Surrogate City Finder Tool

Software Requirements Specification

Version 1

14/09/14

**Team 46**: Akanksha, Arushi Vashist, Ayush Tripathi, Dhruv Das, Siddharth Bhatore

Prepared for

Dr. Vishal Garg

ICS 261—Structured Systems Analysis and Design

Instructor: Prof. Raghu Reddy

Monsoon 2014

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 14/09/14 | Version 1 | Team | SRS version 1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Approval

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | <Your Name> | Lead Software Eng. |  |
|  | A. David McKinnon | Instructor, CptS 322 |  |
|  |  |  |  |

**Table of Contents**

**Revision History** [**ii**](#id.gjdgxs)

**Document Approval** [**ii**](#id.30j0zll)

**1. Introduction** [**1**](#id.1fob9te)

1.1 Purpose [1](#id.3znysh7)

1.2 Scope [1](#id.2et92p0)

1.3 Definitions, Acronyms, and Abbreviations [1](#id.tyjcwt)

1.4 References [1](#id.3dy6vkm)

1.5 Overview [1](#id.1t3h5sf)

**2. General Description** [**2**](#id.4d34og8)

2.1 Product Perspective [2](#id.2s8eyo1)

2.2 Product Functions [2](#id.17dp8vu)

2.3 User Characteristics [2](#id.3rdcrjn)

2.4 General Constraints [2](#id.26in1rg)

2.5 Assumptions and Dependencies [2](#id.lnxbz9)

**3. Specific Requirements** [**2**](#id.35nkun2)

3.1 External Interface Requirements [3](#id.1ksv4uv)

*3.1.1 User Interfaces 3*

*3.1.2 Hardware Interfaces 3*

*3.1.3 Software Interfaces 3*

*3.1.4 Communications Interfaces 3*

3.2 Functional Requirements [3](#id.2jxsxqh)

*3.2.1 <Functional Requirement or Feature #1> 3*

*3.2.2 <Functional Requirement or Feature #2> 3*

3.3 Use Cases [3](#id.3j2qqm3)

*3.3.1 Use Case #1* [*3*](#id.1y810tw)

*3.3.2 Use Case #2* [*3*](#id.4i7ojhp)

3.4 Classes / Objects [3](#id.2p2csry)

*3.4.1 <Class / Object #1>* [*3*](#id.147n2zr)

*3.4.2 <Class / Object #2>* [*3*](#id.3o7alnk)

3.5 Non-Functional Requirements [4](#id.23ckvvd)

*3.5.1 Performance* [*4*](#id.ihv636)

*3.5.2 Reliability* [*4*](#id.32hioqz)

*3.5.3 Availability* [*4*](#id.1hmsyys)

*3.5.4 Security* [*4*](#id.41mghml)

*3.5.5 Maintainability* [*4*](#id.2grqrue)

*3.5.6 Portability* [*4*](#id.vx1227)

3.6 Inverse Requirements [4](#id.3fwokq0)

3.7 Design Constraints [4](#id.1v1yuxt)

3.8 Logical Database Requirements [4](#id.4f1mdlm)

3.9 Other Requirements [4](#id.2u6wntf)

**4. Analysis Models** [**4**](#id.19c6y18)

4.1 Sequence Diagrams [5](#id.3tbugp1)

4.3 Data Flow Diagrams (DFD) [5](#id.28h4qwu)

4.2 State-Transition Diagrams (STD) [5](#id.nmf14n)

**5. Change Management Process** [**5**](#id.37m2jsg)

**A. Appendices** [**5**](#id.1mrcu09)

A.1 Appendix 1 [5](#id.46r0co2)

A.2 Appendix 2 [5](#id.2lwamvv)

# 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 “Surrogate City Finder Tool (SCFT). It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications. This document is primarily intended to be proposed to the client for their approval of our understanding and implementation of the project as well as a reference for developing the first version of the system for the development team.

## 1.2 Scope

* Energy Plus Weather data files (.epw) contain exhaustive weather data for a city which utilized in simulations of the weather of a city for planning constructions and architectural modelling. Currently these files exist only for some cities of the world and (67 in India, to be precise).

* To facilitate the application of this model for the cities for whom weather data files have not yet been developed, we will make the 'Surrogate City Finder Tool' which finds weather data files of those cities around the world whose weather is the closest match to the city in

the question.

* The matching is done on the basis of parameters which induce similar weather conditions like Altitude , Latitude and Yearly Temperature Profile.
* The tool will display weather data files on a map of the world and the user will be able to make queries about specific cities and the tool will display the result on the map. The user will have an option of mailing results to their registered email Ids.
* We will be using Ruby on Rails web framework to develop a deployable web application and HTML, AJAX, CSS and Java scripting to provide a professional look to our WebApp.

## 1.3 Definitions, Acronyms, and Abbreviations

*This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.*

Definitions:

|  |  |
| --- | --- |
| **Term** | **Definitions** |
| User | Someone who interacts with the application |
| Admin/Administrator | System administrator who is given specific permission for managing and  controlling the system |
| Web App/Tool | The tool that is developed to find surrogate cities |

## 1.4 References

*This subsection should:*

*(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.*

*(2) Identify each document by title, report number - if applicable - date, and publishing organization.*

*(3) Specify the sources from which the references can be obtained.*

*This information may be provided by reference to an appendix or to another document.*

## 1.5 Overview

*This subsection should:*

*(1) Describe what the rest of the SRS contains*

*(2) Explain how the SRS is organized.*

# 2. General Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the

system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

## 2.1 Product Perspective

*This subsection of the SRS puts the product into perspective with other related products or*

*projects. (See the IEEE Guide to SRS for more details).*

## 2.2 Product Functions

*This subsection of the SRS should provide a summary of the functions that the software will perform.*

## 2.3 User Characteristics

**Earthquake Engineering Professionals** –Professionals who may want to use this tool to help them in weather assessment and modeling.

**Weather Broadcast Companies**–

**Meteorology experts and Researchers** – They know the domain best and will be the ones who will extensively use the City Finder tool.

**Common man** – They have the least knowledge about the domain, and also the software system. The user interface needs to be simple and appealing to this user.

## 2.4 General Constraints

*This subsection of the SRS should provide a general description of any other items that will*

*limit the developer’s options for designing the system. (See the IEEE Guide to SRS for a partial list of possible general constraints).*

## 2.5 Assumptions and Dependencies

*This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.*

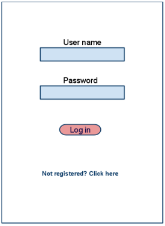
# 3. Specific Requirements

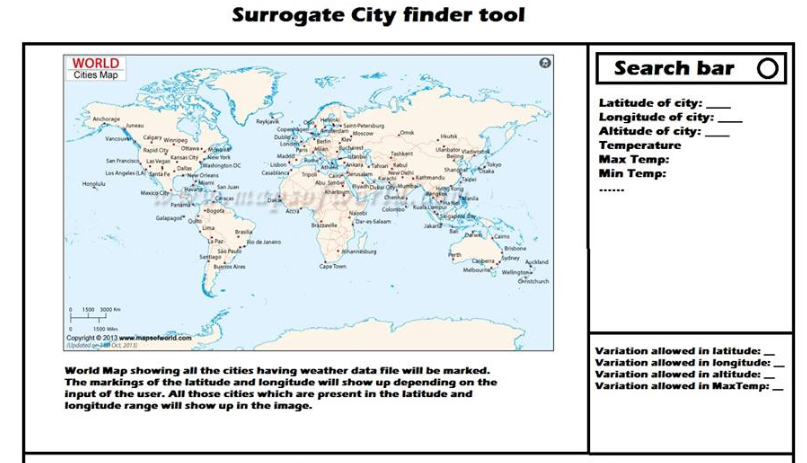
## 

## 3.1 External Interface Requirements

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

**3.1.1 User interfaces**

 Login Page



**3.1.2 Hardware Interfaces**

Since neither the mobile application nor the web portal have any designated hardware, it does not have any direct hardware interfaces. This is a normal Web Application which will be hosted on a server and the database is picked directly from the Energy Plus databases.

**3.1.3 Software Interfaces**

The tool interacts with the Energy Plus Database for weather data files. It retrieves the data files from the Database and displays the files on the map on the tool accordingly. The tool also uses the Google Map API for displaying an interactive world map within the tool and also uses the Google Map GeoCode API for locating places according to address on the map.

### Programming languages – Ruby, Bash, JavaScript, Jquery, AJAX, Python

**Web Framework(MVC)** – Ruby on Rails

**Development platform** – VIM editor, Sublime Text 2, Linux Ubuntu 13.04/14.04

**Collaboration and Version Control** –Git

**Project Documentation tools** – LibreOffice, Creately.com (for USE CASE diagrams), Google Drive, Google Docs

**3.1.4 Communications Interfaces**

The communication between the different parts of the system is important since they de

pend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems for the web application and the database and AP services.

### 

## 3.2 Functional Requirements

### 3.2.1 <Functional Requirement or Feature #1>

*3.2.1.1 Introduction*

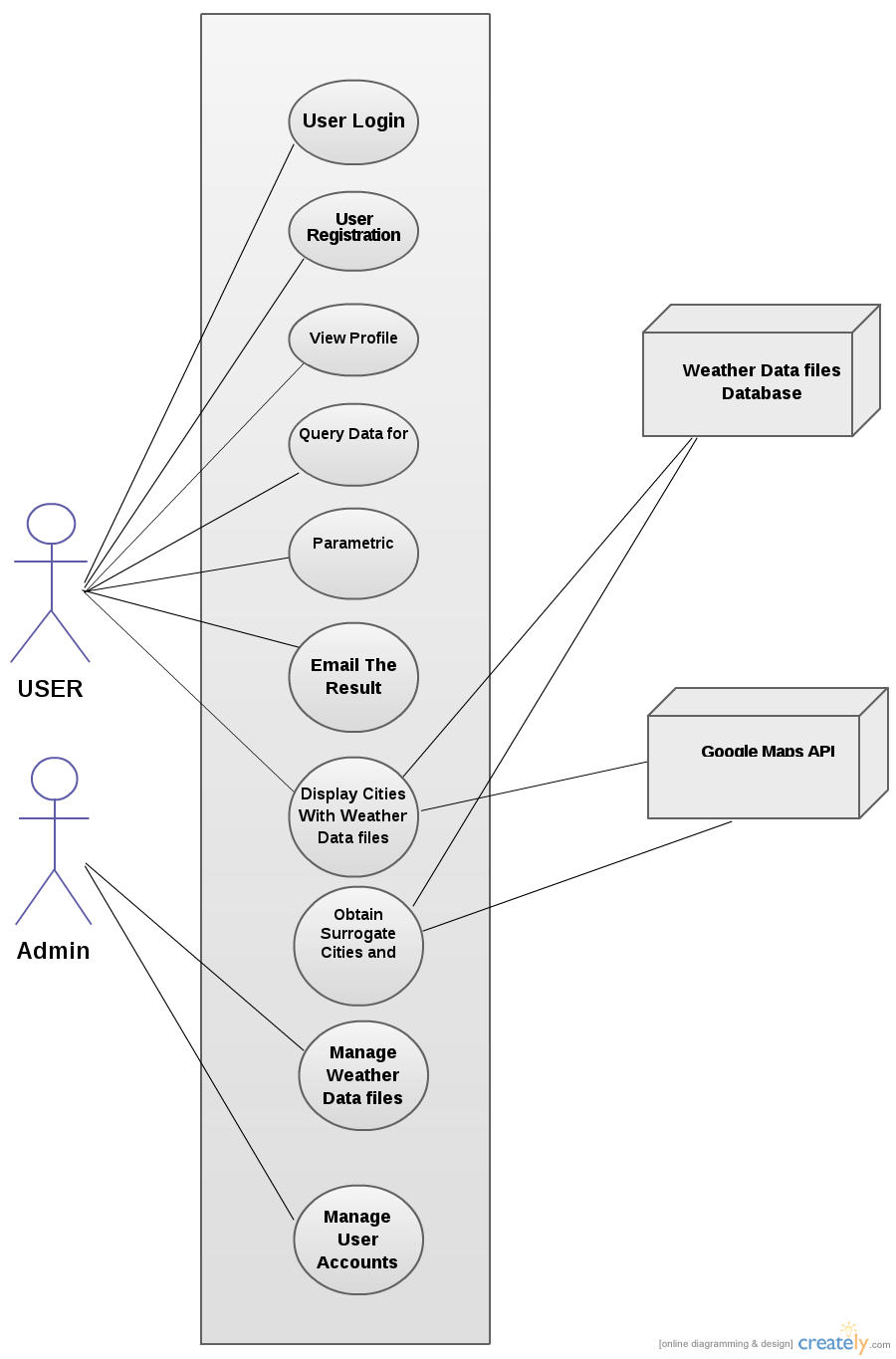
*3.2.1.2 Inputs*

*3.2.1.3 Processing*

*3.2.1.4 Outputs*

*3.2.1.5 Error Handling*

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case No.** | **Use Case Name** | **Brief Description** | **Release** |
| 1. | User login | A Login page will be implemented which will ask user for Email-Id and Password/Re-Type Password. The input will be validated with the information stored in the database and the user will directed to the main page. | R1 |
| 2. | User Registration | Create a database which will hold user information. The user information will have the following fields:  a) Full Name  b) Industry  c) Job Title  d) Email address  e) Password  f) Company Name | R1 |
| 3. | Profile View/Edit | User can view his profile which will show him his account details. It can be edited as per user’s wish. | R1 |
| 4. | Display cities with weather data files. | A map will be marked displaying all the cities existing weather data files of cities. | R1 |
| 5. | Query data for a city | User will give a city name as input. Database will search weather data files for the input city. Result will show the weather details of the city if its file exists. | R1 |
| 6. | Parametric range input | The user will input the range for the latitude, altitude and temperature matching which it permits for matching cities to their surrogates | R2 |
| 7. | Obtain Surrogate cities and save result | The tool calculates result, displays the range band on the map and highlights the matching cities. The user has an option to save the result for the particular query. | R2 |
| 8. | E-mail the result | The result page can be emailed to the user’s registered Email Id or email id of his choice. | R2 |
| 9. | Manage weather data files | The admin has options to manage the database of the weather data files and update it as more and more files become available. | R1 |
| 10. | Manage User accounts | The admin has CRUD options on User accounts. | R1 |



## 3.3 Use Cases

### 3.3.1 Use Case #1

|  |  |
| --- | --- |
| **Use Case Number:** | UC-01 |
| **Use Case Name:** | User Login |
| **Overview:** | A Login page will be implemented which will ask user for Email-Id and Password/Re-Type Password. The input will be validated with the information stored in the database and the user will directed to the main page. |
| **Actors:** | 1. User(Primary) |
| **Pre condition:** | The Application must be ready to take input from the User. The User Must Already be Registered. |
| **Flow:** | Main(success) flow:  1. A user Click on the Login button Present on the Homepage of the site.  2. The User is Redirected to the Login page.  3. User Enters his Details.  4. The Entered data is verified and found from the Database.  5. Profile Page of the user is Displayed for a successful Login, Otherwise the mismatching field is returned as a prompt to the User. |
|  | Alternate Flow:  1.Incorrect login  1.1 The user enters incorrect e-mail or password.  1.2 An error message is displayed informing the user of the wrong login details.  2. E-mail not registered  2.1 The e-mail entered by the user is not present in the database  2.2 User is shown a message that the entered username is not registered yet. |
| **Post Condition:** | The User Must be redirected to the query page. |

### 3.3.2 Use Case #2

|  |  |
| --- | --- |
| **Use Case Number:** | UC-02 |
| **Use Case Name:** | User Registration |
| **Overview:** | A new user can register to the website creating his/her own profile in our database. |
| **Actors:** | First Time User. |
| **Pre condition:** | The Application must be ready to take Input from the New User. |
| **Flow:** | Main(success) flow:  1. A new User enters the Registration page by Clicking the Register Button present on the Main menu.  2.The user enters his/her details in the page and presses the submit button.  3.The data is checked for any redundancy and prompts the user for re-input if required.  4.The data entered by the user is stored in the database  5..A prompt displaying the success/failure of registrations |
|  | Alternate flow:  1.Username already exists  1.1 A message shown to the user to enter a different username.  1.2 User enters the username again.  2.Password mismatch  2.1 A message is shown to user to re-enter the password fields  2.2 User enters the password and the re-type password fields  3. Invalid e-mail id  3.1 A message is shown to the user to enter a valid e-mail id.  3.2 User enters the e-mail id field again |
| **Post Condition:** | The User Must be redirected to his profile page |

**3.3.3 Use Case #3**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-03 |
| **Use Case Name:** | VIew/Edit Profile |
| **Overview:** | Any Registered User can View His/Her Profile and Edit it according to His/Her Requirement. |
| **Actors:** | 1. User(Primary) |
| **Pre**  **condition:** | User Needs to be Logged In. |
| **Flow:** | 1. User Clicks on View/Edit Profile Button.  2. User is Directed to his/her Profile Page.  3. Edit Options like Change Password,Email Id etc will available on the Profile Page. |
|  | Alternate flows:  1. If old Password does not match  1.1 An error will be displayed.  1.2 User will be asked to enter password again. |
| **Post Condition:** | 1. User Will be directed to His Profile page.  2. If the User Changes his/her password or Email Id, the change is made in the database also. |

**3.3.4 Use Case #4**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-04 |
| **Use Case Name:** | Display existing weather data files |
| **Overview:** | A map will be marked displaying all the cities with existing weather data files. |
| **Actors:** | 1. User(Primary) |
| **Pre condition:** | 1. User is logged in.  2. All the weather data files are present in the database. |
| **Flow:** | 1. Homepage of the site contains a map marked with the cities whose data files are present in the database.  2. The name of the cities are extracted from the data files present in the database and they are marked on the map using Google maps API. |
|  | Alternate Flows: |
| **Post Condition:** | Success |

**3.3.5 Use Case #5**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-05 |
| **Use Case Name:** | Query data for a city |
| **Overview:** | User will give a city name as input. Database will search weather data files for the input city. Result will show the weather details of the city if its file exists. |
| **Actors:** | 1. User(Primary) |
| **Pre condition:** | 1. User is Logged in.  2. City should be present in India.. |
| **Flow:** | Main (success) Flow:  1. User gives an input city in the search box.  2. The Database is searched for the input.  3. Database returns all the data (max temp, min temp, altitude, co-ordinates etc.).  4. Result is Displayed on the page. |
|  | Alternate Flows:  1. City not found in database  1.1. Redirect to user to UC-06 |
| **Post Condition:** | 1. Success |

**3.3.6 Use Case #6**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-06 |
| **Use Case Name:** | Parametric Range Input |
| **Overview:** | The user will input the range for the latitude, altitude and temperature matching which it permits for matching cities to their surrogates. |
| **Actors:** | 1. User(Primary) |
| **Pre condition:** | 1. Query(UC-05) is successfully completed. |
| **Flow:** | Main (success) Flow:  1. User must give a range for the latitude(mandatory).  2. User is asked to input the parameters maxTemp, minTemp, Altitude.  3. User is Directed to UC-07. |
|  | Alternate Flows:  1. If latitude entry is empty, an error message will be shown to fill the vacancy. |
| **Post Condition:** | 1.User gets redirected to UC-07. |

**3.3.7 Use Case #7**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-07 |
| **Use Case Name:** | Obtain Surrogate cities and save result |
| **Overview:** | The tool calculates result, displays the range band on the map and highlights the matching cities. The user has an option to save the result for the particular query. |
| **Actors:** | 1. User(Primary) |
| **Pre condition:** | 1.Success of UC-06  2.All other ranges will be initialised to NULL. |
| **Flow:** | Main Flow:  1. User will enter in the remaining parameters in the given space.  2. Depending on the latitude, MaxTemp, MinTemp and altitude range a search will be made to check the database for the weather data files.  3. Due to the latitude range, lines will be made on the map using google API marking the region to be searched.  4. All the Cities having weather Data files in this region will be checked using the remaining ranges to output the Result.  5. The Result is Displayed at the bottom of the page. |
|  | Alternate Flows:  1. If MaxTemp and MinTemp are not real numbers.  1.1. An error will be displayed.  1.2. User will be asked to input a number in the respective range.  2. If No such City exists, a Message will be displayed at the bottom of the page showing the same. |
| **Post Condition:** | 1. Success and User gets directed to UC-08. |

**3.3.8 Use Case #8**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-08 |
| **Use Case Name:** | E-mail the result. |
| **Overview:** | The result page can be emailed to the user’s registered Email Id. |
| **Actors:** | 1. User(Primary) |
| **Pre condition:** | 1. User has been redirected to the result page. |
| **Flow:** | Main flows:  1. An option will be provided to Mail the Result.  2. User may enter email id of his choice or simply mail it to the registered email.  3. Validity of the email address will be checked.  4. To get the registered email the database will be accessed. |
|  | Alternate Flows:  1. Invalid email address  1.1 An error will display the same.  1.2 User will be asked to enter email address again.  2. Email sending failed  2.1 An error message displaying the same will be shown. |
| **Post Condition:** | On Success User must be redirected to the Home Page. |

**3.3.9 Use Case #9**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-09 |
| **Use Case Name:** | Manage weather data files |
| **Overview:** | The admin has options to manage the database of the weather data files and update it as more and more files become available. |
| **Actors:** | 1. Admin(Primary) |
| **Pre condition:** | 1.Admin must be logged in. |
| **Flow:** | Main(success) Flow:  1. An option will be present to redirect admin to update page.  2. Update Weather Data files.  3. Add Weather Data files. |
|  | Alternate Flows:  1. The Weather data file if not in the correct format.  1.1. An Error message will be displayed.  1.2. Admin will be redirected to Home page. |
| **Post Condition:** | 1. Success. |

**3.3.10 Use Case #10**

|  |  |
| --- | --- |
| **Use Case Number:** | UC-10 |
| **Use Case Name:** | Manage User accounts |
| **Overview:** | The admin has CRUD options on User accounts. |
| **Actors:** | 1. Admin(Primary) |
| **Pre condition:** | 1. Admin must be logged in. |
| **Flow:** | Main (success) Flow:  1. Admin has the privilege to see profile of all the registered Users.  2. He can delete the profile of a registered user.  3. Database will be updated on deletion. |
|  | Alternate Flows: |
| **Post Condition:** | 1. Success |

…

## 3.4 Classes / Objects

### 3.4.1 <Class / Object #1>

3.4.1.1 Attributes

3.4.1.2 Functions

<Reference to functional requirements and/or use cases>

### 3.4.2 <Class / Object #2>

…

## 3.5 Non-Functional Requirements

1. Search should take less than five seconds.
2. Maintaining a professional look for the tool.
3. Developing User Friendly interface

### 3.5.1 Performance

### 3.5.2 Reliability

### 3.5.3 Availability

### 3.5.4 Security

### 3.5.5 Maintainability

### 3.5.6 Portability

## 3.6 Inverse Requirements

*State any \*useful\* inverse requirements.*

## 3.7 Design Constraints

*Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.*

## 3.8 Logical Database Requirements

*Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.*

## 3.9 Other Requirements

*Catchall section for any additional requirements.*

# 4. Analysis Models

*List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.*

## 4.1 Sequence Diagrams

## 4.3 Data Flow Diagrams (DFD)

## 4.2 State-Transition Diagrams (STD)

# 5. Change Management Process

*Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.*

# A. Appendices

*Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.*

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## A.1 Appendix 1

## A.2 Appendix 2