash (Password):

A hash algorithm is a function that converts a data string into a numeric string output of fixed length.

IDE:

Integrated development environment. A software tool used by computer programmers to develop and compile code.

Keyword:

A common term which has been identified as relevant to the contents of a resource.

SSL:

Secure Sockets Layer is a standard of web security for the transfer of encrypted data between two computer systems.

TCP/IP:

The Transmission Control Protocol and Internet Protocol are the standard for ensuring error free data transmission from one computer system to another over the internet.

UML:

Unified Modeling Language. Used to aid in the development of software through the development of easy to read diagrams.