**Emanon**

**Software Requirements Specification**

*Modification history:*

|  |  |  |  |
| --- | --- | --- | --- |
| *Version* | *Date* | *Who* | *Comment* |
| V1.0 | 3/3/15 | Sam, Monica, Andrew | Started the document |
| V1.1 | 3/5/15 | Sam, Monica, Andrew, Grant | More stuff |
| V1.2 | 3/16/15 | Andrew | Compiling our work |
| V1.3 | 3/23/15 | Andrew, Sam, Monica | Finalizing SRS document |

# Team Name: **Emanon**

## *Team Members:*

* Acosta, Monica [msacosta2171@eagle.fgcu.edu](mailto:msacosta2171@eagle.fgcu.edu)
* Galitz, Grant [gegalitz3491@eagle.fgcu.edu](mailto:gegalitz3491@eagle.fgcu.edu)
* Henk, Andrew [Ajhenk0955@eagle.fgcu.edu](mailto:Ajhenk0955@eagle.fgcu.edu)
* Ramirez, Samuel [sjramirez7801@eagle.fgcu.edu](mailto:sjramirez7801@eagle.fgcu.edu)

## *Contents of this Document*

1. Introduction

* Software to be Produced
* Applicable Standards
* Definition, Acronyms, and Abbreviations

1. Product Overview

* Assumptions
* Stakeholders
* Use Case Diagram
* Use Case Descriptions

1. Specific Requirements
2. Additional material

# **SECTION 1: Introduction**

## ***Software to be Produced:***

* The software to be produced is a medical electronic record system that will be able to retrieve and send information using HIPAA compliant standards.
* The Electronic record system will have a graphical user interface.
* The user interface will have a variety of micro services.
  + Deleting Files
  + Adding Patients
  + Adding Files
  + Modifying Files
  + Sorting Files

## ***Applicable Standards:***

* HIPAA

## ***Definitions, Acronyms, and Abbreviations:***

* HIPAA is the federal Health Insurance Portability and Accountability Act of 1996. The primary goal of the law is to make it easier for people to keep health insurance, protect the confidentiality and security of healthcare information and help the healthcare industry control administrative costs.
* Defensive programming is a form of defensive design intended to ensure the continuing function of a piece of software under unforeseen circumstances. The idea can be viewed as reducing or eliminating the prospect of Finagle's law having effect. Defensive programming techniques are used especially when a piece of software could be misused.
  + Defensive programming is an approach to improve software and source code, in terms of:
    - General quality - reducing the number of software bugs and problems.
    - Making the source code comprehensible - the source code should be readable and understandable so it is approved in a code audit.
    - Making the software behave in a predictable manner despite unexpected inputs or user actions.

# **SECTION 2: Product Overview**

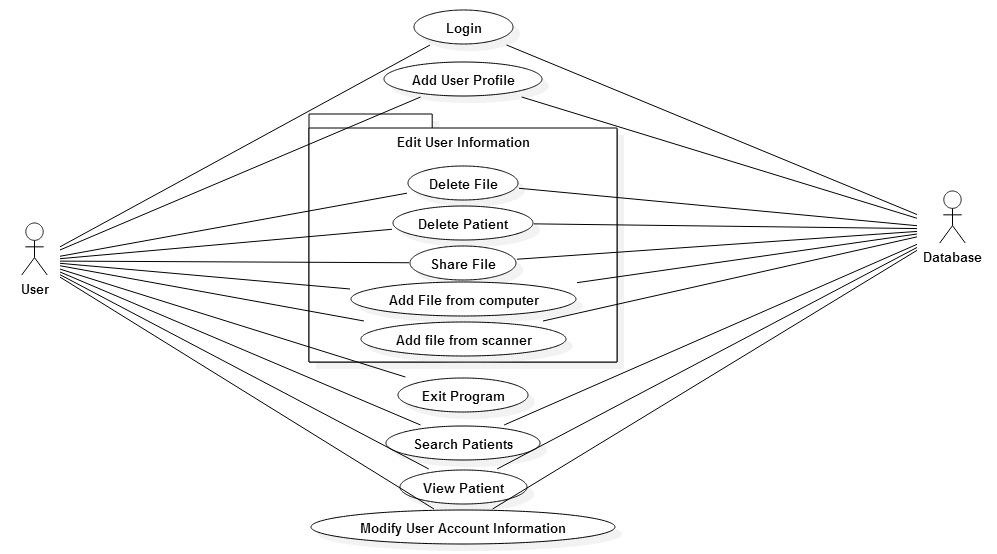
Assumptions:

* The program will be run by a medical office
* The program will be run on a windows operating system
* The program will be run on a machine with the following specification:
  + Processor - dual core @ 2.4 GHz
  + RAM - 4 GB
  + Hard Drive - 160 GB
* The user will follow HIPAA compliancy

## ***Stakeholders:***

* Medical Office
  + Uses the program
* Medical Patient
  + Information is stored
* Insurance Companies
  + Clients information is stored
* The Government
  + HIPAA Compliancy

## ***Use Case Diagram:***



# **Use Case Descriptions:**

* Login
  + Securely sign in to the program
* Add User Profile
  + Create a new patient profile
* Edit User Information
  + Delete File
    - Remove a file from a patient profile
  + Delete Patient
    - Remove a patient
  + Share File
    - Send a copy of the file
  + Add file from computer
    - Add a patient file stored within the user’s computer
  + Add file from scanner
    - Scan a file, then add the file to a patient’s profile
* Exit program
  + Log out and close the program
* Search Patients
  + Filter through patient database via search terms and settings
* View Patients
  + View all patients
* Modify User Account Information

Edit user settings

**File Management  
Use-Case: Adding a patient profile**

**Brief Description**

This use case describes how the user utilizes the file management system to add a new patient’s information.

**Actors**

User

**System**

**Preconditions**

User is successfully logged into the system.

**Basic Flow of Events**

1. The use case begins when the “Add Patient” button is activated
2. A “Patient Demographics” window is loaded onto the screen
3. The user inputs patient billing information
   1. Name, Phone Number, Address, SSN, Date of Birth, Insurance ID if applicable
4. The user clicks the “Save” button
5. The system validates user input
6. The system saves all changes.
7. The system uploads all changes to the database
8. Use Case: Modify Patient information
9. The use case ends successfully

**Alternative Flows**

**Invalid User Input**

If in step 5 of the basic flow Add Patient use case: user input is not valid, then

1. The system informs the user of invalid input
2. The system redirects to step 3

**Program exits**

If in any step the user exits the program, then

1. Program exits
2. The use case ends

**No database connection**

If while the system saves, the system is not connected to the database, then

1. The system will locally store data
2. When the system has connection with the database
3. The system uploads all data

**Post-conditions**

**Successful Completion**

The user has successfully added a new patient.

**Failure Condition**

The data has not been uploaded to the database or saved.

**File Management  
Use-Case: Modify Patient Information**

## ***Brief Description***

This use case describes how the User modifies patient information.

## ***Actors***

### User

## ***Preconditions***

User is successfully logged into the system.

## ***Basic Flow of Events***

1. The use case begins when the user selects the “Modify patient information” option within the “Patient Profile” screen or when the “add patient” use case is completed
2. The “Action Buttons” are loaded onto the screen
3. The user selects one of the buttons
   1. Add file from scanner
   2. Add file from computer
   3. Edit billing information
   4. Share File
   5. Exit to Main Menu
4. The user selects the “Exit to Main Menu” button
5. The system saves all changes
6. The system uploads all changes to the database
7. The “Main menu” screen is loaded
8. The use case ends successfully

## ***Alternative Flows***

### The use case ends with a failure condition Alternate Button Selected

#### Add file from scanner

If in step 3 of the basic flow the button selected by the user is “add file from scanner,” then

1. The system loads the scanner utility
2. The system grabs the scanned file
3. The system moves the file to the local system storage
4. The system saves all changes
5. The system returns to step 2

#### Add file from computer

If in step 3 of the basic flow the button selected by the user is “add file from computer,” then

1. The system loads file selection utility
2. The system moves the file to the local system storage
3. The system saves all changes
4. The system returns to step 2

#### Edit Billing Information

If in step 3 of the basic flow the button selected by the user is “Edit Billing Information,” then

1. Load the “patient demographic” screen
2. The user edits patient billing information
   1. Name, Phone Number, Address, SSN, Date of Birth, Insurance ID if applicable
3. The user clicks the “Save” button
4. The system validates user input
5. The system saves all changes.
6. The system uploads all changes to the database
7. The system returns to step 2

#### Share File

If in step 3 of the basic flow the button selected by the user is “Share File,” then

1. The system prompts the user to log in
2. The system verifies user
3. The user selects files to share
4. The user types the recipients email
5. The user types in the passcode for the file
6. The file is locked and loaded to the email
7. The user selects “send”
8. File is sent
9. The user must share the passcode with the recipient
10. The system returns to step 2

### Program exits

If in any step the user exits the program, then

1. Program exits
2. The use case ends

### No database connection

If while the system saves, the system is not connected to the database, then

1. The system will locally store data
2. When the system has connection with the database
3. The system uploads all data

### Post-conditions

#### **Successful Completion**

The user has successfully added a new patient.

#### **Failure Condition**

The data has not been uploaded to the database or saved.

# **SECTION 3: Specific Requirements**

## *3.1 Functional Requirements:*

|  |
| --- |
| No: 3.1.1 |
| Statement: The program shall perform validity checks on inputs |
| Dependency: none |
| No: 3.1.2 |
| Statement: The program shall not need the internet, but can use the internet |
| Dependency: none |
| No: 3.1.3 |
| Statement: The program files shall be encrypted based on HIPAA guidelines |
| Dependency: none |
| No: 3.1.4 |
| Statement: The program shall be constructed using defensive approaches |
| Dependency: none |
| No: 3.1.5 |
| Statement: Patient data shall have its own viewing page |
| Dependency: none |
| No: 3.1.6 |
| Statement: The user shall be able to search for patients |
| Dependency: none |
| No: 3.1.7 |
| Statement: The user shall be able to add patient files |
| Dependency: none |
| No: 3.1.8 |
| Statement: The user shall be able to send securely patient files to other healthcare professionals |
| Dependency: none |
| No: 3.1.9 |
| Statement: The program shall implement a user session timeout system |
| Dependency: none |
| No: 3.1.10 |
| Statement: The program shall restrict log in capabilities on successive failed log in attempts |
| Dependency: none |
| No: 3.1.11 |
| Statement: The program shall allow users to change passwords |
| Dependency: none |
| No: 3.1.12 |
| Statement: The program shall allow two types of user access, Admin & User |
| Dependency: none |
| No: 3.1.13 |
| Statement: The program shall prompt the user to change the password after one month |
| Dependency: none |
| No: 3.1.14 |
| Statement: The user shall be able to annotate all patient file types |
| Dependency: none |
| No: 3.1.15 |
| Statement: The program shall send the patient billing info to the billing form automatically |
| Dependency: 3.4.6 |

## *3.2 Interface Requirements:*

|  |
| --- |
| No: 3.2.1 |
| Statement: The system shall use PDF, JPEG, and Text files for patient data display and sharing |
| Dependency: none |
| No: 3.2.2 |
| Statement: The program files shall be stored in an SQL database |
| Dependency: none |

## *3.3 Physical Environment Requirements:*

|  |
| --- |
| No: 3.3.1 |
| Statement: The program shall be run on a Windows 8 or higher operating system |
| Dependency: none |
| No: 3.3.2 |
| Statement: The program shall be run on machine with at least 4 GB of ram |
| Dependency: none |
| No: 3.3.3 |
| Statement: The program shall be run on machine with at least 160 GB of storage |
| Dependency: none |
| No: 3.3.4 |
| Statement: The program shall be run on a machine with at least a 2.4 GHz Dual Processor |
| Dependency: none |
| No: 3.3.5 |
| Statement: The machine executing the program shall not be used outside |
| Dependency: none |
| No: 3.3.6 |
| Statement: Scanner or other image capture device shall be compatible with Windows 8 operating system |
| Dependency: none |

## *3.4 Users and Human Factors Requirements:*

|  |
| --- |
| No: 3.4.1 |
| Statement: The system shall support Authorized user access |
| Dependency: 3.4.2 |
| No: 3.4.2 |
| Statement: Authorized users shall have a username and password |
| Dependency: 3.4.3 |
| No: 3.4.3 |
| Statement: Passwords shall be encrypted in a database |
| Dependency: 3.5.1 |
| No: 3.4.4 |
| Statement: Upon deleting a file the program shall require admin authorization |
| Dependency: none |
| No: 3.4.5 |
| Statement: User shall verify validity of data inputs |
| Dependency: none |
| No: 3.4.6 |
| Statement: User shall be prompted for Patient information upon patient addition to the system |
| Dependency: none |

## *3.5 Data Requirements:*

|  |
| --- |
| No: 3.5.1 |
| Statement: The system shall use a secure SQL database |
| Dependency: none |
| No: 3.5.2 |
| Statement: The data shall not be openly stored without a form of encryption |
| Dependency: 3.1.3 |

## *3.6 Quality Requirements:*

|  |
| --- |
| No: 3.6.1 |
| Statement: The system shall have database access every day the program is used |
| Dependency: none |

*This page last modified by Andrew Henk, Sam Ramirez, and Monica Acosta on 3/23/15*