Software Requirements Specification

for

The SolR Front-End Project

Version 1.0

Prepared by

|  |  |
| --- | --- |
| Corey McCandless | CoreyMcCandless@letu.edu |
| Israel Terrill | IsraelTerrill@letu.edu |
| Joel Burdette | JoelBurdette@letu.edu |

|  |  |
| --- | --- |
| Instructor: | Dr. Brent Baas |
| Course: | Software Engineering II |
| Date: | 1 October 2016 |

|  |  |
| --- | --- |
|  |  |

Contents

Contents ii

Revisions iii

1 Introduction 1

1.1 Document Purpose 1

1.2 Product Scope 1

1.3 Intended Audience and Document Overview 1

1.4 Definitions, Acronyms and Abbreviations 1

1.5 Document Conventions 1

1.6 References and Acknowledgments 1

2 Overall Description 2

2.1 Product Perspective 2

2.2 Product Functionality 2

2.3 Users and Characteristics 2

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 3

2.6 User Documentation 3

2.7 Assumptions and Dependencies 3

3 Specific Requirements 4

3.1 External Interface Requirements 4

3.2 Functional Requirements 6

3.3 Behaviour Requirements 7

Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
| --- | --- | --- | --- |
| 1.0 | Corey McCandless  Israel Terrill  Joel Burdette | This is the initial version of the SRS | 00/00/16 |

# 

# Introduction

## Document Purpose

This document describes the requirements for the LeTourneau University email log SolR-accessible database web interface. The purpose of this document is as a reference for those reviewing or developing this project in the future in order to give them a fuller understanding of the requirements and decisions made for this project. As the first specification for the development of this particular project, this document is the starting point of documentation related to this project.

## Product Scope

Currently, LeTourneau’s Information Technology department stores the emails that it receives in a database accessed through Apache SolR. Although IT has a REST API, they do not have a UI for the database that intuitively and easily allows them to access the emails that they have stored. This project attempts to create a UI for this database that will meet LeTourneau IT’s needs.

## Intended Audience and Document Overview

The intended audience or this specification document is LeTourneau’s Information Technology department. Expected users of this document include future developers attempting to fix bugs or ascertain why certain design decisions were made. Those maintaining or modifying the documentation of the project are expected to read this document to keep things consistent. Additionally, those reviewing this project for quality assurance are expected to read this document.

## Definitions, Acronyms and Abbreviations

**ReactJS**:an open-source UI library written by the developers of Facebook

**SolR**: an open-source search-engine based on Lucene (part of the Apache project) searches the database

## Document Conventions

Document Conventions:

* Font type: Arial
* Font Size: 11pt

## References and Acknowledgments

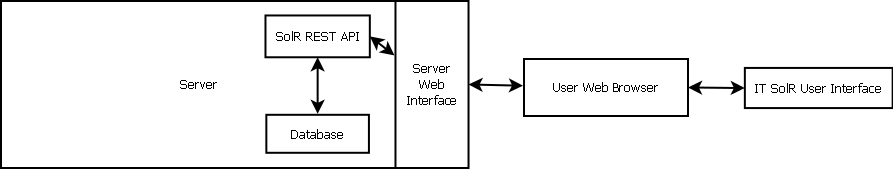
<http://lucene.apache.org/solr/>

<https://facebook.github.io/react/>

# Overall Description

## Product Perspective

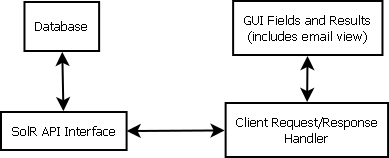
This product is a totally new, self-contained product. As a user interface, this product will interact with the REST API of the SolR interface in order to deliver content to IT personnel in a more aesthetically pleasing way than formerly existed.



**Figure 2.1**

## Product Functionality

Display a user interface for the SolR framework to:

* Allow for advanced input searches
* Allow for easier general search and drill-down search functionality
* Allow for a view of the emails contained in the SolR-accessible database
* Allow for easy access of email attachments

**Figure 2.2**

## Users and Characteristics

The user most likely to use this product is the information technology department of LeTourneau University. Most of these users who will be accessing this system will be experienced in technical disciplines. They may use this front-end frequently in order to access the emails sent to this server.

## Operating Environment

This underlying system of the front-end will exist on an Apache server that will connect to and give the clients the needed html and Javascript code. Since this application will run in a web browser, this application will run on most platforms that support a standard web browser (such as Chrome/Chromium and Mozilla Firefox). However, this application will be targeted mostly at desktop and laptop devices rather than the mobile platform, which will be reflected in the layout of the front-end.

## Design and Implementation Constraints

* Web platforms have limited languages and capabilities available to use for development
* A lightweight design was needed, so some frameworks had too much overhead
* The platform had to be compatible with Apache
* The language used had to be something that LeTourneau Information Technology can maintain in the future
* The framework that used had to be compatible with a REST API to connect to a SolR framework
* The platform has to be able to be secured in the future

## User Documentation

To use this product, a user will only require a basic user manual (probably online via webpage) that outlines the purposes of the different fields. In addition, a basic overview of the syntax that can be entered into the fields and some basic directions related to the what options are available for filtering and viewing are the only things necessary.

## Assumptions and Dependencies

We are assuming that one or two IT specialists will be running/monitoring this app throughout the day.

We assume the rights to use open-source projects (who have clearly marked their work as free to use) as guidelines for further development of this application.

Finally, we can assume that users will have a working knowledge of SolR constructs and how to query the database as is.

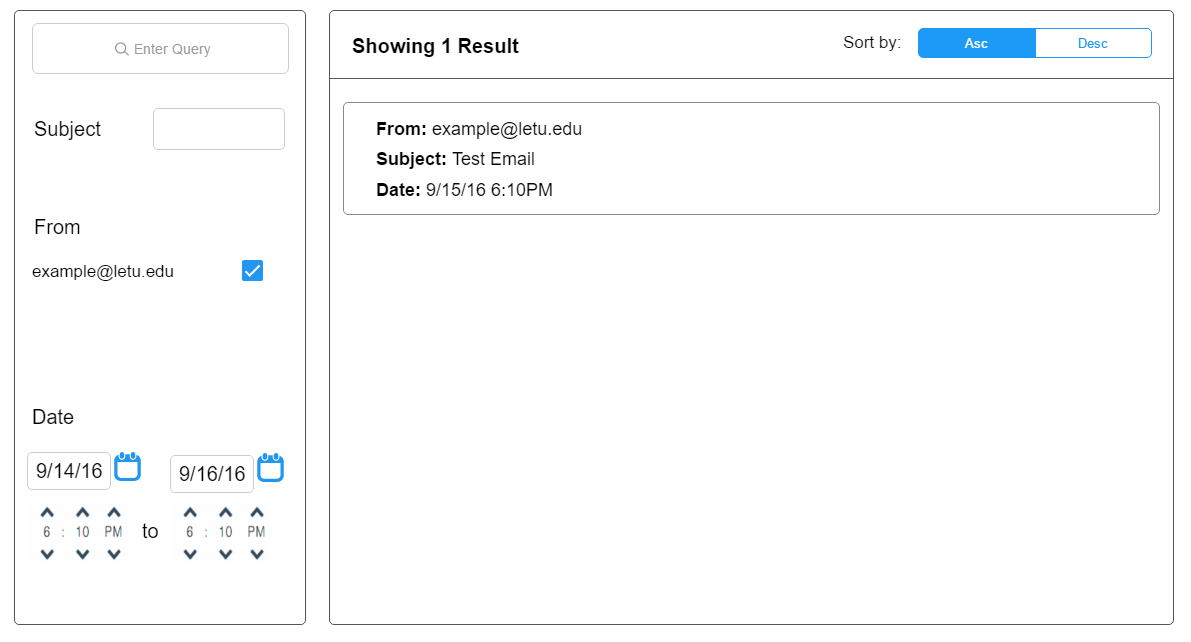
# Specific Requirements

## External Interface Requirements

### User Interfaces

**Listing view-** The user will be greeted by and spent most of their time using the app on the listing view. This view has three major components:

1. The search bar
2. The results section
3. The facet section



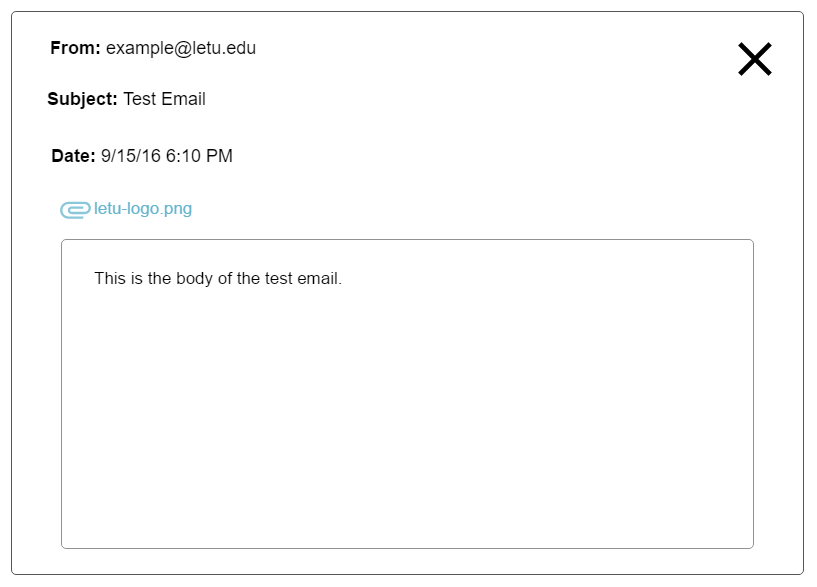
**Figure 3.1**

The search bar is a simple but crucial element in the UI. This is where the user will enter their initial query and the beginning of almost every use case.  As such it has been placed at the top left corner of the view to be used first. It needs to be wide enough to fit an entire average query, and will support basic SolR syntax.

The results section will take up most of the listing view and contains all of the results received from the query.  Each result listing will possess the format of displaying the sender, email subject, and date received.  Results are dependant on the initial query as well as the facet section.

The facet section exists under the search bar and within the left column of the listing view.  This handy tool contains selectors that allow the user to fine tune their initial query.  For example, pre populated sender checkboxes appear based on the results retrieved from the initial query.  Unchecking these boxes will result in an updated results section where the previously displayed results from the unchecked sender will now be omitted.  The date range selector at the bottom of the facet section and the subject specifier near the top will work in a similar fashion-restricting displayed results to the user-defined date range and subject(s).

**Email view-** If the user clicks on a result within the results section of the listing view, the selected email will populate the screen with all relevant information present as shown below.



**Figure 3.2**

Users will be able to see all information that was displayed in the results section for the email, as well as the body of the email along with any attachments linked to the email.  Clicking the “X” in the top right corner will return the user to the listing view.

### Hardware Interfaces

The hardware interacting with our app will be restricted to the user’s device, the cs-lab server where our app is stored, and the SolR server itself.  While this app will be optimized for desktop use; it should be able to run fine on mobile.

### Software Interfaces

The database used to store and query emails is a SolR-accessible database.  Naturally, communicating effectively with the SolR-accessible database will be the top priority in completing the app. Alongside this, the app will live on the cs-lab server as a web app, so any basic browser and internet connection will be required for use.  Web framework of choice for this app is ReactJS.

### Communications Interfaces

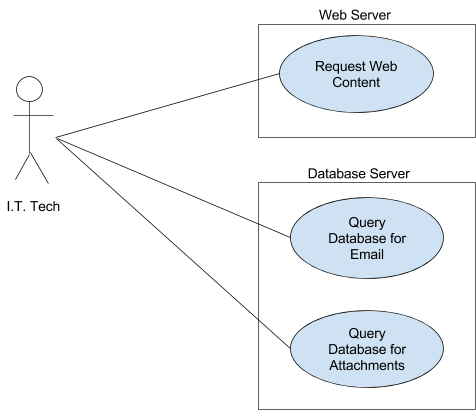
The project is a web app, so the HTML/Javascript content will be served upon HTTP request to a server. The served content will take query information from the user and send requests to a database server (HTTP or HTTPS) using a REST API.

## Functional Requirements

* Database Querying
  + Query database with user-provided parameters
  + Display results received from database
  + Support query (or query-like) syntax in primary search box
  + Provide appropriate controls (faceted checkboxes, sliders, or text box) for filtering
  + Allow user to sort displayed results (ascending or descending) by a predefined list of fields
  + Displayed results can be rendered individually in email format
* Email View
  + Query database for attachments associated with selected email
  + Display relevant email information in format similar to standard email clients
  + Link to attachments (if any)

## Behaviour Requirements

### Use Case View



**Figure 3.3**