# Software Requirements

Version 3, last updated by Shizhe LIU at 2020-10-5

# Software Requirements Specification (SRS)

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 9-18-20 | Weikang DU &  Di AN &  Shizhe LIU | Adding/Editing Use Cases |
| 9-18-20 | Yuzhe CHEN | Introduction, Concept of Operations |
| 9-18-20 | Kedong XIU | Behavioral Requirements, Quality Requirements |
| 9-25-20 | Shizhe LIU | Add some User Requirements |
| 5-10-20 | Shizhe LIU | Add use case diagram |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[Software Requirements 1](#_Toc52806010)

[Software Requirements Specification (SRS) 1](#_Toc52806011)

[1.  Introduction 3](#_Toc52806012)

[1.1    Intended Audience and Purpose 3](#_Toc52806013)

[1.2    How to use the document 3](#_Toc52806014)

[2.  Concept of Operations 4](#_Toc52806015)

[2.1    System Context 4](#_Toc52806016)

[2.2 System capabilities 4](#_Toc52806017)

[3.  Use Cases 5](#_Toc52806018)

[Case 1: User Log in 5](#_Toc52806019)

[Case 2: User Log out 5](#_Toc52806020)

[Case 3: User Sign up 5](#_Toc52806021)

[Case 4: User Wants to Call the Algorithm 5](#_Toc52806022)

[Case 5: Store X-ray Photo data to database 6](#_Toc52806023)

[Case 6: Store results data to database 6](#_Toc52806024)

[Case 7: Modify relevant results 6](#_Toc52806025)

[Case 8: Query history detection results 6](#_Toc52806026)

[9: Use case diagram 7](#_Toc52806027)

[3.    Behavioral Requirements 8](#_Toc52806028)

[3.1 System Inputs and Outputs 8](#_Toc52806029)

[3.2 Detailed Output Behavior 9](#_Toc52806030)

[4. Quality Requirements 9](#_Toc52806031)

[5.    Expected Subsets 10](#_Toc52806032)

[6.    Fundamental Assumptions 10](#_Toc52806033)

[7.    Expected Changes 10](#_Toc52806034)

[8.    Appendices 10](#_Toc52806035)

[8.1    Definitions and acronyms 10](#_Toc52806036)

[8.2    References 11](#_Toc52806037)

## 1.  Introduction

### 1.1    Intended Audience and Purpose

This document is intended to provided information guiding the development process, ensuring that all system requirements are met. The following entities may find the document useful:

Primary Customer - This page will detail all of the server requirements as understood by the production team. The customer should be able to determine that their requirements will be correctly reflected in the final product through the information found on this page.

Development Team – Details of specific requirements that the final software build must include will be located here. Developers can use this document to ensure the software addresses each of these requirements.

QA Team - By developing testing procedures founded in the system requirements, the QA Team can create a comprehensive testing regimen that will guarantee requirements are met.

Client Team - Analyze the relevant instructions of the database through this document, call the relevant methods to complete the client functional requirements.

### 1.2    How to use the document

Table of Contents:  
1. Introduction  
2. Concept of Operations - broad description of the purpose of the application  
  2.1 System Context - details any specific system requirements the application will require to run  
  2.2 System Capabilities - description in prose of all capabilities available to the user in the address book  
  2.3 Use cases - A detailed look at each functional requirement, describing the application context both before and after an action is taken  
3. Behavioral Requirements - How the application will interact with a user  
  3.1 Input and output requirements - A description of allowed inputs and generated outputs  
    3.1.1 Input - Describes any restrictions that will be placed on allowed input  
    3.1.2 Output - Describes the range of outputs that can be generated  
  3.2 Detailed Output Behavior - Output descriptions in prose  
4. Quality Requirements - Requirements not pertaining to the function of the application will be listed here  
5. Expected Subsets - Expected levels of functionality at checkpoints during development  
6. Fundamental Assumptions - Some specifics about input, output, or behavior upon which other requirements are founded will be listed here  
7. Expected Changes - Future features and directions the project is expected to take  
8. Appendices - Details aiding the understanding of this document  
  8.1 Definitions and acronyms - Any technical terms or abbreviations will be spelled out here for ease of use of the document  
    8.1 Definitions - Definitions of technical or unusual terminology  
    8.1.2 Acronyms and Abbreviations - Any abbreviated terms will be expanded here  
  8.2 References - any external references necessary or helpful to understanding this document will be listed here

## 2.  Concept of Operations

  The main functions required on the server side implemented by our group are as follows:

1. The interaction between the client and the database is completed through the server, user information can be obtained through the database, and the page loading of the client can be completed through the database.

2. Complete algorithm call through the server side, perform data processing in the background, and store, load and modify data through the database.

3. Security issues, through the server and client encryption algorithms to complete data encryption, to solve the security problem of data transmission.

### 2.1    System Context

**System Requirements:**  
We mainly use the Java language for server development, so we need the latest Java runtime environment on the system.

Windows:

* Windows 10
* Windows 8.x
* RAM: 128 MB
* Disk space: 124 MB for JRE; 2 MB for Java Update
* Processor: Minimum Pentium 2 266 MHz processor

Linux:

* Ubuntu Linux 14.x (8u25 and above)
* Ubuntu Linux 15.04 (8u45 and above)
* Ubuntu Linux 15.10 (8u65 and above)

### 2.2 System capabilities

 First, the client completes the login through the login system, which requires the server to determine the data format and access the database to determine the correctness of the user input information; second, the user uses the scoliosis function, and the server receives the x-ray input from the user The film is judged by calling the written algorithm and calling the data stored in the database. The result is given and returned to the client. The result is entered in the client interface; finally, in the process of information transmission and access to the database, The server also needs to encrypt and decrypt data to prevent data loss and theft during transmission. The above is a summary of our server-side system functions.

## 3.  Use Cases

### Case 1: User Log in

**Players:**End User  
**Goals:**The user wants to log in.

**Preconditions:**The user has already registered for an account.  
**Main Flow:**  
1.1 The user enters username and password.  
1.2 The server confirms the user’s information.  
1.3 The server returns the home page of the website.

**Alternate Flow:**  
1.2.1 the username or password is incorrect.  
      1.2.1.1  The server prints the error message.  
      1.2.1.2 The server returns the same page.

1.2.1.3 The logger records this attempt.

**Post conditions:**If the user changed the password of the account, the changes should persist the next time that the user log in.

### Case 2: User Log out

**Players:**End User  
**Goals:**The user would like to log out, and go back to the log in page.

**Preconditions:**The user has already logged in.  
**Main Flow:**  
2.1 The user clicks the *log out* button.  
2.2 The server returns the log in page.

### Case 3: User Sign up

**Players:** End User  
**Goals:** The user would like to create a new account.

**Main Flow:**  
3.1 The user clicks the *sign up* button.

3.2 The user enters mobile, email, username and password.  
3.3 The server checks if the username has already been registered.

3.4 The server creates an account for the user, and returns the home page.  
**Alternate Flow**:  
3.3.1 The server prints the error message.

3.3.2 The server returns the log in page.

**Postconditions:**The account and user information should be encrypted before saving to database to protect privacy.

### Case 4: User Wants to Call the Algorithm

**Players:**End User  
**Goals:**The end user wants to call the algorithm to calculate the result.  
**Preconditions:**The server is open and running, and has get data from the client.  
**Main Flow:**  
4.1 The server receives data from the client.  
4.2 The server transfer the data as parameters to the algorithm.  
4.3 The algorithm does the calculation and get a result.

4.4 The algorithm transfers the result to the server.

4.5 The server shows the result to the user.  
**Exception Flows:**  
4.1.1 The server fails to get the data from the client.

The server shows the error messages to the user.

### Case 5: Store X-ray Photo data to database

**Players:**End User  
**Goals:**Users can store X-ray photos of patients in the database.  
**Main Flow:**  
5.1 Upload X-ray Photo.  
5.2 Check the validity of the Photo.  
5.3 Store photos to database.

### Case 6: Store results data to database

**Players:**End User  
**Goals:**Users get the pathological analysis results then store them to database.

**Preconditions:**The user has already registered for an account.  
**Main Flow:**  
6.1 Computer analyses the pathological results.  
6.2 Results be store in database.

### Case 7: Modify relevant results

**Players:**End User  
**Goals:**When the calculated results deviate greatly from the actual results, users should actively modify the results to ensure that patients will not receive wrong examination results.  
**Main Flow:**  
7.1 The user views the inspection results.  
7.2 It does not conform to the actual results, and the inspection results need to be modified.  
7.3 Update the database.

### Case 8: Query history detection results

**Players:**End User  
**Goals:**The user can query the detection results of the last two years.

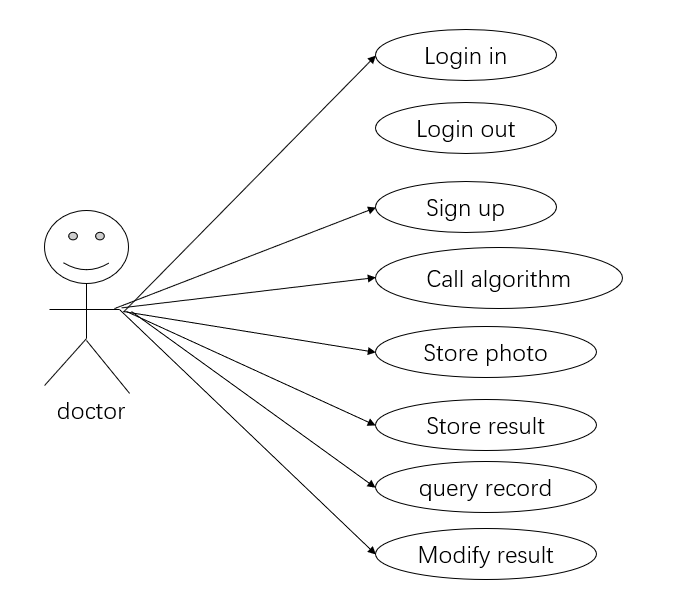
**Main Flow:**

8.1 The user enters the relevant information of the object to be queried, such as ID card or medical card.

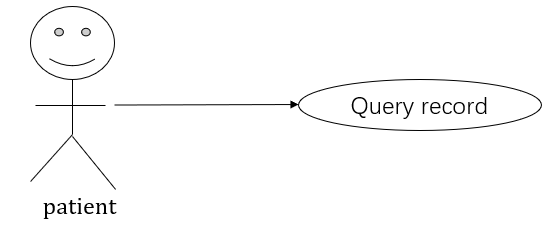
8.2 The system returns the relevant query results.

### 9: Use case diagram

i. For doctor:



ii. for patient:



Note:

i. Patients can view their own results through mobile devices such as Android phones;

ii. For the use cases that patients can query records, the server should be able to communicate with mobile devices such as Android phones.

iii. In fact, the server also needs to establish communication with other devices, but this is not considered in the use case, so it is omitted.

## 3.    Behavioral Requirements

### 3.1 System Inputs and Outputs

#### 3.1.1 Inputs

The input of the application comes from the user. Users can choose registration, login, and logout options. You can also choose the following options: "Import Picture", "Export Judgment Results", “Query history detection results”. For the registration and login options, the user enters through the text box provided on the registration and login interface.  
  
Input when registering/login:  
      \*General: At least two inputs will be required for a new contact. One of those two inputs must be a name. There is no length limit on any of the fields below.  
      \* Nickname: A nickname can have a first name or last name, or both. The first name and last name will be separate fields. The system will accept any characters in the first name and last name fields.  
      \* Telephone Numbers: Only Chinese phone number will be considered valid. The only acceptable input will be numeric characters between 0-9 and special characters'(',')' and'-'.  
      \* Email Address: Email address will be a field. Acceptable input will be alphanumeric and must contain "@" and ".". symbol. If the system does not detect "@" and ".", the email address is considered invalid.  
  
Importing:  
       \* A file must be imported in a very specific format. The selected file format must be a medical DICOM format file or a picture format file such as JPG and JEPG. In addition, we agreed that the size of an image should not exceed 10M.

#### 3.1.2 Outputs

A GUI that displays the imported X-ray pictures and judgment results. Below the picture are the diagnosis results of this X-ray picture and interactive buttons created for users.  
   Outputs to The User:  
      Contact the doctor online:  
      \* If the user is offline after clicking the button, a system warning will pop up. If the user wants to have a further understanding of the condition after seeing the diagnosis result, the user can click the contact doctor button to let the system know that the user wants to contact the doctor. If the user repeatedly clicks this button, a system prompt will pop up to prevent multiple clicks.  
      Re-diagnosis results:  
     \* If the user has doubts about the diagnosis result for the first time and wants to perform the test again, the user can click the Rebuild Test button to let the system know that the picture will be tested again. But if the user does not click for the first time, a system prompt will appear. Prevent users from over-detecting the same picture.       
      Print result:

      \* If the user wants to save the diagnosis result locally, the user can get a PDF file composed of X-ray pictures and diagnosis results by clicking the print result button.

### 3.2 Detailed Output Behavior

The output is a GUI, and it can be said that the user provides the function of displaying pictures and diagnosis results of pictures. "Contact Doctor Online" will contact the doctor directly. "Re-diagnosis results" can provide users with the function of re-detecting X-ray pictures. "Print Results" allows users to combine X-ray pictures and diagnostic results of the pictures into a PDF file and save it locally. "Logout" can close the currently opened diagnosis result and the user logs out.

The application contains the basic system requirements provided by the user, and has functions such as saving diagnosis results, contacting doctors online, and retesting. These features were chosen because they provide the most user-friendly experience.

## 4. Quality Requirements

The application must be competitive with similar applications in regards to performance, reliability, consistency, and scalability.  
Performance: Responsiveness to user input  
      \* Standard actions that response the client should not exceed 500ms execution time.  
      \* Receiving the photos is a possible exception to the above standard, as download time will depend on the speed of the Internet.  
Reliability: Confidence that actions taken will not result in errors, and that changes made to address books are persistent  
      \* User input should not produce faults or errors that impact or hinder use of the application

\* The maximum number of clients allowed to connect to the related database is 50  
Consistency: Persistent data on the database   
      \* Address photos should be modifiable after being added in such a way that target fields can be changed without affecting data in other fields.  
Scalability: Ease of extending application capabilities  
      \* Application should be modularized such that adding/extending features and functions only require changes to a single component and the interface with that component, if applicable.  
Furthermore, the application should adhere to U.S. Postal standards, validating input against the appropriate specifications. Input that deviates from the U.S postal standards will not be rejected, but the user will be notified that the input deviates from standards.

## 5.    Expected Subsets

L0:  
-Ability to contact with the client  
-Ability to call the interface of algorithm to calculate the results  
  
L1:  
-Database capable of: storing complete photos and results, editing entries, and saving   
-Ability to import and export address data such as photos on the database

## 6.    Fundamental Assumptions

The application can run on any system that is capable of running Java Version 8.  
The application will not terminate when all windows are closed.  
Software updates will be downloaded by the end user as opposed to pushed out by the developers.

## 7.    Expected Changes

   Features to Add:

1. Support related algorithms written in various languages, such as C language, Java language, python language, etc.

   Future Platforms:

## 

## 8.    Appendices

### 8.1    Definitions and acronyms

#### 8.1.1    Definitions

|  |  |
| --- | --- |
| **Keyword** | **Definitions** |
|  |  |
|  |  |

#### 8.1.2    Acronyms and abbreviations

|  |  |
| --- | --- |
| **Acronym or**  **Abbreviation** | **Definitions** |
| GUI | Graphical User Interface |
|  |  |
|  |  |

### 8.2    References

Comments are disabled for this space. In order to enable comments, Messages tool must be added to project.

You can add Messages tool from Tools section on the Admin tab.