Software Requirement Specification Document for TrainIt System

Habiba Hegazy, Mohamed Abdelsalam,
Mostafa Hussein, Seif Elmosalamy
Supervised By: Dr. Ayman Ezzat, dr Ayman Nabil
and Eng. Youmna Ibrahim

January 25, 2020

1 Introduction

1.1 Purpose of this document

The main purpose of this Software Requirements Specification document is to illustrate and outline the requirements for a table tennis guiding system (TrainIt) that are mainly differentiating, detecting, classifying and analyze various incorrect behaviors in the table tennis strokes movements. TrainIt is considered to be sensor based system that will be build using sensors such as IR depth camera, accelerometer and gyroscope. Our aim is to develop a full guiding system that's supposed to help the athletes/users enhance their performance in playing table tennis. This will be done by working on four different joints in the body (Waist, wrist, shoulder, elbow). This document will provide a fulfilled and detailed description about each stage input, output and algorithms used in this stage. Along with a full illustration for each stage interfaces, hardware, software requirements and development process. And stating down system constraints, what difficulties have we faced during development and how should we interact with it. Moreover, this software requirements specification (SRS) document defines how our stakeholder, team, and audience see the product and its functionality.

1.2 Scope of this document

TrainIt is aimed to target and help those who are interested in learning the correct techniques of playing the table tennis stroke movements. The system is virtually-based real-time that should be very easy to use. The users will get tips and guidance points on how to improve and enhance their performance. Also coaches will be able to keep track of their players improvement. The system will also be helpful for users that aims to self-learn the basics of the game.

1.3 Overview

In TrainIt application we will implement a non-existing system with a high accuracy, speed differentiation and detection of various table tennis stroke movements. The real time monitoring and analyzing of incorrect behaviors in the table tennis stroke movements to enhance the player's movements and techniques. The wearable systems of player should always aim to provide him/her with a trustworthy feedback, which also assists the coaches to obtain the incorrect movements of all players who are monitored. Moreover, the use of and IR depth cameras will be very helpful it obtaining different joins information not only the joint which contains the wearable device. TrainIt goes through three essential stages. Firstly, obtaining real time data from different sensors and passing it to pre-processing phase. Secondly, classifying the data obtained through three classification stages. Finally, classify weather the stroke is correct or incorrect and if it was incorrect where was the problem in the movement.

The application workflow is illustrated below in figure (2). Starting with collecting readings from wearable device accelometer and gyroscope sensors and the IR depth camera. Thereafter, the collected readings is passed through a pre-processing to filter the readings from any noise and get better results by using Kalman filtration and Signal interpolation / extrapolation. The filtered data will then pass to an inner door server will be used for the classification of the movements and detect mistakes, algorithms that might be used is fastDTW, SVM, KNN RCNN, Naive Bayes, Deep learning time sries, one dollar algorithm .. etc. Moreover, Player behavior will be analyzed and stored. Eventually, the analyzed data takes two ways, the rating data is always sent to data storage cloud to show reports later and AR application and vibration alert will be produced to the player when detecting wrong behavior of the strokes. In addition, the cloud speed is not necessary to be fast as it is not used for classification more than a storage space. Moreover, the report can be used by the coach to monitor the ratings of the players by monitor the reports. The system overview is shown in figure (1).

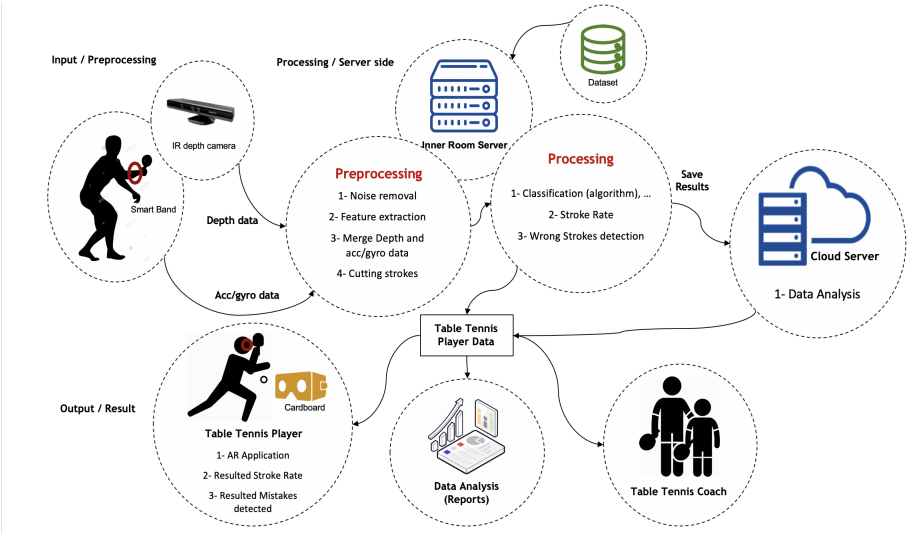

Figure 1: System overview Diagram

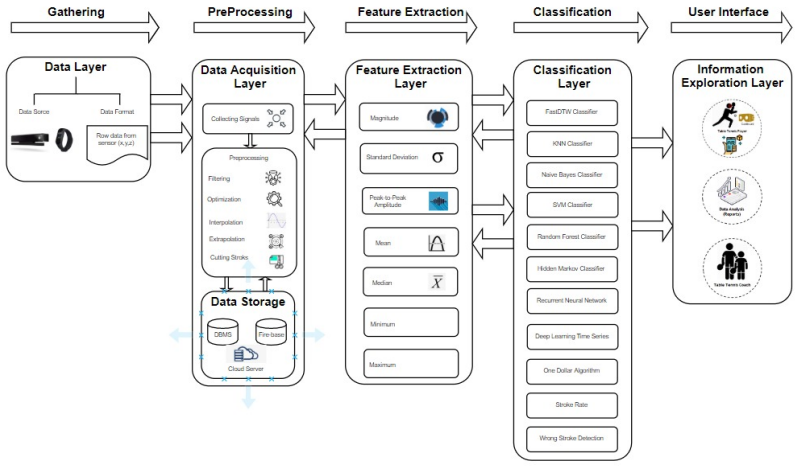

Figure 2: Block Diagram

1.4 Business Context

According to many statistics competitive sports have been the main focus of various spectators all over the world. One of the most popular competitive internationally is table tennis or ping pong [4, 1]. It is being viewed and enjoyed

by more than 291 million viewers [3]. To increase competitiveness, it's very important to focus on the training period for the trainees. In table tennis, it is very difficult to monitor the accuracy of the stroke in the training phase consistently.

Waraporn Viyanon, Vimvipa Kosasaeng, Sittichai Chatchawal and Ab-hirat Komonpetch [6] have developed an android application which analyzesuser's motion data gathered from accelerometer and gyroscope sensors on a smart phone attached on the user's wrist and gives feedbacks to improve user's swings. Peter Blank, Julian Hoßbach, Dominik Schuldhaus and Bjoern M. Es-kofier [2] proposed a detection and classification system by attaching a sensor to the racket itself. Marko Kos, Jernej zenko, Damjan Vla and Iztok Kramberger [5] presents an autonomous wearable IMU device for tennis stroke detection and classification placed on the forearm of the player. The project idea was presented to many coaches and showing their interest and if it will be a successful project they will be a part of this project stakeholder team.

2 General Description

2.1 Product Functions

1. The system will detect joints using Kinect sensor and smart band sensors.
2. The system will detect the whole stroke movements according to different parts of the body.
3. System will have different users (coach and player).
4. Coach can monitor the player stroke shooting behavior.
5. Coach can view a full report about his players.
6. Immediate Alert to the player if there is a wrong. stroke.

2.2 Product Context

2.2.1 Context Diagram

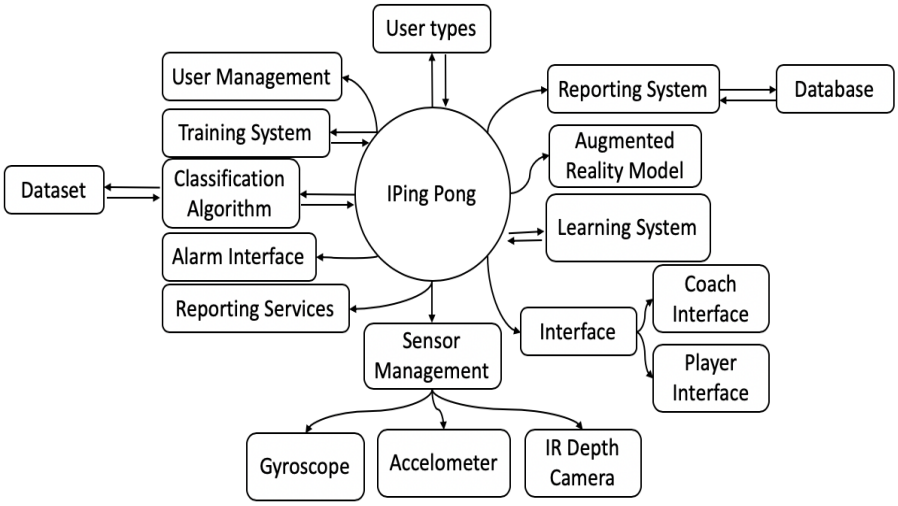

Figure 3: Context Diagram

2.3 Similar System Information

- SwingPong [6]: is an android application on a smartphone analyzing player's forehand and backhand strokes in order to provide helpful suggestion to players. These user's motion data gathered from accelerometer and gyroscope sensors on a smartphone attached on the user's wrist as shown in figure (4) and gives feed backs. This application deals with two main modules; one for storing angular acceleration and angular velocity of strokes. And the other one for providing suggestions as shown in figure (5) depends on the first module to evaluate user's basic strokes and analyzing them.

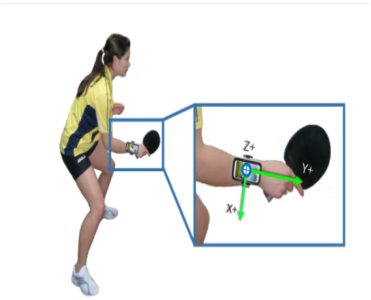

Figure 4: The smartphone attached on a player

Figure 5: Graphical result and suggestion interfaces

- Sensor-based table tennis stroke detection and classication system [2]: a system of attached inertial sensors to table tennis rackets as shown in figure (6) that is able to detect table tennis strokes in time-series data and to classify each stroke into correct stroke type categories. The system has the potential to be implemented as an embedded real-time application for other racket sports, to analyze training exercises and competitions, to present match statistics or to support the athletes' training progress. This system worked on collected data of different basic stroke types from amateur and professional players using multiple classifiers to be compared regarding classifications rates.
- Wearable IMU device for tennis stoke detection and classification [5]: For arm movement acquisition a miniature wearable IMU device, positioned on the player's forearm (right above the wrist) as shown in figure (7). The proposed system is able to detect and classify three most common tennis strokes: forehand, backhand, and serve using a MEMS-based accelerometer and gyroscope with 6-DOF. the system uses the accelerometer data for accurate and reliable tennis stroke detection. And process the gyroscope data for tennis stroke classification.

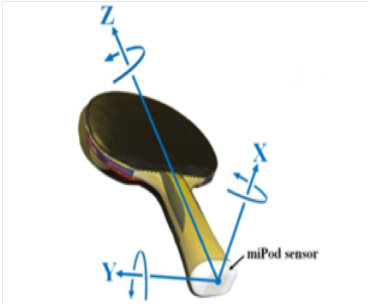

Figure 6: miPod sensor attached to the racket handle

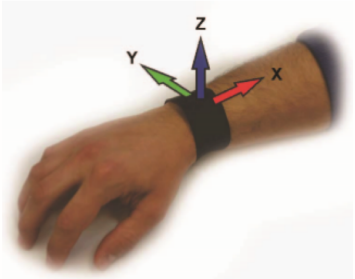

Figure 7: IMU device placement and sensor orientation.

2.4 User Characteristics

1. Coach:
    - Must have basic knowledge in using Android mobile devices and also with augmented reality application.
2. Player:
    - Must have basic knowledge in using Android mobile devices.

2.5 User Problem Statement

Enhance the classification accuracy and provide online real-time feedback for enhancing the player stroke shooting style by classifying the correct and wrong strokes using sensor device and IR depth camera on different body joints.

2.6 User Objectives

By using our system, the coach can view reports of the player not only he is playing wrong but also giving where is wrong of the movements.

2.7 General Constraints

One of the main constraints of the system is the connection between different sensors (kinect and wearable device). In addition, the timing of classification need to be less that a second. However, it difference between a player and the other according to his speed in playing a stroke. Different problems would be challenging the system.

3 Functional Requirements

Title	check Email Validation
ID	FR00
Description	This Function is to check if the email is valid before checking it in the database
Type	Boolean
Input	email
Action	Checks if the email entered in a real exist email and if it email is valid in format. If so it returns true means that the email is valid to be checked if exist in the database
Output	Boolean if email is valid
Precondition	user entered email into the required EditText
Post-condition	return to login function (FR02)
Dependencies	FR02

Title	check Password Validation
ID	FR01
Description	This Function is to check if the password is valid before checking it in the database
Type	Boolean
Input	password
Action	Checks if the password is valid in format. If so it returns true means that the password is valid to be checked if exist in the database with its email
Output	Boolean if password is valid
Precondition	user entered password into the required EditText
Post-condition	return to login function (FR02)
Dependencies	FR02

Title	Login
ID	FR02
Description	This Function is for the user to login into the system using his/her account
Type	Boolean
Input	Email and Password
Action	Check if all data are filled and compare data that was entered to the record in the database, if so function returns true otherwise returns false
Output	Boolean acceptance login
Precondition	the user needs to into his email and password into the EditTexts and that the inputs are verified (FR00, FR01)
Post-condition	Redirect to the home page
Dependencies	FR00, FR01

Title	Forget Password
ID	FR03
Description	This Function was fired up if the user forgets the password, it generates a new password and sends email with it
Type	Boolean
Input	email
Action	first the email is checked using FR00 then if the email exists in the database, the function sends email FR30 with the new password generated
Output	if record updated
Precondition	the user enters his email in the required place
Post-condition	email is sent to the user with new password
Dependencies	FR00, FR30

Function	Check phone number validation
ID	FR04
Description	This Function is to check if the phone number is valid before checking it in the database
Type	Boolean
Input	user's phone number
Action	Checks if the phone number is valid in format. If so it returns true means that the phone number is valid to be checked if exist in the database
Output	Boolean if phone number is valid
Precondition	User entered the phone number into the required EditText
Post-condition	Return to login function (FR02)
Dependencies	FR02

Function	Create User
ID	FR05
Description	The functions get the user type ID according which user is inserting record, then it uses the inputs to insert a new row in Users table in the system database
Type	Boolean
Input	Users object
Action	Checks if the user's email, password and phone number are validated, if so it enters the user record in the database and returns true else it returns false to show error message
Output	if user created
Precondition	the user's information needed to be entered is inserted in its write place and validated
Post-condition	Creating a new row in the database table with new ID for the user
Dependencies	FR00, FR01, FR04

Function	Delete User
ID	FR06
Description	The function fire up a query to remove the selected user row from the Users table
Type	Boolean
Input	ID of the user selected to be removed
Action	check if the user id exists in the database to remove it
Output	if user deleted
Precondition	the user needed to be selected from the list
Post-condition	Updating the database by removing the user id row from the Users table
Dependencies	-

Function	List User
ID	FR07
Description	the function fire up a query lists all the users IDs from Users table that is available in the system database.
Type	Returns Array
Input	user type ID's required to be listed
Action	Retrieving all information about the user
Output	Array by each user's data
Precondition	check if the user exists
Post-condition	-
Dependencies	FR05

Function	Search User
ID	FR08
Description	This Function fires up a query to search for user ID in users table
Type	Returns Array
Input	array of user data
Action	Check the user information and return the user ID that was searched for
Output	Array of user's data
Precondition	Check if the user exists
Post-condition	-
Dependencies	FR05

Function	Update User
ID	FR09
Description	This Function fires up a query to users table for updating the user data according to the user ID
Type	Boolean
Input	user ID to be updated
Action	Check if the user data is updated and return true else it returns false to show error message
Output	if user updated
Precondition	Check if the user exists
Post-condition	update the system database row of the user in users table
Dependencies	FR05

Title	Get Address
ID	FR10
Description	This Function is to get the address of the user and save it in address table in the database
Type	array
Input	-
Action	Check if the address is filled and validated
Output	array of address object
Precondition	Check if the user exists
Post-condition	Inserting an address record with the ID of user in address table in the database
Dependencies	FR05

Title	Create User Type
ID	FR11
Description	This Function is to add new user type to the system in the database
Type	Boolean
Input	User type name
Action	Checks if the user type name is validated, if so it enters the user type record in the database and returns true else it returns false to show error message
Output	if usertype created
Precondition	Check if the user type exists
Post-condition	Creating a new row in the database table with new ID for the user type
Dependencies	-

Title	Delete User Type
ID	FR12
Description	This Function is to delete the selected user type row from user type table in database
Type	Boolean
Input	ID of the user type selected to be removed
Action	Check if the user type id exists in the database to delete it
Output	if usertype deleted
Precondition	the selected user type needed to be removed
Post-condition	Updating the database by removing the user type id row from the user type table
Dependencies	-

Title	List User Types
ID	FR13
Description	This function for listing all the user types that exists in the database
Type	returns array
Input	-
Action	Retrieving all user types in the database
Output	array by each user type
Precondition	check if the user type exists
Post-condition	-
Dependencies	FR11

Title	Update User Types
ID	FR14
Description	This Function is for updating the user types according to the ID if there were any changes or modifications
Type	Boolean
Input	User type ID to be updated
Action	Check if the user type is updated and return true else it returns false to show error message
Output	if user type updated
Precondition	check if the user type exists
Post-condition	Update the system database row of the user type in user type table
Dependencies	FR11

Title	Create Player
ID	FR15
Description	This Function called after FR05 which adds a new player record to the system's database from the coach module
Type	Boolean
Input	Player object
Action	Check if FR05 is fired and that all fields in the Player object are filled then insert a new record to the database and returns true
Output	if player created
Precondition	the player's information needed to be entered is inserted in its write place and validated
Post-condition	A new player's record is created with new ID for the player
Dependencies	FR05

Title	Delete Player
ID	FR16
Description	This function called after FR06 which deletes the player record in the database
Type	Boolean
Input	Player object
Action	Check if FR06 is fired and The player's record exists in the database to delete it
Output	player's record is deleted and returns true, else returns false to show error message
Precondition	select the player to be deleted
Post-condition	The record of the player is deleted from the database
Dependencies	FR00 , FR01 , FR05 , FR06 , FR10

Title	List Players
ID	FR17
Description	This function is called after FR07 which lists the players records in the database
Type	return array
Input	player object to be listed
Action	Check if FR07 is fired and list all the information about the player
Output	Array by each player's data
Precondition	Check if the player exists
Post-condition	-
Dependencies	FR00 , FR01 , FR05 , FR07 , FR10

Title	Search Player
ID	FR18
Description	This Function called after FR08 which searches for player's record in the database
Type	return array
Input	player object
Action	Check if FR08 is fired up and then search for the player's ID
Output	Array of player's data
Precondition	Check if the player exists
Post-condition	-
Dependencies	FR00 , FR01 , FR05 , FR08 , FR10

Title	Update Player
ID	FR19
Description	This Function called after FR09 which updates the existing player's information in the database
Type	Boolean
Input	Player object
Action	Check if FR09 is fired up and The player data is updated
Output	if user is updated returns true, else it returns false to show error message
Precondition	Check if the player exists
Post-condition	The record of the player is updated in the database
Dependencies	FR00 , FR01 , FR05 , FR09 , FR10

Title	Start training
ID	FR20
Description	functions classifies the strokes and get all strokes played and store with its timestamp
Type	Void
Input	start Training TimeStamp and array of strokes
Action	if inputs does not equal to null then functions starts classification and storing data into database
Output	-
Precondition	player press start training button
Post-condition	stroke classified and player notified and database records are added
Dependencies	-

Title	End Training
ID	FR21
Description	functions stops (FR20) and get out a full report about the performance through training period
Type	Object
Input	end Training TimeStamp and array of strokes
Action	if FR20 is fired, the function stops it
Output	object of Report class
Precondition	FR20 to be fired
Post-condition	a report will be viewed to the coach and player
Dependencies	FR20

Title	View Profile
ID	FR22
Description	This Function is for the user to view his/her profile that contain all the data existed in the database
Type	return array
Input	User Type object
Action	Check if FR11 is fired and retrieve all the data of the user
Output	Array by each user's data
Precondition	Check if the user exists
Post-condition	-
Dependencies	FR11

Title	Create Club
ID	FR23
Description	This Function is to add new club record in the database to assign it to players
Type	Boolean
Input	Club object
Action	Check if the club's name , address and rate are validated and if so it inserted in the database and returns true else it returns false to show error message.
Output	if club created
Precondition	The club's information should be validated
Post-condition	Creating a new club record in the database
Dependencies	-

Title	Delete Club
ID	FR24
Description	This Function is for the admin to delete club who no longer assigned to players in the database
Type	Boolean
Input	ID of the club selected to be deleted
Action	Check if the club record existed and delete it
Output	if club deleted
Precondition	-
Post-condition	The database is updated by deleting the club id row from the clubs table in the database
Dependencies	-

Title	Update Club
ID	FR25
Description	This function is to update the existing information that related to the clubs that the players assigned to
Type	Boolean
Input	Club ID to be updated
Action	Check if the club data is updated and return true, else it returns false to show error message
Output	if user updated
Precondition	Check if the club exists
Post-condition	Update the club table in the database of the system
Dependencies	FR19

Title	View Club
ID	FR26
Description	This function is to list all the clubs in the database that are assigned to the players
Type	return array
Input	Club object
Action	Retrieving all the information related to each club in the database
Output	Array by each Club's data
Precondition	Check if the club exists
Post-condition	-
Dependencies	FR19

Title	Send Notifications
ID	FR27
Description	This Function is for the coach to send notifications to the player
Type	String
Input	Notification object
Action	Check if the notification is sent to the player else error message will appear
Output	String notification is sent or error message
Precondition	Player must be existed in the database
Post-condition	Notification is created and inserted in the database.
Dependencies	FR10

Title	View Notifications
ID	FR28
Description	This Function is for listing all the notifications in the database
Type	return array
Input	-
Action	Retrieving all the notification in the database
Output	Array of notifications
Precondition	check if the notifications exist
Post-condition	-
Dependencies	FR15

Title	Delete Notifications
ID	FR29
Description	This Function is for the coach if he wants to delete notification from the database
Type	Boolean
Input	Notification object
Action	Check if the notification exists in the database and delete it
Output	Acceptance message if the record of notification is deleted, else false to show error message
Precondition	select the notification to be deleted
Post-condition	The record of notification in database is deleted
Dependencies	-

Title	Send Message
ID	FR30
Description	the functions send message by different methods according from which class it was called
Type	void
Input	array of users
Action	if the array is not empty the function sends massage by different methods
Output	-
Precondition	the user press on the send button
Post-condition	notifications are sent
Dependencies	-

Title	Intake Stroke Signals
ID	FR31
Description	This function is for collecting sensor readings (x,y,z)
Type	Float Array
Input	Accelerometer and Gyroscope (x,y,z) and time stamp
Action	Sending x,y,z readings into filtering method
Output	Array of Accelerometer and Gyroscope data
Precondition	Check if there is a duplicate of readings and time stamp
Post-condition	Calculate the magnitude of (x,y,z) and update into the database
Dependencies	-

Title	kalman Filteration
ID	FR32
Description	This function is for filtering sensor readings (x,y,z) using kalman filter
Type	Float Array
Input	Array of magnitude points
Action	Sending array into filtering method
Output	Array of Data filtered
Precondition	Check array to be cut into windows
Post-condition	Array is updated into the database
Dependencies	FR31

Title	cutting Stroke
ID	FR33
Description	This function is for cutting the array into number windows to send to be classified
Type	Array
Input	Array of filtered points
Action	Sending array of (x,y,z) into filtering method
Output	Array of windows, each window is a stroke
Precondition	Check id data is filtered
Post-condition	Array is updated into the database
Dependencies	FR31, FR32

Title	Stroke Merge
ID	FR34
Description	This function is responsible for merging the data from kinect with the data from the smart band
Type	Array
Input	Array of optimized windows from kinect and array of point from smart band
Action	Combining the two arrays from kinect and smart band to one array for each stroke
Output	Array of point (x,y,z) for each stroke
Precondition	Get the data from kinect and smart band
Post-condition	array is uploaded into database
Dependencies	FR31

Title	Classify
ID	FR35
Description	This Function is for classifying player stroke by comparing the array of optimized strokes from FR33 and FR32 with the matched templates from the dataset
Type	Stroke Classified object
Input	Array of testing data
Action	If the classifier result close to any of the templates from the dataset then the result will be returned. Otherwise, the stroke will be classified as unknown movement
Output	Stroke Classified result
Precondition	Array of cutted strokes(FR38) and filtered strokes (FR37)
Post-condition	Database is updated and stroke marked as classified
Dependencies	FR36 , FR37 , FR38

Title	Create Report
ID	FR36
Description	This function is for the coach to create a full reports about the players after training periods
Type	Void
Input	Player object and Stroke Classified object
Action	If the inputs are not empty the report will be generated
Output	-
Precondition	The coach choose player to create report on
Post-condition	Report will be shown on the screen for the coach
Dependencies	-

Title	Calculate Performance
ID	FR37
Description	This function calculate the the performance of the player according to the results of the correct strokes and wrong strokes played
Type	Performance object
Input	Object from player results
Action	Calculate the number of correct stroke and wrong and calculate the total performance of the player
Output	Object from performance
Precondition	Check if the player has result record is database
Post-condition	Calculate the player performance
Dependencies	FR15, FR20

4 Interface Requirements

4.1 User Interfaces

4.1.1 GUI

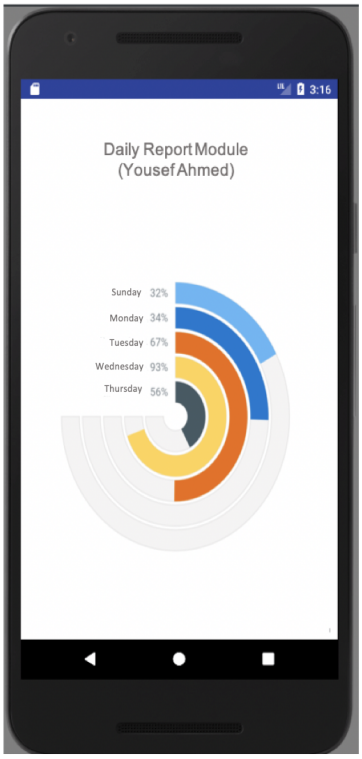

Figure 8: Daily Report about the player performance

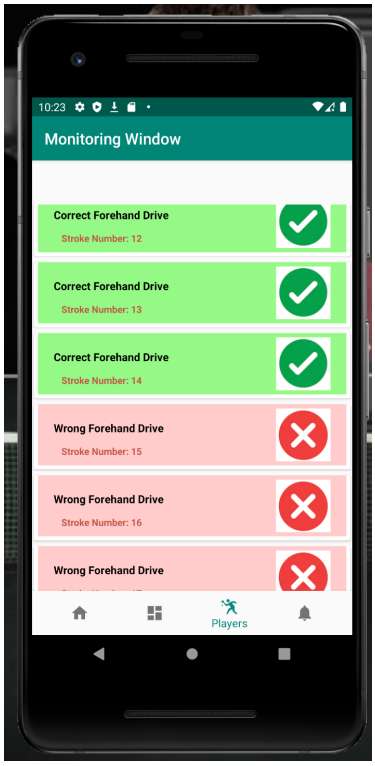

Figure 9: Real-time monitoring Window about player movements

Figure 10: Augmented reality glasses

4.1.2 API

1. Google Sign-in, Sign-up
2. Facebook Sign-in, Sign-up
3. SMS Messages
4. Email Messages
5. API connecting android app and inner server
6. Google Report API
7. API connecting Kinect with inner server

4.1.3 Python Libraries

1. numpy Libraries
2. fastdtw Libraries
3. scipy Libraries

4.2 Communications Interfaces

The communication interface is one of the most important requirements of our software as it will need a connection to the internet or a local host connection.

5 Performance Requirements

The system shall be able to process a huge number of points (x,y,z) per window. The system also must be able to handle large datasets to ensure accuracy. Our system requires fast machines to handle the many techniques of the correct and wrong strokes.

6 Design Constraints

Any smart mobile device that include the android operating system and must have connection with the internet.

6.1 Hardware Limitations

Mobile device must have the accelerometer and gyroscope sensor.

6.2 others as appropriate

7 Other non-functional attributes

7.1 Performance and Speed

As the system deals with many techniques of playing, it must be interactive and the detection and classification must have no delays to give a reliable online feedback.

7.2 Scalability

The system should be highly scalable as we are dealing with many techniques of playing and training for table tennis strokes.

7.3 Usability

Proportion of functionalities or tasks mastered doesn't need time to be learned. Also, this system is easy to be memorized due to the small number of tasks the user will do.

7.4 Reliability

The system is reliable. As it is mainly focusing on detection and classification of the strokes. Sensor readings should be very accurate dealing with big amount of data. When a wrong detected stroke is being classified, it is important for the system to identify the type of this wrong stroke. The user should be able to trust the system as it aims for high accuracy to insure the correct and good experience of training.

7.5 Maintainability

The system ensures ease of maintainability through the implementation of MVC. It should be easy to maintain to minimize the amount of changes that would be done to the code.

7.6 Portability

The system could be implemented on different operating systems such as Android or iOS.

8 Preliminary Object-Oriented Domain Analysis

8.1 Class Diagram

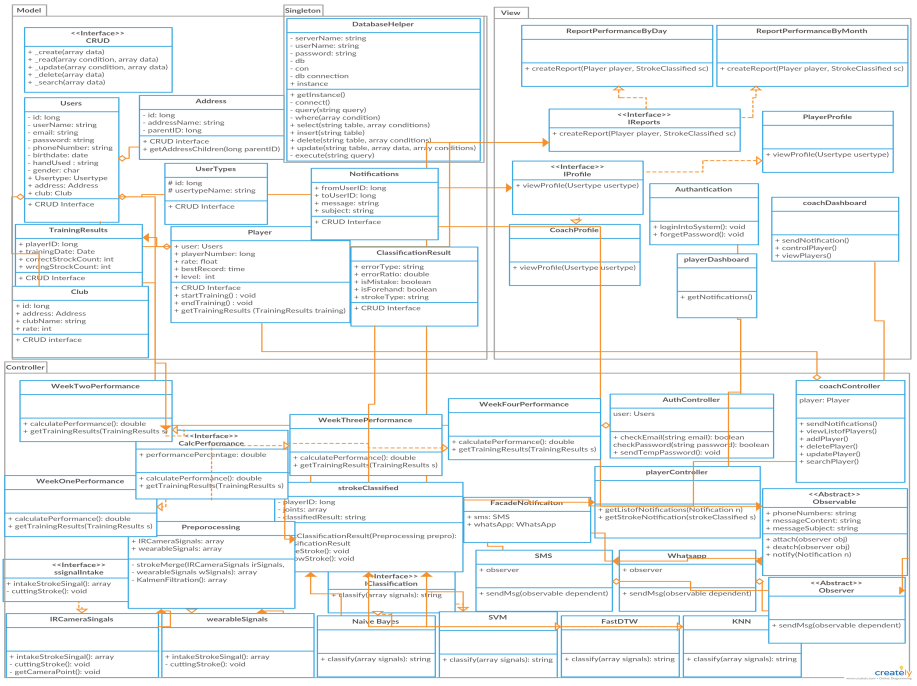

Figure 11: Class Diagram

8.2 Database Schema

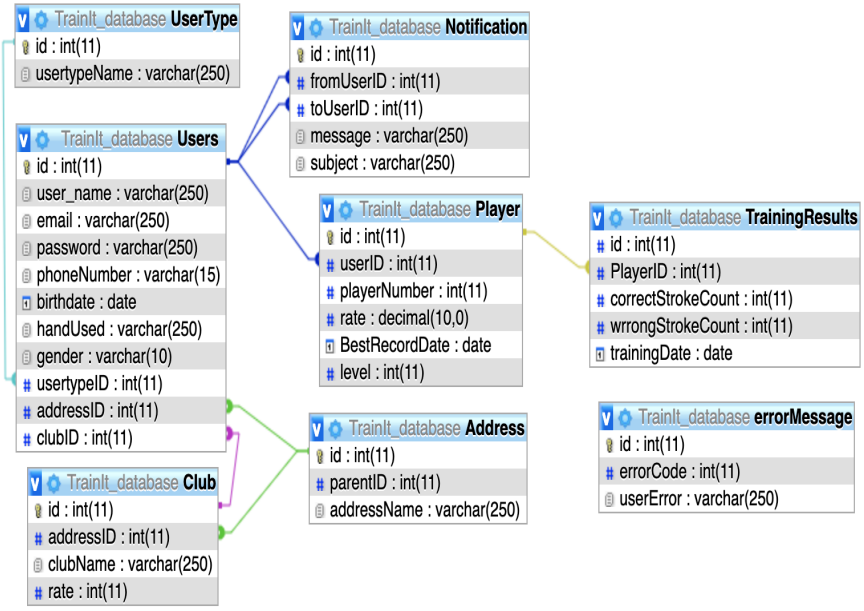

Figure 12: Database Schema

The Current database is made for a single club, and by using MVC and design patterns we can open the application to be for any user.

9 Operational Scenarios

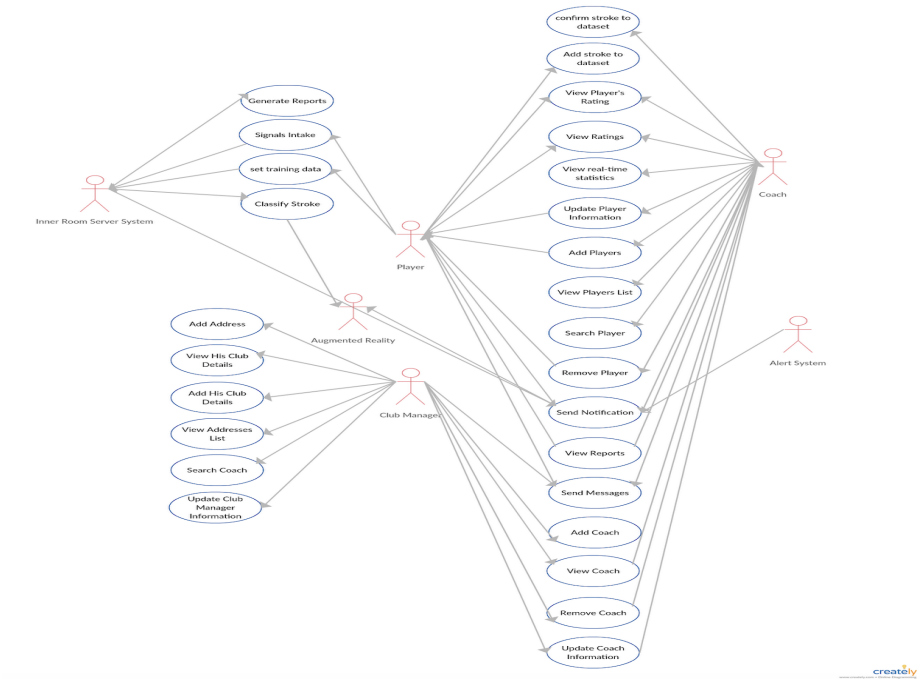

Figure 13: Usecase Diagram

9.1 Scenario 1

A Beginner table tennis player wants to practice his skills more to be professional. So he got his Kinect IR and his tennis table racket and wore his smart watch. Then he started playing the correct forhand push stroke, each time he does a correct play the smart watch vibrate once, and with each false play it vibrates twice so he have to keep playing it again till he does it correctly.

9.2 Scenario 2

A table tennis coach wants to watch all of his players and track their performances, at the same time it's hard to be reached, because there is no other coaches with him. So he will use the mobile application where he can monitor all of the players and detect how many correct and false strokes each player did.

9.3 Scenario 3

A table tennis player wants to practice alone after the training, so he can easily put the set up, play and analyze himself and monitor his performance after

playing for a couple of hours through the mobile application that shows his analytics and full report.

9.4 Scenario 4

In a table tennis tournament the judges many players can get the same score but there is only one award for the most correct movements player, so the judges wanted an automated system to help them take more accurate decisions in giving their decisions that is not based on his score but on the body movements, so they can use the app as a 3rd party judge to tell them the percentage of correct movements each player did.

10 Preliminary Schedule Adjusted

Task	Start	End
Idea discussion	26-07-2019	11-08-2019
Idea Research	11-08-2019	11-09-2019
Proposal presentation	08-10-2020	08-10-2020
Dataset collection	10-10-2019	01-11-2019
Implementing System (1/2)	02-11-2019	03-12-2019
Writing paper (1)	19-10-2019	29-11-2019
Delivering Paper (1/2)	18-12-2019	18-12-2019
Designing Application	01-01-2020	07-01-2020
Implementing GUI Design	08-01-2020	10-01-2020
Implementing Application	18-01-2020	25-01-2020
Designing Database	08-01-2020	09-01-2020
Designing Class Diagram	06-01-2020	07-01-2020
SRS writing	12-12-2019	13-01-2020
SRS presentation	15-01-2020	15-01-2020
Implementing System (2/2)	20-01-2020	03-03-2020
SDD writing	01-02-2020	20-02-2020
SDD presentation	25-02-2020	25-02-2020
Validation and Testing	10-02-2020	10-02-2020
Writing paper (2)	30-03-2020	10-04-2020
Delivering Paper (2/2)	10-04-2020	30-05-2020
Writing Thesis	21-04-2020	31-05-2020
Final presentation	26-06-2020	26-06-2020

Figure 14: Timeline

11 Preliminary Budget Adjusted

Item	Quantity	Cost
Smart Band (having 3 axis gyroscope and accelerometer)	1	$41.99
Kinect	1	$200
Google Cardboard	1	$14.74
Amazon Cloud Drive	-	5GB for $11.99/year

Table 1: Different classification algorithm comparison

References

[1] "Table tennis: number of participants u.s. 2017." [Online]. Available: https://www.statista.com/statistics/191959/participants-in-table-tennis-in-the-us-since-2006/

[2] P. Blank, J. Hossbach, D. Schuldhaus, and B. Eskofier, "Sensor-based stroke detection and stroke type classification in table tennis," 09 2015, pp. 93–100.

[3] Y. Caifeng, "2018 world team championships the most followed table tennis team event in history," Sep 2018. [Online]. Available: https://www.ittf.com/2018/09/13/2018-world-team-championships-followed-table-tennis-team-event-history/

[4] Ioc, "Table tennis started as a genteel, after-dinner game, but is now a fast, high-tech sport. it also has the most participants of any sport in the world." Sep 2019. [Online]. Available: https://www.olympic.org/table-tennis

[5] M. Kos, J. Ženko, D. Vlaj, and I. Kramberger, "Tennis stroke detection and classification using miniature wearable imu device," 05 2016.

[6] W. Viyanon, V. Kosasaeng, S. Chatchawal, and A. Komonpetch, "Swing-pong: analysis and suggestion based on motion data from mobile sensors for table tennis strokes using decision tree," 12 2016, pp. 1–6.