Software Design Document for Smart Movement Recognition

Nour Tadros, Sara Mohamed , Nourhan Ali , Maha Fayez

Supervised by Dr Mosatafa , Eng. Huda Mohammed

February 2019

1 Introduction

1.1 Purpose

This software design document describes the architecture and system design of Smart Movement Recognition.In this project, we detect the patients activity as falling and tremor by using the accelerometer sensor. Also,we detect the hand gesture for blind. Every gesture has a related move that will help blind people to deal with things easily. The accelerometer API will have access to the accelerometer using real-time classification for stream data from accelerometer, then according to this classification alert will be sent to doctors/nurses with notifications. The doctor/nurse must accept the notification when they reach the corresponding patient. It will consist of one mobile application accessible with any android phone. Furthermore, this document will illustrate the features and purposes of this project.

1.2 Scope

Smart Movement Recognition system aims to help doctors to detect and predict abnormal behavior the patient will do as falling and tremor. Also, hand gestures is made for blind to help them to deal with things easily.

1.3 Overview

Detecting patient abnormal activity and hand gesture for blind are an important researches nowadays, as most of the hospitals need this application to help them to take care of patients by a mobile application instead of continuous monitoring the patient. Moreover, most of the housing for the blind need this application to facilitate their life and help them live in easier way.

2 System Overview

Smart Movement Recognition is to detect patient's activity and helping him rapidly if something went wrong. Detecting the abnormal behavior of patient play a vital role in hospitals and will help doctors and nurses to be more careful to the patients. The project aims to enhance the accuracy of detecting patient abnormal behavior as falling and tremor, through a mobile application using the accelerometer sensor. This sensor will send classified readings to doctors and nurses if anything went wrong to help patient.

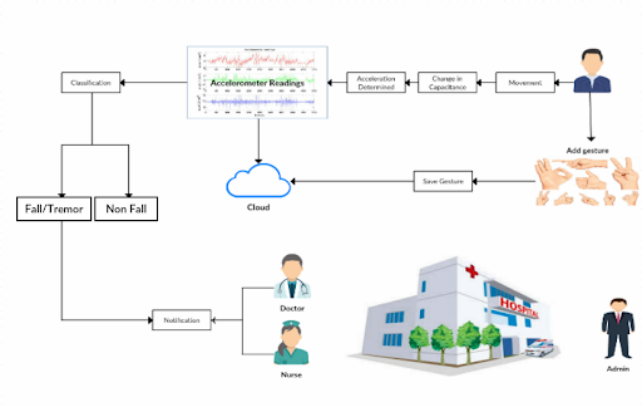


Figure 1: System Overview

Our block diagram

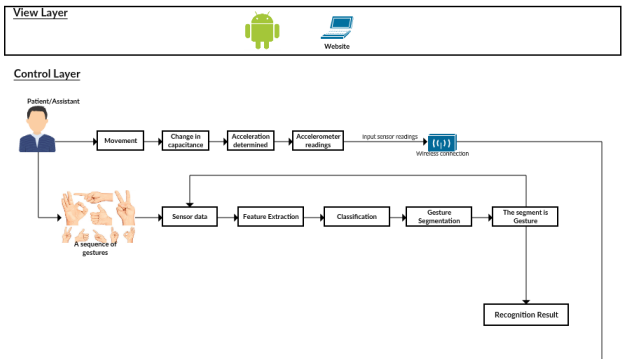


Figure 2: Block Diagram part1

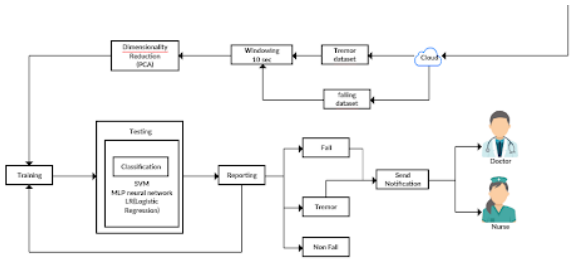


Figure 3: Block Diagram part2

3 System Architecture

3.1 Architectural Design

In our system we have a mobile interface and a wearable device to take reading from patient to track their activities and help patient to submit their gesture. The reading that taken from the a wearable device (Accelerometer sensor) is classififed if patient is fall or tremor happened or if patient make any gesture that refer to something he need or asking for any help, then send notification to doctor/nurse/assistant to help him.

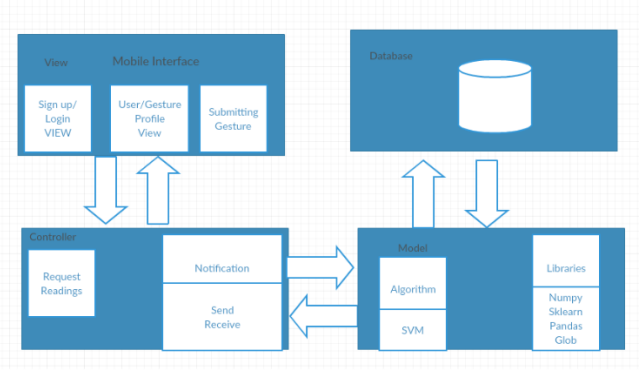
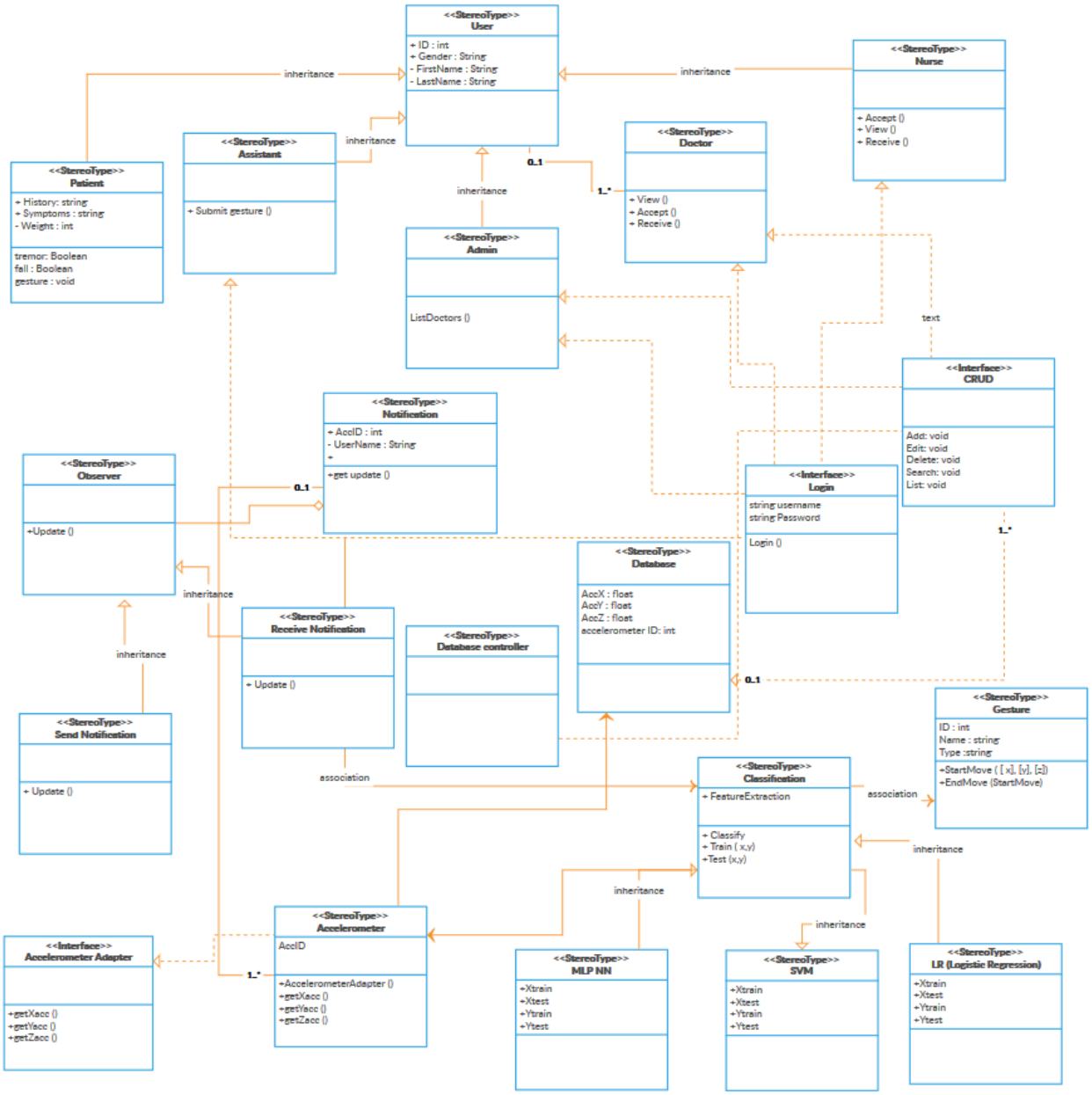


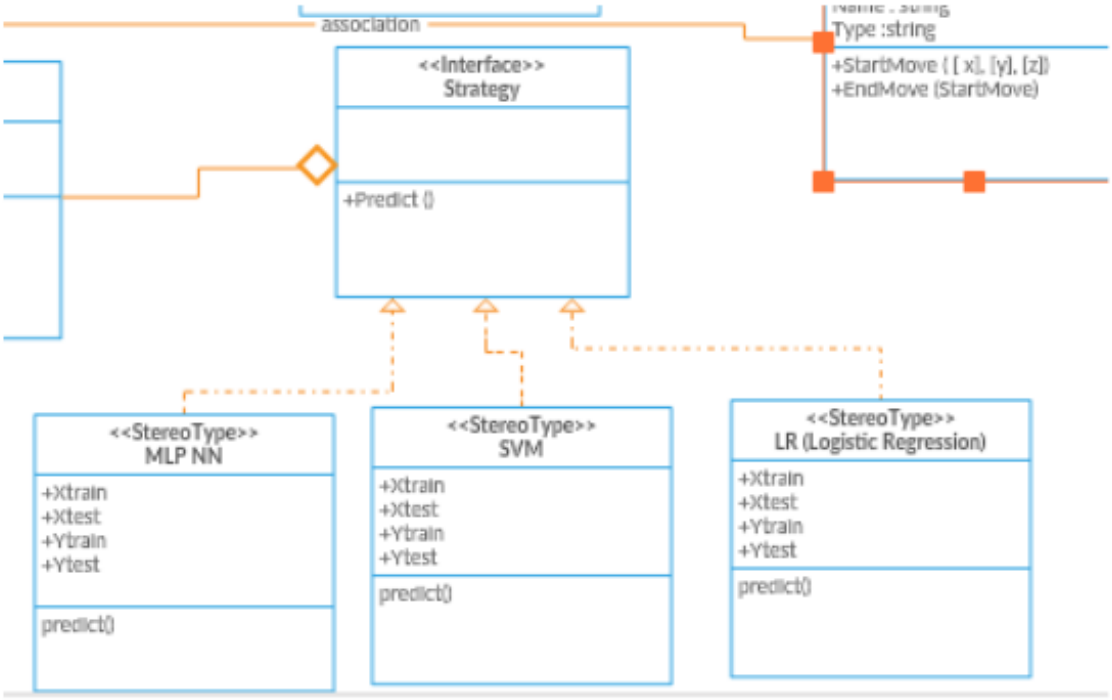
Figure 4: Architectural Design

3.2 Decomposition Description

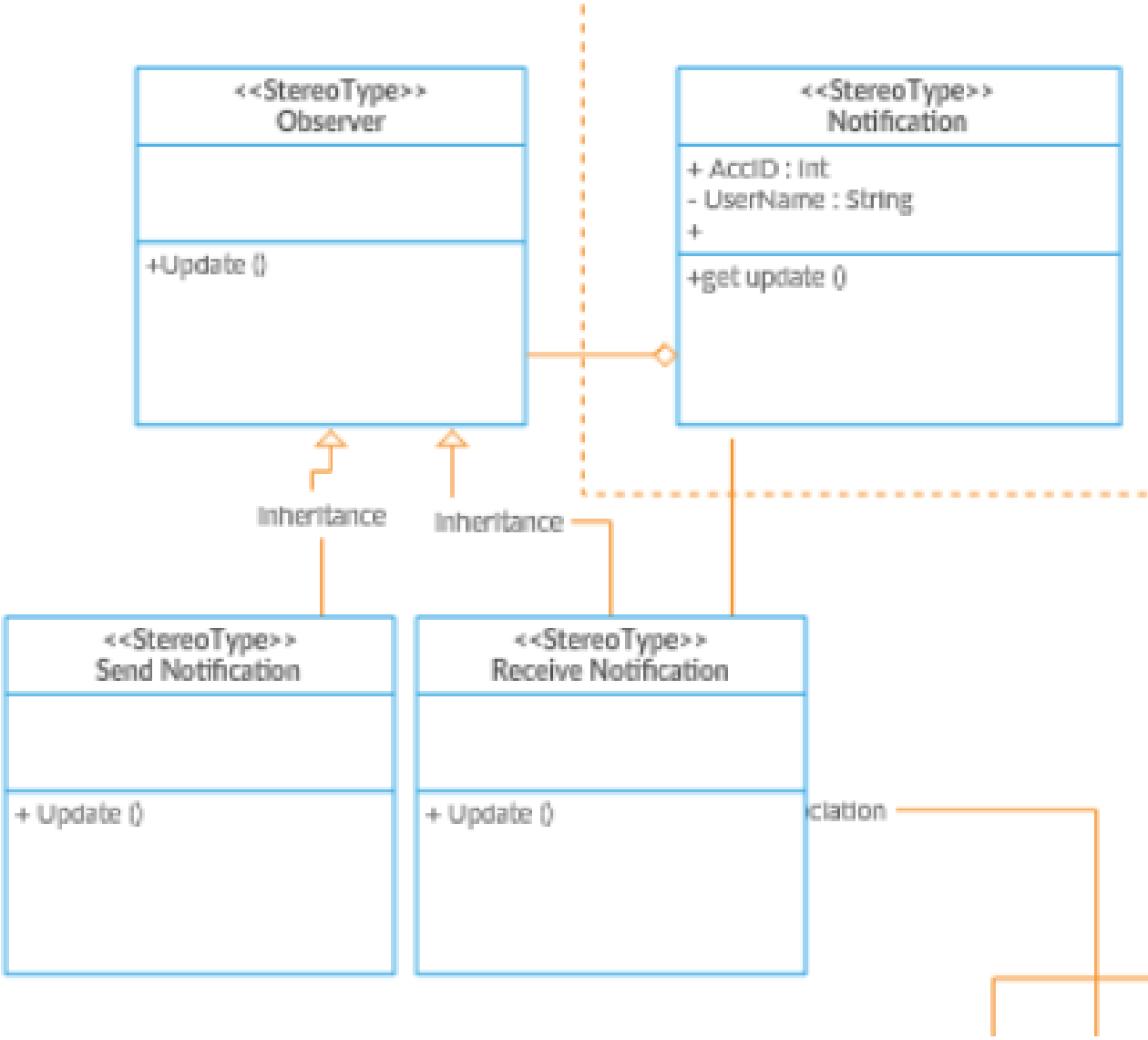
3.2.1 Class Diagram



1. Strategy Design Pattern

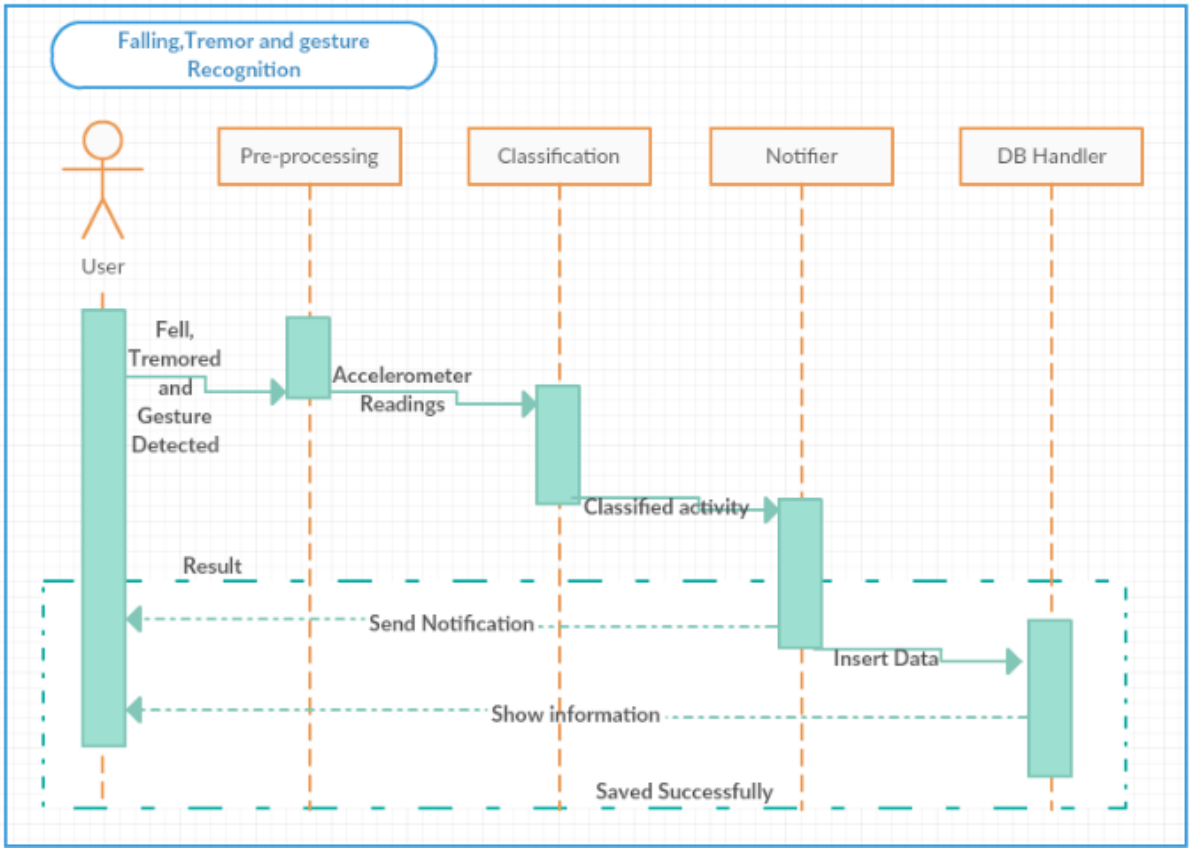


2. Observer Design Pattern



3.2.2 Sequence Diagram

Falling, Tremor and Gesture recognition Sequence diagram



3.3 Design Rationale

Falling, tremor and gesture recognition is based on MVC. it's architecture is in MVC from in figure 1. MVC is consists of Model, View and controller classes. The main advantages of using MVC is for Re usability of code and better maintenance .

4 Data Design

4.1 Data Description

This is a relational preview of the database.

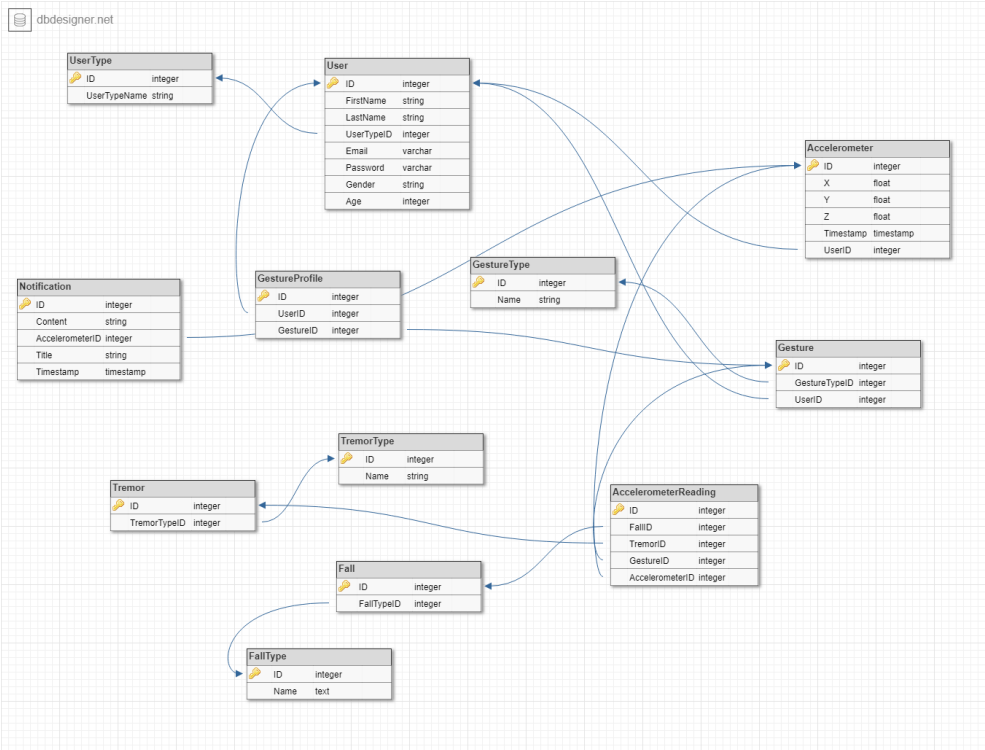


Fig:Database.

4.2 Data Dictionary

- User: This collection has the details of the users of the Smart Movement Recognition. It stores their details like first name, last name, email, password, gender, age ,and a reference to their position.
- User Type: This entity stores the type of the user that we have in Smart Movement Recognition like Doctor, Nurse, Assistant, Patient and Admin.
- Accelerometer:This collection has the details of the accelerometer sensor which stores the x, y, z readings and the timestamp at which those readings were taken. Also, it has a reference to the user owning the sensor.
- Accelerometer Readings: This entity stores a reference to fall, tremor, gesture and accelerometer.
- Fall: This entity stores a reference to the fall type.
- Fall Type: This entity stores the type of the falls we have in Smart Movement Recognition like falling forward, backward or side-ward.

- Tremor: This entity stores a reference to the tremor type.
- Tremor Type: This entity stores the type of tremors we have in Smart Movement Recognition.
- Gesture: This entity stores a reference to gesture type and user.
- Gesture Type: This entity stores the type of gestures we have in Smart Movement Recognition like circle clockwise and anticlockwise, arrow upwards and downwards and other gesture shapes.
- Gesture Profile: This entity stores a reference to user and gesture to know which user has done which gesture.
- Notification: This collection has a details of the notification received in Smart Movement Recognition. It stores the details like description, title, timestamp and a reference to accelerometer.

5 Component Design

5.1 Primitive Results

MobiAct V2 Fall Types with Standing and Laying	Accelerometer and Gyroscope
Classifier	Percentage
Naïve Bayes	77.80%
SVM(gamma='auto')	90.30%
Naive Bayes for Bernoulli models	66.80%
KNN, N=3	97.70%
Logistic Regression	70.40%
MLP Neural Network	86.70%
Random Forest (n_estimator=100)	97.90%

Fig:Primitive results.

5.2 Random Forest

Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of it's simplicity and the fact that it can be used for both classification and regression tasks. it is a supervised learning algorithm. Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.[1]

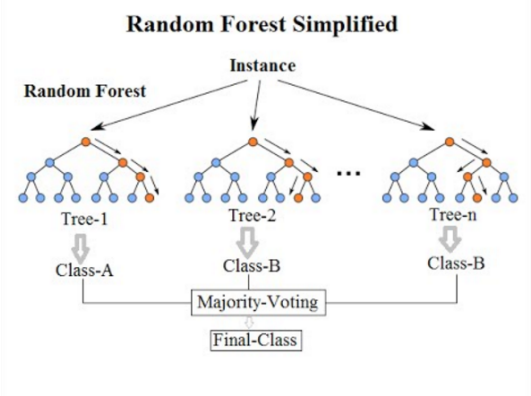


Fig:Random Forest Illustration.[3]

Random forest classifier creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object. [2]

5.3 K-Nearest Neighbour - KNN

K-nearest neighbor is a classification algorithm where a set of examples is collected and each one has a known class. This set is used as a training dataset for the algorithm so when a new data is fed to it a comparison takes place between the training dataset and the new data. Thus the nearest k training datasets to the testing data and its labels are checked. Among the k training datasets that have been selected, the label that is found repeated the most is the label set to the testing data; thus classifying it as that label. KNN has different calculation methods; they are euclidean, manhattan and cosine similarity distances[4]. The KNN works by taking vote of the nearest neighbours as follows:

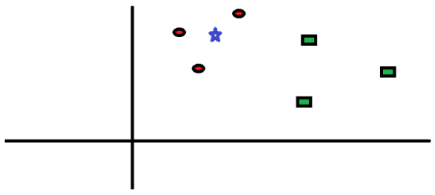


Fig:Scenario1 in KNN.

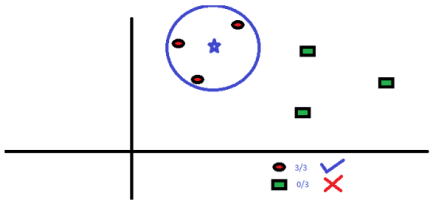


Fig:Scenario2 in KNN.[5]

5.4 Support Vector Machine - SVM

SVM is a classifier that is a powerful machine method developed from statistical learning and has made achievement in a specified field; it is applied on different data that have two or multi classes[5].It has many attractive features and promising empirical performance; as for instance, it does not suffer limitations of data dimensionality and limited samples[6][7]. Performance of SVM is very sensitive to how the cost parameter and kernel parameters are set. For some datasets that are in a 2D view, it is necessary to move it to a 3D view. Mapping the data into a higher dimension is known as kernelling

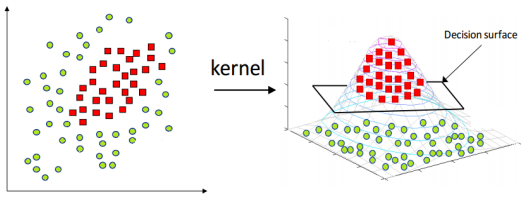


Fig:Kernelling SVM.[8]

The equation of the hyperplane: You must have come across the equation of a straight line as y=mx+c, where m is the slope and c is the y-intercept of the line.

The generalized equation of a hyperplane is as follows:

wTx=0

Here w and x are the vectors and wTx represents the dot product of the two vectors. The vector w is often called as the weight vector.

Consider the equation of the line as ymxc=0. In this case,

$$w = cm1$$

and

$$x = 1xy$$

wTx

$$= c1mx + y = ymxc = 0$$

It is just two different ways of representing the same thing.[8]
SVM is well known for its accuracy and more efficient given that i uses a subset of training points since it works well on smaller, cleaner datasets. On the other hand, it is less effective on noisier datasets with overlapping classes.

6 Humnan Interface Design

6.1 Overview of User Interface

There is an android based mobile application for the Doctor, nurse and assistant, we have one user interface application. First of all we have the signup and login pages that allow the user to add his valid information. Once the user is logged in, (s)he is redirected to the homepage which contains a side menu and the information of the user. All the pages will have a side menu to make it easier for the user to navigate through the pages. There is three main interfaces for the smart movement recognition application: The first one is for the notification alert, and this page appears to the user when the patient has made a certain behavior as falling, it contains (Patient's name, Timestamp and the accelerometer ID). The Doctor/Nurse will receive the notification and be able to accept it through the "Accept" button. The second interface is "Create Gesture" for the hand gestures, there is 3 buttons (Add move, End move and Submit activity), and this activity is done by the assistant of the patient. Firstly, the nurse will click "Add move" button then the patient will do a certain shape and the assistant will click the "End move" button after that "submit activity" button. The last main interface is "Save Gesture" and it is for the gestures to be assigned to an action needed from the patient. We have two drop down lists, the first one is for the open/close action and the second one is for assigning the open/close action to an object then the "submit" button.

6.2 Screen Images

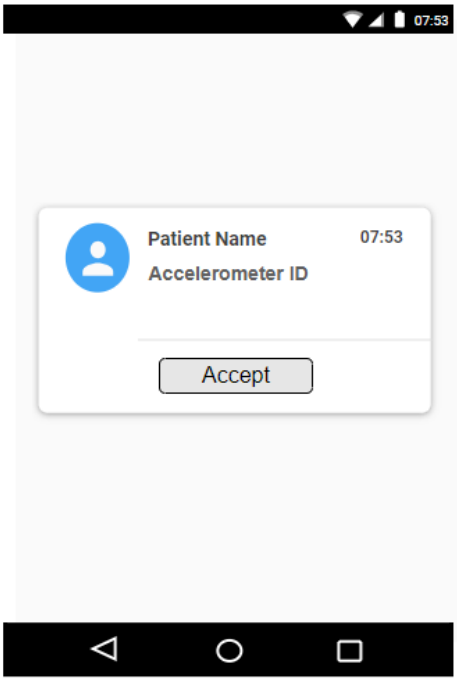


Figure Notification received to doctor.

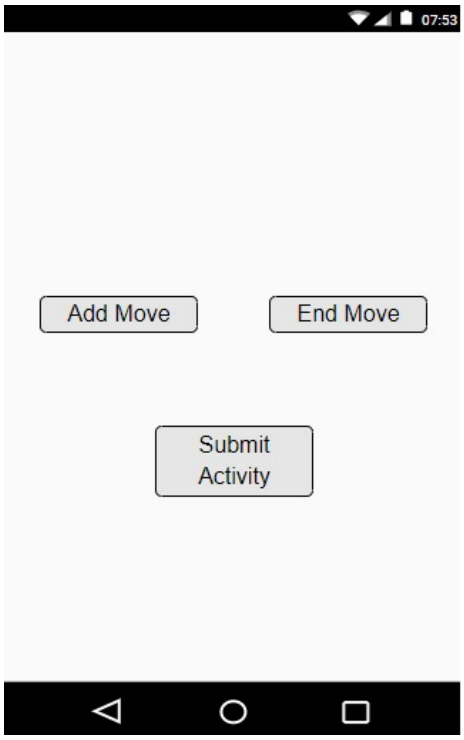


Figure Create new Gesture.

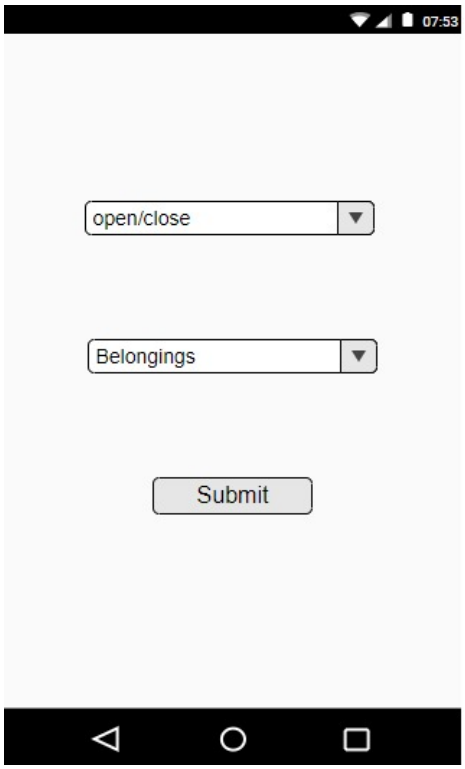
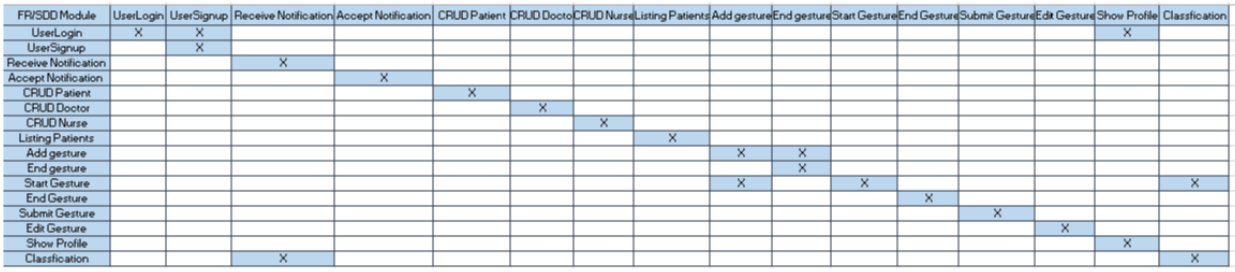


Figure Submit gesture.

7 Requirements Matrix



FR/SDD Module	UserLogin	UserSignup	Receive Notification	Accept Notification	CRUD Patient	CRUD Docto	CRUD Nurse	Listing Patients	Add gesture	End gesture	Start Gesture	End Gesture	Submit Gesture	Edit Gesture	Show Profile	Classfication
UserLogin	X	X													X	
UserSignup		X														
Receive Notification			X													
Accept Notification				X												
CRUD Patient					X											
CRUD Doctor						X										
CRUD Nurse							X									
Listing Patients								X								
Add gesture									X	X						
End gesture										X						
Start Gesture									X		X					X
End Gesture												X				
Submit Gesture													X			
Edit Gesture														X		
Show Profile															X	
Classfication			X													X

Figure 5: fig:Requirements Matrix, mapping between functional requirements and the SDD modules..

References

[1] The Random Forest Algorithm – Towards Data Science Niklas Donges-Niklas Donges - https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd

[2] Chapter 5: Random Forest Classifier – Machine Learning 101 – Medium Savan Patel-Savan Patel - https://medium.com/machine-learning-101/chapter-5-random-forest-classifier-56dc7425c3e1

[3] Random Forest Simple Explanation – Will Koehrsen – Medium Will Koehrsen-Will Koehrsen - https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d

[4] Sadegh Bafandeh Imandoust and Mohammad Bolandraftar. "Application of k-nearest neighbor (knn) approach for predicting economic events: Theoretical background". In: International Journal of Engineering Research and Applications 3.5 (2013), pp. 605–610.

[5] Introduction To Knn, K-nearest Neighbors : Simplified Tavish Srivastava - https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

[6] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. "A training algorithm for optimal margin classifiers". In: Proceedings of the fifth annual workshop on Computational learning theory. ACM. 1992, pp. 144–152.

[7] Vladimir Vapnik. The nature of statistical learning theory. Springer science

business media, 2013.
[8] Simple Tutorial on Svm and Parameter Tuning in Python and R https://www.hackerearth.com/blog/machine
learning/simple-tutorial-svm-parameter-tuning-python-r/