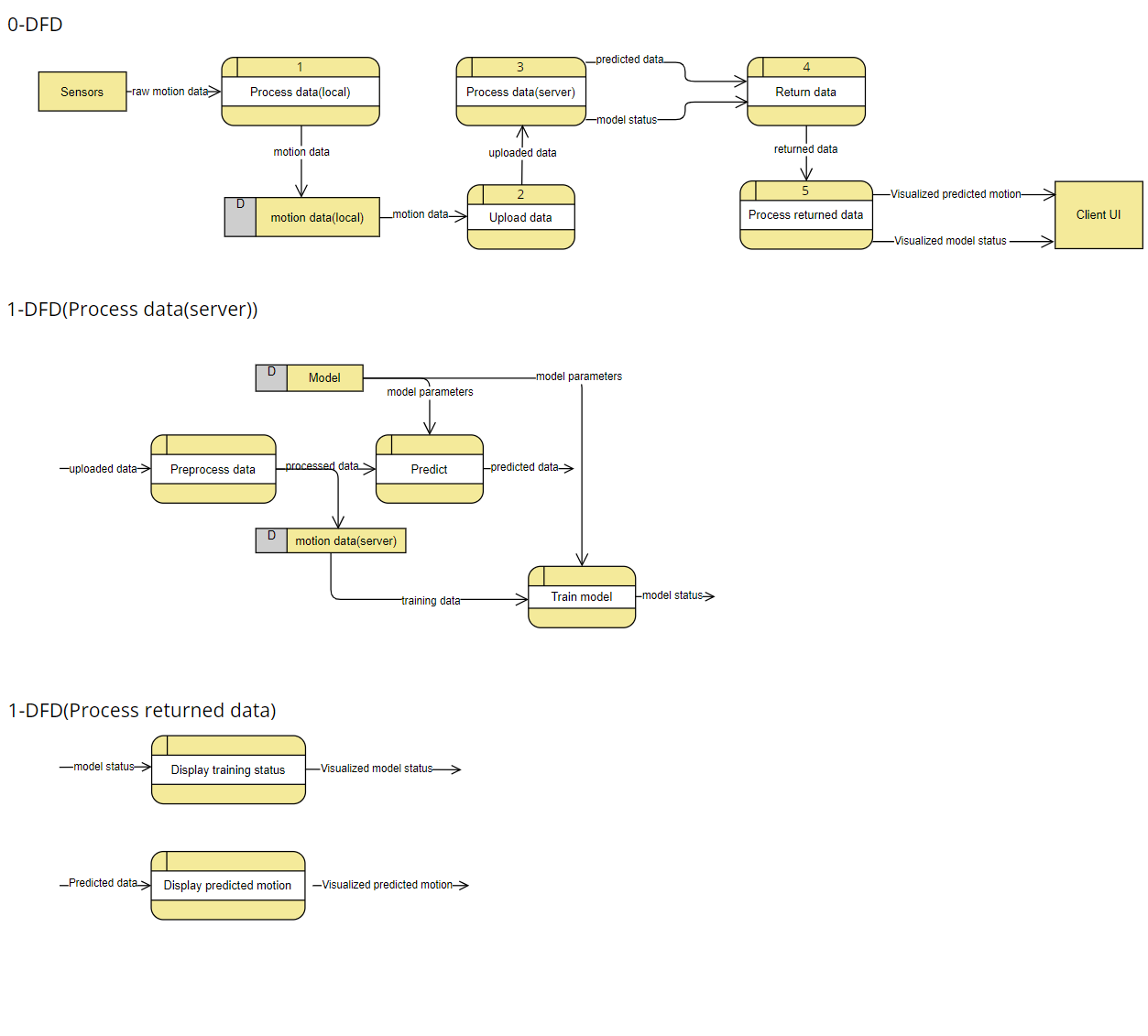
# Software Requirements

Version 51, last updated by [rylandm](https://uocis.assembla.com/profile/rylandm) at 2018-02-02

# Software Requirements Specification (SRS)

Revision History:

|  |  |  |
| --- | --- | --- |
| Date | Author | Description |
| 3.25 | Haotian Tang | Introduction/Concept of Operations |
| 3.26 | Haotian Tang | Use Cases |
| 3.27 | Haotian Tang | Behavioral Requirements/ Expected Subsets/ Fundamental Assumptions/Expected Changes |
| 3.29 | Jiazheng Pan | Revise the System capabilities and Use case |



Contents

[Software Requirements 1](#_Toc99386084)

[Software Requirements Specification (SRS) 1](#_Toc99386085)

[1.  Introduction 4](#_Toc99386086)

[1.1    Intended Audience and Purpose 4](#_Toc99386087)

[1.2    How to use the document 4](#_Toc99386088)

[2.  Concept of Operations 5](#_Toc99386089)

[2.1    System Context 5](#_Toc99386090)

[2.2 System capabilities 6](#_Toc99386091)

[3.  Use Cases 6](#_Toc99386092)

[Case 1: User Wants to Stop using the Website 6](#_Toc99386093)

[Case 2: User Wants to regist an account 6](#_Toc99386094)

[Case 3: User Wants to login their account 7](#_Toc99386095)

[Case 4: User Wants to Delete Account 7](#_Toc99386096)

[Case 5: User Wants to Start/Stop to Collect human motion data 8](#_Toc99386097)

[Case 6: User Wants to View current status of the whole model. 8](#_Toc99386098)

[Case 7: User Wants to reset model 8](#_Toc99386099)

[Case 8: User Wants to Individualize model 9](#_Toc99386100)

[3.    Behavioral Requirements 9](#_Toc99386101)

[3.1 System Inputs and Outputs 9](#_Toc99386102)

[3.2 Detailed Output Behavior 10](#_Toc99386103)

[4.2   Quality Requirements 10](#_Toc99386104)

[5.    Expected Subsets 11](#_Toc99386105)

[6.    Fundamental Assumptions 11](#_Toc99386106)

[7.    Expected Changes 11](#_Toc99386107)

[8.    Appendices 11](#_Toc99386108)

[8.1    Definitions and acronyms 11](#_Toc99386109)

[8.2    References 12](#_Toc99386110)

## 1.  Introduction

### 1.1    Intended Audience and Purpose

This document is intended to provided information guiding the installation and 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 application 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.  
**User** - A prospective user will be able to use this document to identify the main functionailty included in the application. Furthermore, the application will have a set of system requirements before the application can be run. Details regarding these requirements can be found here.  
**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.

### 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 requirments - 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. Appendicies - 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 Abreviations - Any abreviated 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 goal is to create a system interface that uses algorithms to predict the user’s motion in the next step. It allows the user to choose one among several available models to use and monitor the status of the system in real time. Users don't bother to install any app. They just type the URL of the project into the browser to use it. For more details on the usage and capabilities of the application, read the section [System Capabilities](https://uocis.assembla.com/spaces/cis422w18-team2/wiki/Software_Requirements#System%20Capabilities).

### 2.1    System Context

**System Requirements:**  
Requires a browser such as chrome, edge, firefox and et.

Windows:

* Windows 10 (8u51 and above)
* Windows 8.x (Desktop)
* Windows 7 SP1
* Windows Vista SP2
* Windows Server 2008 R2 SP1 (64-bit)
* Windows Server 2012 and 2012 R2 (64-bit)
* RAM: 128 MB
* Disk space: 124 MB for JRE; 2 MB for Java Update
* Processor: Minimum Pentium 2 266 MHz processor

Mac OS X:

* Intel-based Mac running Mac OS X 10.8.3+, 10.9+
* Administrator privileges for installation

Linux:

* Oracle Linux 5.5+1
* Oracle Linux 6.x (32-bit), 6.x (64-bit)2
* Oracle Linux 7.x (64-bit)2 (8u20 and above)
* Red Hat Enterprise Linux 5.5+1, 6.x (32-bit), 6.x (64-bit)2
* Red Hat Enterprise Linux 7.x (64-bit)2 (8u20 and above)
* Ubuntu Linux 12.04 LTS, 13.x
* 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

This interface is mainly based on HTML + CSS + JS. As the interactive interface of the whole system oriented to users, it is convenient and easy to use. Instead of downloading an app, users just use the browser to visit our website. Firstly the user is required to register an account before they use it. Users need to fill in their personal information as well as their phone number. Then a captcha will be sent to their phone in order ot prevent bots. After filling all the information, the account is successfully registered and users can use it to log in. In the personal homepage, users can change their information and upload a personalized profile picture.

In order to record motions and collect the data, a motion sensor is needed. Any ternimal device that can link the Internet is acceptable. The sensor should be added/connected to the system through bluetooth so that the collected data can be uploaded. With an connected sensor, users can start to record their motion. Also they can pause or ternimate the process during recording. If some problems occur such as disconnection to the server and unsuccessful collection of the data, they can retry the recording. After the data is collected, they can choose whether to upload it or discard it since the data may be unsatisfactory.

There are several available models in the system. Users should specify one model to use. They can view the current status of the model such as whether the model is trainig, changes of the loss function and so on. This may help users to better utilize the model. What’s more, they can do some specific motions to individualize the model. That is, they use the sensor to record their motions and finetune the basic model with the collected data. In this way, the model is able to predict the user’s motion in next step with more confidence. If the new trained model is unsatisfactory, they can discard it and reset it to the initial status.

## 3.  Use Cases

### Case 1: User Wants to Stop using the Website

**Players:**End User  
**Goals:**The end user wants to stop using the website.  
**Preconditions:**The website is open.  
**Case:**  
1.1 From the section of the operation of models, the end user selects the "disconnect" option.  
1.2 The user selects "stop training model" option.  
**Alternate Flows:**  
1.2.1 One or more of the address books that are open have unsaved changes  
         The user is warned about quitting before disconnecting and stoping training.  
      1.2.1.1 The user decides to disconnect their connection.   
                  The website closes after disconnecting the connection.  
      1.2.1.2 The user decides to continue with the termination.  
                  No changes are saved. The application terminates.  
**Exception Flows:**  
1.2.2 The user forces the termination (by shutting down their machine, using Mac's Force Quit, Quitting from the Application menu on the Menu Bar on a Mac, etc)  
         The system forcibly disconnects the connection and stops training the model.

**Postconditons:**If a user exits the system normally, the status of the last exit remains. However, when the user forcibly exits, the saved state may not be correct.

### Case 2: User Wants to regist an account

**Players:**End User  
**Goals:**The end user would like regist an account to use the system.

**Preconditions:**The registration page is displayed.  
**Case:**  
2.1 The end user clicks the button of registration.  
2.2 The system prompts user to fill in neccessary information.  
**Alternate Flows:**  
2.2.1 The user did not fill in the neccessarily required information  
         The user is warned to fill them in.

**Exception Flows:**  
2.1.1 The required information is still blank.  
         User will fail in registing an account.  
**Postconditions:**Automatically log in to the lastly registered account.

### Case 3: User Wants to login their account

**Players:** End User  
**Goal:** The user would like to login the account.   
**Preconditions:** The login page is displayed.  
**Case**:  
3.1 The end user clicks the button of login.  
3.2 The user fills in account id and password.  
**Alternate Flows**:  
3.1.1 The wrong account id is filled.  
         The system prompts user to retry.

3.1.2 The wrong password is filled.  
         The system prompts user to retry.  
  
**Exception Flows:**  
3.2.1 The wrong password is filled.  
          User fail in logining the account.

**Postconditions:**jump to the main page.

### Case 4: User Wants to Delete Account

**Players:** End User  
**Goal:**The end user wishes to delete his account and never use system again.  
**Preconditions:** The information in the account is no longer useful.  
**Case:**  
4.1 From the menu, the end user selects the "delete account" option  
4.2 Then user need reconfirm "delete account" option.  
**Alternate Flows:**  
4.2.1 If the user is asked again to confirm deleting the account whether the user cancels the deletion.  
         The end user will be returned to the last page.  
**Exception Flows:**  
4.2.2 If the user deletes their account by mistake.  
         The account is never got back again.  
  
**Postconditions:** All information in the account will be destroyed. If users would like to use it again, they will need to apply for a new account.

### Case 5: User Wants to Start/Stop to Collect human motion data

**Players:**End User  
**Goals:**The user wishes to collect his own motion data from sensors.  
**Preconditions:**The bluetooth is connected actually.  
**Case**:  
5.1 From the menu, the end user selects the "start/stop to collect" option  
5.2 The user visually selects motion data.  
**Alternate Flows:**  
5.3.1 If user click "start", system starts to receive data from sensors.  
5.3.2 If user click "stop", system stops to receive data from sensors.  
**Exception Flows:**  
5.4.1 The Bluetooth connection is not ready.  
          The user is prompted by the system to connect bluetooth with terminal equipment.  
**Postconditions:**The system displays that data is loaded in real time.

### Case 6: User Wants to View current status of the whole model.

**Players:**End User  
**Goals:**The end user wants to moniter the process of model training.  
**Preconditions:**The application is open and running.  
**Case:**  
6.1 From the menu, the end user selects the "the status of model" page  
6.2 A line graph by time as X-axis is displayed on the GUI.  
**Alternate Flows:**  
6.2.1 If the data format does not conform to the model's input requirements, the model throws an error.  
         6.2.1.1 The user is warned to check the format of data.  
         6.2.1.2 The user is prompted to change a suitable model to train.   
**Exception Flows:**  
6.3.1 The Bluetooth connection is suddenly disconnected, resulting in the model lacking of input.

The user is prompted to reconnected the bluetooth and retrain the model again.

**Postconditions:**The precision and loss of the model are displayed on the GUI in real time.

### Case 7: User Wants to reset model

**Players**: End User  
**Goals:**The user wants to reset model.  
**Preconditions:** The current model has stoped training.  
**Case:**   
7.1 From the menu, the end user selects the "reset" option.  
7.2 The user selects a new model.  
7.4 The user click "start " button.  
**Alternate Flows:**  
7.5.2 If the last model is still running.  
         The system provides the user the option to close the last model or continue running it.  
  
**Exception Flows:**  
7.5.1 The reset operation was forcibly performed.  
         Previous training results will be discarded, the system automatically switches to a new model.  
**Postconditions: The system will be trained with a new model.**

### Case 8: User Wants to Individualize model

**Players:**End User  
**Goals**: The user wants to individualize model for better prediction.  
**Preconditions: The account is loged in**.  
**Case:**  
8.1 The end user select the "model defined by self " option.  
8.2 The user uploads the model prepared locally to the server.  
**Exception Flows:**  
8.4.1 If there is a problem with the user's network connection.  
         The system will prompt user to check the connection and retry.  
  
**Postconditions:**The new model is displayed in the queue of model.

### Case 9: User Wants to add a new sensor

**Players:**End User  
**Goals**: The user wants to add/ connect a new sensor to the system to record motions.  
**Preconditions: The account is loged in**. The motion sensor is available and can link the Internet.  
**Case:**  
9.1 The end user selects the "add devices" option.  
9.2 The user selects the certain device in the detection list and connects it to the system.

9.3 The user receives the successful tips and sees the new device in the lists.

**Alternate Flows:**

9.4.1 If the sensor cannot be detected, the system will propmt the user to try again.

**Exception Flows:**  
9.5.1 The sensor is running down.  
         The system cannot detect the device.  
  
**Postconditions:**The new added device is shown in the device list.

## 3.    Behavioral Requirements

### 3.1 System Inputs and Outputs

#### 3.1.1 Inputs

Inputs to the application come from the user. The user can select the following options from the "File" menu: "input", "output", "model list", "the visualization of result", "profile", "start training", "stop training", and "export training logs". After connecting the sensor, the user can upload the data in the sensor for training the model. Users can also upload their own models for personalized customization.  
  
   Inputs When Adding/Editing An Entry:  
      \*General: Users upload data either manually or by connecting to sensors.  
      \* manual: Data that meets specific format is uploaded to the server for training.  
      \* sensors: Users input data from the sensor and system preprocess input to the model.  
      \* self-defined model: Users can upload models, but the inputs and outputs of model should conform to a specific format.  
  
Importing:  
       \* A file must be in a very specific format for importing.

#### 3.1.2 Outputs

A GUI displaying model status and predicted motion.  
   Outputs to The User:  
      Adding Model:  
      \* If the system receives a model with incorrect format. Errors will occur in subsequent training.  
  
      Viewing the status of training:  
      \* The training will be presented as a line chart.  
       
      Saving Training Logs:  
      \* The system prompts user to save the logs. (usually export in .txt format)

### 3.2 Detailed Output Behavior

The output is a GUI which makes it easy for users to use.

The output is a GUI "Export As" that export logs to a local PC/Phone.

"display" can display the process of training as a line chart.

### 4.2   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 manipulate address books or their contents should not exceed 500ms execution time.  
      \* Searching on address books is a possible exception to the above standard, as search performance will depend on the size of the address book.  
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  
      \* Any modifications to an address book should produce a lasting change that persists through any following series of actions taken by the user.  
Consistency: Persistent data in address book contents  
      \* Address book contents 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:  
-Basic GUI with the ability to send data to and recall data from device  
-Ability to view the status of model in GUI  
  
L1:  
-System capable of: storing data, select models, Individualize model, and saving result.  
-Complete GUI for access to training model.  
-Ability to import and export data.  
-Recent model

## 6.    Fundamental Assumptions

Users can visit our website through any brower.  
The application will terminate when all windows are closed.

## 7.    Expected Changes

   Features to Add:  
      connect sensors

   Future Platforms:  
      Write Front End For Andriod  
      Port Application to iOS

## 

## 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.