|  |
| --- |
| **VIA CLUB (group 11)** |

**Students:**

**Vaclav Dvorak, 285049**

**Alejandra Leticia Gonzalez Roldan, 285133**

**Ramanpreet Singh, 285506**

**Jose Martinez, 285046**

**Supervisor:**

**Mona Wendel Andersen**

**Alan Rune Henriksen**

**ICT - Software Engineering**

**1st Semester**

**1.6.2019**

**Table of content**

[Abstract iv](#_Toc9928473)

[1 Introduction 1](#_Toc9928474)

[2 Requirements 4](#_Toc9928475)

[2.1 Functional Requirements 4](#_Toc9928476)

[2.2 Non-Functional Requirements 6](#_Toc9928477)

[3 Analysis 7](#_Toc9928478)

[4 Design 12](#_Toc9928479)

[5 Implementation 14](#_Toc9928480)

[6 Test 19](#_Toc9928481)

[6.1 Test Specification 19](#_Toc9928482)

[6.1.1 Functional testing 19](#_Toc9928483)

[6.1.2 Non-Functional testing 22](#_Toc9928484)

[6.2 Test Specifications 23](#_Toc9928485)

[7 Results and Discussion 24](#_Toc9928486)

[8 Conclusions 25](#_Toc9928487)

[9 Project future 26](#_Toc9928488)

[10 Sources of information 27](#_Toc9928489)

[11 Appendices 1](#_Toc9928490)

# Abstract

The purpose of this project is to provide customer (VIA Club, soccer club, represented by their manager) with a Sport Management System in form of computer application, that would allow customer to work more efficiently when creating, editing and storing Players and Squads. Second part of the project is to deliver website, that would be attractive for club fans.

Functionalities, that cover customer requirements, have been collected via interview with the customer (reference to appendices) and further transformed into in requirements, followed by UML diagrams such as Use Case diagrams, Activity diagrams, Sequence diagrams and Analysis Class Diagram.

For developing/transforming the requirements into a code, Java language has been used. As part of the code, GUI has been implemented while using FXML. Data is being stored in .bin files.

The website part of the project has been coded in HTML with using CSS for styling.

The whole project has been driven via Iteration methodology.

*An abstract is a shortened version of the report and should contain all information necessary for the reader to determine:*

1. *What are the aim and objectives of the project*
2. *What are the main technical choices*
3. *What are the results*

*Frequently, readers of a report will only read the abstract, choosing to read at length those reports that are most interesting to them. For this reason, and because abstracts are frequently made available to engineers by various computer abstracting services, this section should be written carefully and succinctly to have the greatest impact in as few words as possible.*

*Although it appears as the first section in a paper, most report writers write the abstract section last.*

*Cf. (Dawson 2009, p.195).*

# Introduction

VIA Club is an outdoor soccer club with a veteran sport manager(customer) leading the team on his own without using any Sport Management System application.

As a result, to this day all tasks are being handled manually. In general, that means creating, editing and keeping the players and their attributes, the squad lists, the match lists manually. To be more specific, the manager(customer) needs to write down the squad of players, both start and substitutes players, for upcoming match just on a paper. That can easily result in lost of the paper and therefor all the work. In order to set the squad properly, the manager(customer) needs to keep all related player data in his head. That can easily result in forgetting the data or missing some interesting point of view on such a data. On top of that, editing the data, that has been put on the paper once, has proven ineffective and chaotic, especially since it is useful to create even future squad lists for future games at the same time. Since VIA Club plays several leagues at the same time, the manager(customer) needs to keep in mind what type of match is upcoming, because the type of match can change the rules in terms of how many substitute players can be on the bench. Similar problem occurs while keeping list of players and their data such as if a player has been injured. As before, it is all stored on a piece of paper, which results in same problems as mentioned above.

There is also a separate problem in a form of outdated team webpage, which is not representable anymore and there is a risk that it would not be attractive enough for potential new fans

Reality of today’s professional sport is, that it has become more about business rather than sport itself and good sport manager(customer) needs a proper Sport Management System in a form of a computer application in order to be avoid as many problems as possible and be successful (Ramesh, 2016). Representative web site as a tool of attracting new fans is always a welcome accessory (View, 2016).

As a conclusion of above mentioned, the objective of this case is to:

1) provide the VIA Club’s manager(customer) with a Sport Management System in a form of a computer application, that would allow him to do all the activites mentioned above

2)provide VIA Club’s sport manager(customer) with new representative webpage

Both above mentioned objectives are seperated into individual requirements in [Requirements](#_Requirements) section of this document. [Analysis](#_Analysis) section then shows these requirements in form of activity and use case diagrams. Design Class and Sequence diagrams then further transform requirements into code level.

*The purpose of the introduction is to provide background information and set the scene for your project. Within which business or organization are you doing the project? Who are the stakeholders and who is the customer?*

*The background information is adapted from your project description where you have already described the problem domain. Describe the current situation and existing context. Your statements must be supported by references to reliable and relevant sources.*

*This should lead to why this project is relevant and outline your aim and objectives. Which technical problems and challenges will be presented in this report, again taken from your project description. System illustrations and rich pictures are welcome here.*

*State delimitations relevant for your project in the introduction. Delimitations include what the project will not cover in relation to your project description, i.e. what could have been expected in your project. Remember that you can only make delimitations to aspects mentioned in the project description and you must argue well for your delimitations.*

*The last sentences of the introduction should be an overview of the sections to follow. This will be a good transition to the next sections.*

*Remember: You must ensure a clear connection between sections in the project report, from Project Description, Requirements, Analysis, Design, Implementation to Test. This means that everything that is implemented can be found in design, everything that is designed is based on the analysis, and anything that is found in analysis has a clear link to requirements, etc.*

# Requirements

Based on the interview (reference to appendices) with the manager(customer, both Sport Management System application and the webpage will be able to provide with following functional and non-functional requirements:

*The purpose of the requirement section is to define functional and non-functional requirements. Requirements are perceived as a contract with the stakeholders (customer), and are specified to ensure a common understanding.*

*Identify the users and describe their roles (e.g. actor descriptions, personas and scenarios).*

*Note: Remember that all requirements must be precise and testable.*

*Use the SMART principle (YourCoach n.d.) and MoSCoW (Business Analyst Learnings 2013).*

*Present a numbered and prioritised list of all the requirements of the users, customer and stakeholders for the project.*

## Functional Requirements

**NUMBER DESCRIPTION PRIORITY**

**(High, Medium, Low)**

R01 User must be able to access LIST OF High

PLAYERS and see it’s content.

R02 User must be able to edit a PLAYER High

and his information in the LIST OF PLAYERS.

R04 User must be able to access SQUAD LIST High

and see it’s Content.

R05 User must be able to create a SQUAD of High

players for a match where user specifies

information:

1.match date and time,

2.name of the opponent,

3.type of the match (league, cup, friendly),

4.players that will start on the pitch,

5.players that will start on the bench

R06 User must be able to edit a SQUAD of players High

and it’s information in SQUAD LIST.

R07 User must be able to create new PLAYER High

in the LIST OF PLAYERS and his information:

1.name,

2.number,

3.position,

4.if suspended,

5.and for how long,

6.if injured

R08 User must be able to delete a PLAYER Medium

in the LIST OF PLAYERS.

R09 While creating/editing SQUAD, system Medium

shows hard validation (application

will not allow user to save such a SQUAD)

when adding:

1.more than 4 players that will start on the

bench for *type of match: league*.

2.more than 5 players that will start on the

bench for *type of match: cup*.

3.how many players will start on the

bench for *type of match: friendly*.

R12 While creating/editing SQUAD system shows Medium hard validation (application will not

allow user to save such a SQUAD) when adding:

1.player that will start either on the bench or

players that will start on the pitch *if suspended*

and at the same time *type of match: cup/league*

2.*type of match: friendly* accepts player even

*if suspended.*

R13 While creating/editing PLAYER, system Medium

shows hard validation (application

will not allow user to save such a PLAYER)

if:

a) number is not filled

b) there is already existing Player with the same number

## Non-Functional Requirements

Non-functional requirements include following:

**NUMBER DESCRIPTION PRIORITY**

**(High, Medium, Low)**

R03 Website design/content looks per design High

R10 Graphic User Interface looks per design Medium

R11 There is only one user role for the Medium

application

Functional requirements represent desirable functionalities that the Sport Management System application needs to provide. In order to analyze the requirements properly, they need to be translated into UML diagrams.

# Analysis

To further analyze the above mentioned functional requirements, they have been transformed into Use Case diagram. Diagram shows 7 Use Cases in total:

1.Access List of Players

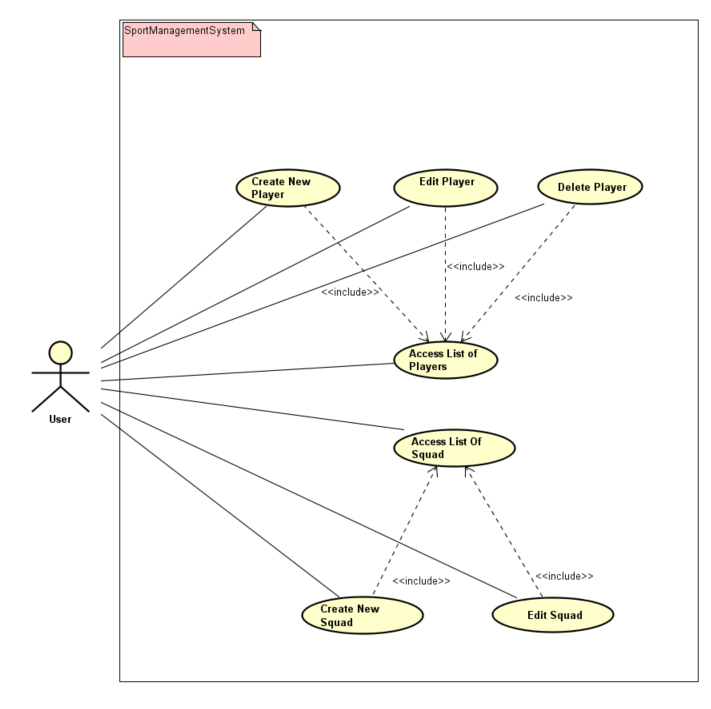
2.Create New Player

3.Edit Player

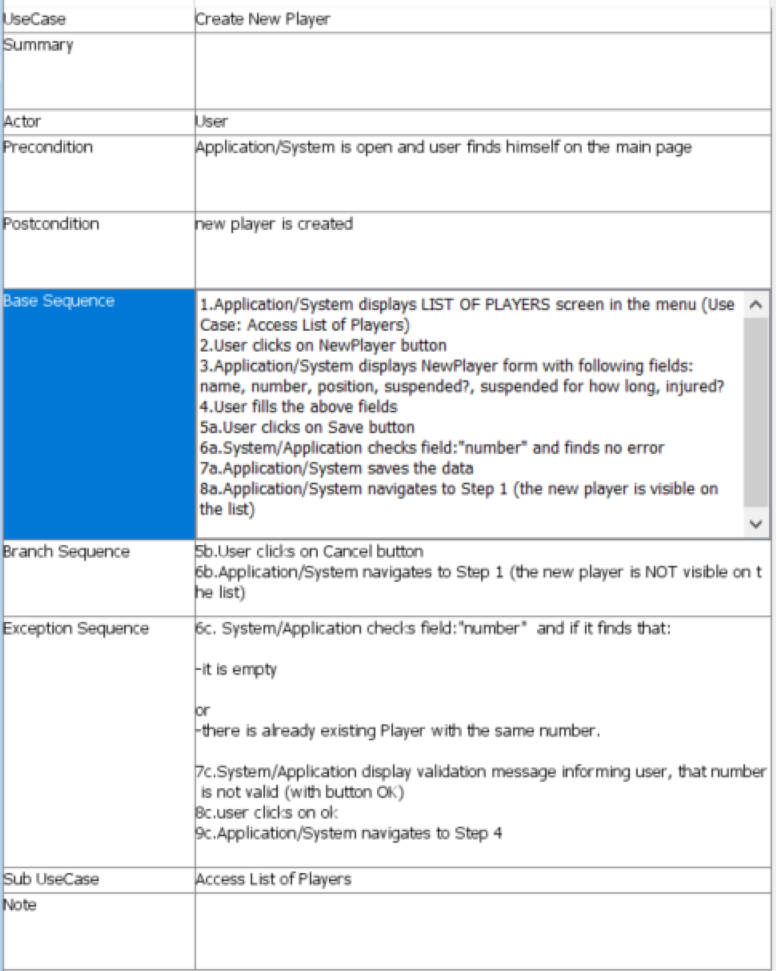
4.Delete Player

5.Access List of Squads

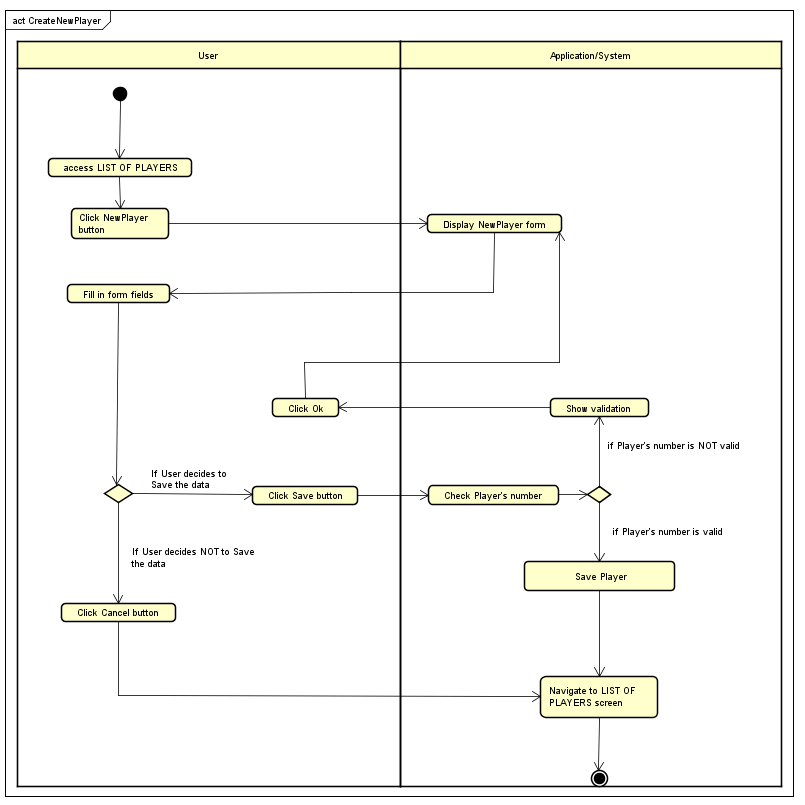
6.Create New Squad

7.Edit Squad

The Key element in the whole diagram has been identified as **Player,** because both Squads and List of Players are ultimately based on one or more **Player**s as their key information. For that reason, the description of Use Case “Create New Players” is displayed below (for whole Use Case diagram please see appendices)



From Activity diagram perspective, the above Use Case has been translated into following diagram. The diagram describes activities (and their order) on both User and Sport Management System application (change the picture, the label “Application/System”--🡪 Sport Management System Application)sides, that need to performed in order to create a Player in the Sport Management System.



For rest of the Activity Diagrams please see [Appendices](#_Appendices)

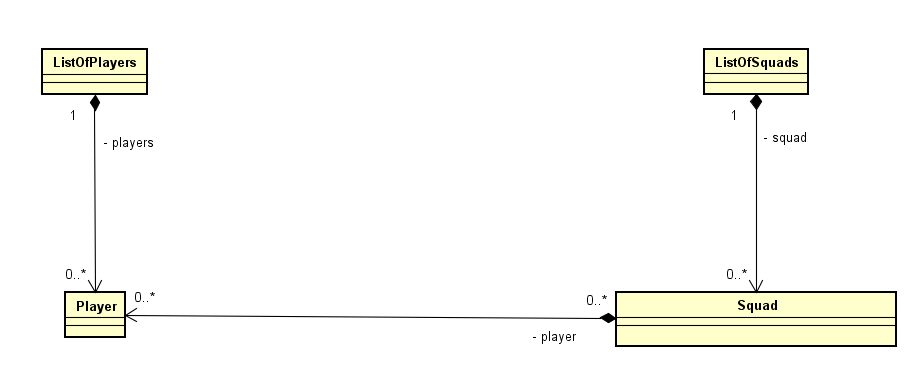
Domain class model builds on Use Case and Activity diagrams and shows the Sport Management System application from domain perspective. That includes basic classes and their relationships. The core of the whole application includes classes:

-*Player*

-followed by *ListOfPlayers*, that gathers the players

-*Squad*, that also includes their own *ListOfPlayers*

-and *ListOfSquads*, that gathers the squads



*The purpose of the analysis section is to outline an understanding of the problem domain and specifically WHAT the stakeholders want. Here, you elaborate on your background description.*

*You identify objects in the problem domain that will be involved in the solution and how these objects cooperate. The result of this analysis is a Domain Model (Larman 2004, chap.9) and other relevant diagrams.*

*Use the UML standard for all diagrams where relevant.*

*Note: Remember that all implementation dependent objects are not part of the domain model only conceptual classes related to the requirements and the domain.*

# Design

All the classes need to translated into a proper class model in order to develop the code properly. In order to do that, following diagrams are introduced:

-Class Diagram

Following diagram represents all the classes. For convinience purposes, all GUI classes are represented by one GUI class

-Sequence Diagram

Sequence diagram was created for only one method that is being perceived as one of the most important ones by the whole group:

For the code development Java language was used while coding in Eclipse application.

Scene Builder application served as a tool for developing Graphic User Interface(GUI).

For modelling diagrams the Astah application was used.

The purpose of the design section is to outline HOW the system is structured; i.e. to transform the artefacts of the analysis into a model that can be implemented. The design section is relevant for the programmer, whereas the analysis is relevant for the stakeholder.

Elements that may be relevant in this section:

* Architecture: Find architecture patterns here (Leszek Maciaszek 2004, chap.9).
* Technologies: Describe technologies used, also alternative technologies. Argue for choice of technology according to the project aim.
* Design Patterns: Describe which design patterns (GoF (Gamma et al. 2002) etc.) you are using and why.
* Class Diagrams
* Interaction Diagrams
* UI design choices
* Data models, persistence, etc.

You must explain all diagrams in the report. These diagrams including descriptions are the blueprints for the implementation.

Hint: One way to figure out which objects/classes are needed in the design is to apply the General Responsibility Assignment Software Patterns/principles (GRASP) (Larman 2004, chap.17).

Hint: Consider how to design your system to make it testable.

# Implementation

Implementations began after finishing first drafts of Class Diagrams. At that point, it was evident, that additional changes of diagrams (mainly Use Case, Activity and Class diagrams) will be necessary. But in order to figure what changes exactly, the development of key classes had to start.

The key classes included:

-ListOfPlayers

-ListOfSquads

-Player

-Squad

These are the classes that take care of building key stones of the application.

Once these Classes were developed, additional classes followed:

-Testing Class with main method.

This class is not part of the application itself, it’s only purpose is to test the code it would be evident if there are some code issues(bugs) as soon as possible.

The development then continued with coding other additional classes, which also had only supporting purpose and were only suppose to help test the code while the ongoing development:

-TextFile Class, that reads test data(squads, players) from text files

-Loading Class with main method, that loads testing data from text files and converts it to .bin files

These classes were used for testing later on, when GUI was ready

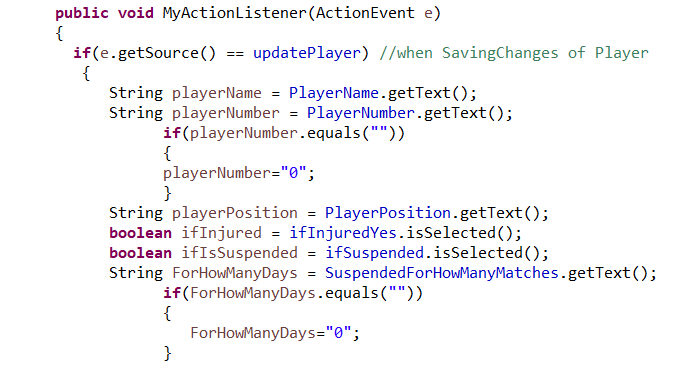
-the GUI elements are coded while using FXML (in Scene Builder application) and are implemented by Controller class along with methods that listen to GUI events.

-PlayersSquadFileAdapter class then servers as a middle layer between the Controller class and the rest of the application.

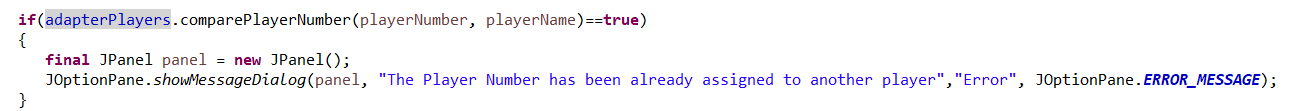
One of the interesting parts of the code, that ilustrates above mentioned class architecture, would be GUI button „updatePlayer“. The whole process linked to this button goes:

*Controller* listens, untill „updatePlayer“ button is pushed. After that,

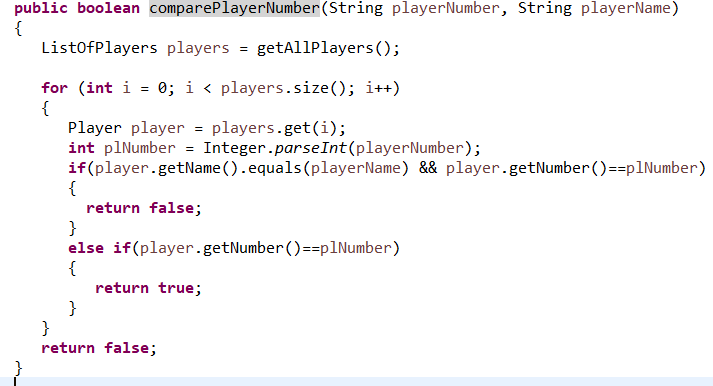
form fields values, that represent selected player’s attributes, are taken and assigned to local variables.



Object of *PlayersSquadFileAdapter* class then first takes two of these variables: *playerNumber* and *playerName* as arguments in method comparePlayerNumber()



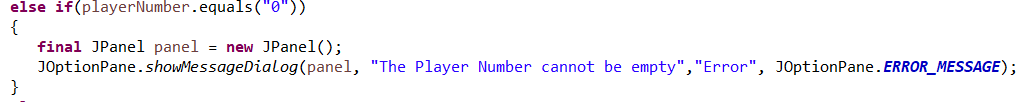
Object of *PlayersSquadFileAdapter* and it’s met*hod comparePlayerNumber() then* creates object of class *ListOfPlayers* with all the existing players by method getAllPlayers(). And then checks, if there is already a player with the same *playerNumber* (other than the player for which this whole process was started in the first place) by calling class *Player* methods getName() and getNumber().



If there is a *playerNumber* duplicity, then *comparePlayerNumber()* returns true. At which point *Controller* shows validation



If there is no *playerNumber* duplicity, then *Controller* checks if the *playerNumber* value equals “0” and if so, then shows another validation

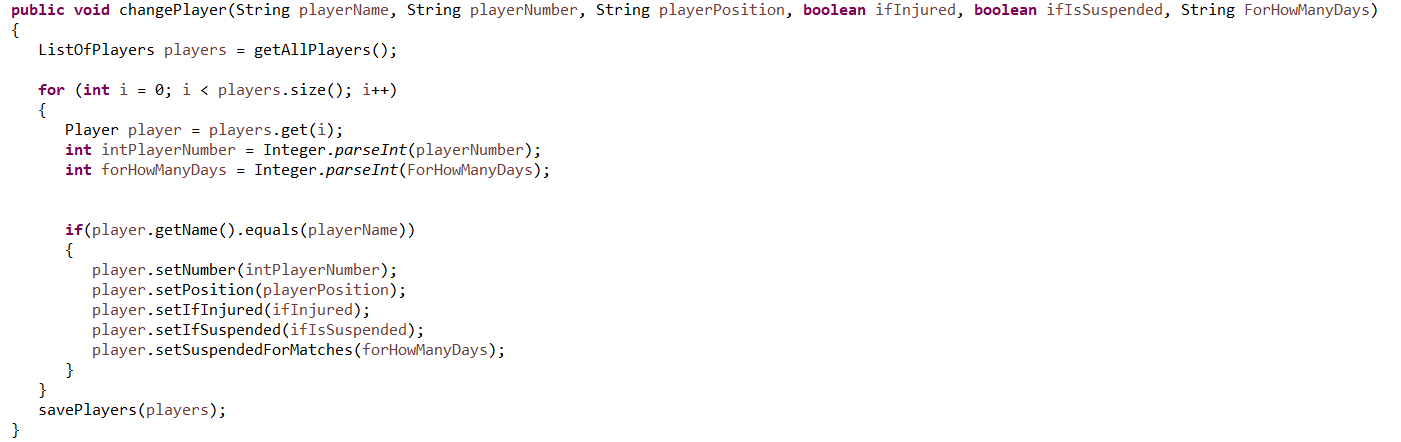


In case there is no duplicity nor “0”, *Controller* calls object of *PlayersSquadFileAdapter* class again and it’s method *changePlayer(),* but this time with all the local variables that represent player’s attributes.



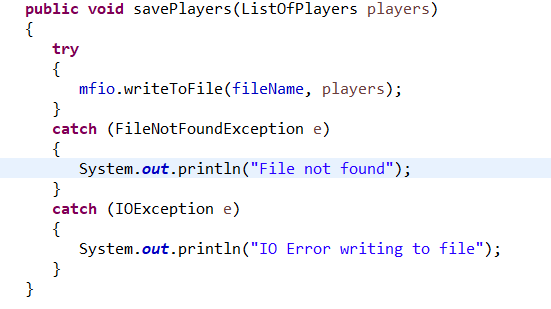
*PlayersSquadFileAdapter* and it’s met*hod changePlayer()*

thencreates object of class *ListOfPlayers* with all the existing players by method getAllPlayers(). Then an object of Class *Player* is created. After that, every player from object of *ListOfPlayers* is assigned to object of *Player,* one by one. And one by one, they are checked via *Player method* getName() for *playerName* to see, which player is supposed to be updated. In other words: which player’s attributes are supposed to be changed. And, after doing necessary variables type conversion, the values are being assigned to the player by calling appropriate class *Player* methods



As the final step of changePlayer() method, new values need to be stored by calling method savePlayers() with argument in form of object of class *ListOfPlayers*, which at this moment includes the updated player.

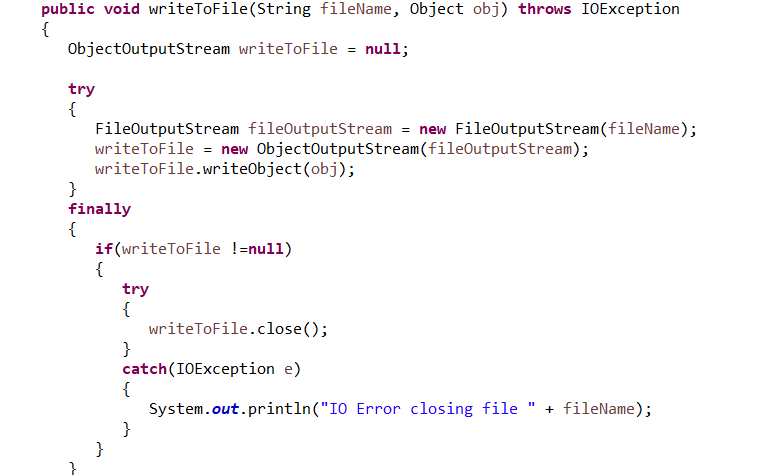
Method savePlayers() is yet another method of class *PlayersSquadFileAdapter*, which takes and object of class *MyFileIO*



and calls a method writeToFile() with arguments:

object of class *ListOfPlayers* and String that represents path of file where the new object of class *ListOfPlayers* is supposed to be saved

and then saves the object.

(NOTE:IF WE CHANGE THE MYFILEIO CLASS, THEN WE NEED TO CHANGE THE TEXT AND PICTURES HERE)

This above example shows typical collaboration of classes, that needed to be tested to secure they work properly.

The purpose of the implementation section is to explain interesting code snippets. An idea is to explain the complete path through your system from UI to database etc.

Remember that your implementation must be consistent with your design (Larman 2004, chap.20).

Which standard libraries are used? How are design patterns implemented, etc.

Hint: Implement your code in a testable manner.

# Test

Testing has been continuously performed thruout the whole developing process and began once first classes were developed.

## Test Specification

Testing has been divided into two categories:

-Functional testing-for testing functional requirements (covering Sport Management System application only)

-Non-Functional testing- for testing non-functional requirements:

### Functional testing

Functional testing has been divided into two phases:

1)WhiteBox testing which included unit testing of parts of the code continuously thru the whole development and did not included any GUI. For this testing, special additional classes have been developed. This testing served as a first line of control and was neccesary in or ther to begin BlackBox testing phase.

2)BlackBox testing which included Requirements being tested via GUI. No specific Test Cases were created for this project. Therefor Requirements themself served as a substitute.

The results can be seen in below tab:

**FUNCTIONAL REQUIREMENTS:**

**NUMBER DESCRIPTION PRIORITY RESULT AND COMMENT**

R01 User must be able to access High **OK**

LIST OF PLAYERS and see

it’s content.

R02 User must be able to edit a High **OK**

PLAYER and his information

in the LIST OF PLAYERS.

R04 User must be able to access High **????**

SQUAD LIST and see it’s

Content.

R05 User must be able to create a High **????**

SQUAD of players for a match

where user specifies information:

1.match date and time,

2.name of the opponent,

3.type of the match

(league, cup, friendly),

4.players that will start on

The pitch,

5.players that will start on

the bench

R06 User must be able to edit a SQUAD High **????**

of players and it’s information in

SQUAD LIST.

R07 User must be able to create new High **????**

PLAYER in the LIST OF PLAYERS

and his information:

1.name,

2.number,

3.position,

4.if suspended,

5.and for how long,

6.if injured

R08 User must be able to delete a Medium **OK**

PLAYER in the LIST OF PLAYERS.

R09 While creating/editing SQUAD, High **????**

system shows hard validation

(application will not allow

user to save such a SQUAD)

when adding:

1.more than 4 players that

will start on the bench for

*type of match: league*.

2.more than 5 players that

will start on the bench for

*type of match: cup*.

3.how many players will start

on the bench for *type of match:*

*friendly*.

R12 While creating/editing SQUAD system Medium **????**

Shows hard validation (

application will not allow user to save

such a SQUAD) when adding:

1.player that will start either on the

bench or players that will start on the

pitch *if suspended* and at the same

time *type of match: cup/league*

2.*type of match: friendly* accepts

Player even *if suspended.*

R13 While creating/editing PLAYER, system Medium **OK**

shows hard validation (application

will not allow user to save such a PLAYER)

if:

a) number is not filled

b) there is already existing Player

with the same number

### Non-Functional testing

Non-functional testing included requirements, that stand seperated from the Sport Management System application functionalities and were typicaly tested by specific method.

**NUMBER DESCRIPTION PRIORITY RESULT AND COMMENT**

R03 Website design/content looks High **OK** (Website has

per design been checked/ testedby manager/customer)

R10 Graphic User Interface looks Medium **OK**(after discussion

per design with manager/customer,

design was left up to development team)

R11 There is only one user role for the Medium **OK**(No user role is

application used, application can be simply turned ON and then OFF via Menu Bar)

The purpose of the test section is to document the result of your testing; to verify if the content of the requirements section has been fulfilled. How is the system tested, which strategy has been used; e.g. White Box (Unit Test), Black Box, etc.

## Test Specifications

For functional requirements, test specifications must be listed. These test specifications can be described as soon as the functional requirements have been completed (Use Cases including descriptions).

IEEE can be used as a template for test specification (IEEE Computer Society 2008). VIA Library can give you access to this standard.

# Results and Discussion

The focus of development has been to provide manager(customer) with Sport Management System that is capable of providing required functionalities. That includes creating and manipulating Players and Squads and their lists.That has been achieved.

GUI part of the code has been kept simple. No real styling of GUI elements has been used. Yet the GUI is working and is clear in order to provide manager(customer) with easy access to the required functionalites

As an additional requirement, representative web page was delivered, which has been checked/tested and confirmed by manager(customer).

The purpose of the results and discussion section is to present the outcome and achieved results of the project.

# Conclusions

The objective was to create Sport Management System, that would be able to efficiently create, manipulate and store data about Players and Squads based on Customer’s requirements. That has been achieved.

The other objective, thogh not that important, was to deliver representative web site. That has been achieved as well.

The purpose of the conclusion section is to compile the results from each section in the report. What is the conclusion? Did the project fulfil the requirements? Etc.

You can only comment on report contents, no new topics or content can be introduced in this section.

# Project future

Even though the manager(customer) requirements have been implemented into the application, there is still a question regarding the class architecture. In terms of potential future updates or requirements, it is possible that the architecture would prove to be not optimal specially in a case, that brand new functionalities would need to be introduced. That is a potential risk though, not necessarily inevitable future.

Reflect on your project from a technical viewpoint and describe what you would change if you could.

Suggest how the project could be improved or made ready for production. Discuss scalability, suggest possible spin offs, what is needed, missing, etc.?

# Sources of information

Duckett, J., 2011. HTML&CSS design and build websites, Indianapolis: John Wiley & Sons, Inc.

Duckett, J., Ruppert, G., Moore, J., 2014. JAVASCRIPT & JQUERY Interactive Front-End Web Development, Indianapolis: John Wiley & Sons, Inc.

Gaddis, T., 2015. Starting Out with Java: Early Objects, 5th edition. Harlow: Pearson.

International Journal of Physical Education, Sports and Health, 2016. Role of information technology in enhancing sports performance. [online] Available at: < http://www.kheljournal.com/archives/2016/vol3issue5/PartE/3-5-19-453.pdf> [Accessed 1 March 2019].

LaGrone, B., 2013. HTML5 and CSS3 Responsive Web Design Cookbook. Birmingham:Packt Publishing Ltd.

ThemeBoy, 2016. 10 Features All Sports Websites Need to Make an Impact. [online] Available at: https://www.themeboy.com/blog/10-features-sports-websites-need-make-impact/ [Accessed 1 March 2019].

# Appendices

The purpose of your appendices is to provide extra information to the expert reader. List the appendices in order of mention.

Examples of appendices

* Project Description
* User Guide
* Source code – source documentation
* Diagrams
* Data sheets
* Etc.

**Appendix A Project Description**

Insert the original Project Description here