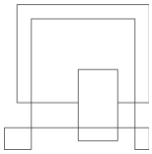


First Semester Project

Software Technology Engineering

Group 1

Adrian-Cristian Militaru 308842

Adrian Pompierescu 309774

Gabriel Moutinho Tristan 304376

Freja Hansen 308840

Supervisors:

Allan Henriksen

Mona Andersen

Software Technology Engineering

First semester

4th of June 2021

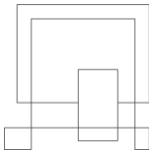


Table of content

Abstract 4

1. Introduction 1

2. Analysis 2

2.1 Functional Requirements 2

2.2 Non-Functional Requirements 3

2.3 Use Case Diagram 4

2.4 Use Case Description to manage player details 4

2.5 Activity Diagram for export the match list to the website 7

2.6 Domain Model 8

3. Design 9

3.1 Class Diagram 9

3.2 Sequence Diagram 11

3.3 VIAClub GUI 12

3.4 Website 16

4. Implementation 18

4.1 GUI 18

4.1.1 Create a player 18

4.1.2 Export to XML 19

4.2 Website 20

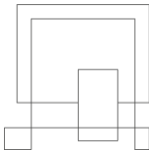
5. Test 22

6. Results and Discussion 24

7. Conclusions 29

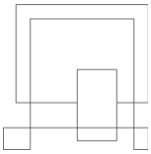
8. Sources of information 29

9. List of Appendices 30



List of figures and tables

- Figure 1. Use case diagram
- Figure 2. Use case description for Manage player details
- Figure 4. Activity diagram for Export match list to the website
- Figure 5. Domain model
- Figure 6. Relationship in Class Diagram
- Figure 7. Utils class diagram
- Figure 8. Sequence diagram
- Figure 9. GUI home screen
- Figure 10. GUI edit player screen
- Figure 11. GUI edit match screen
- Figure 12. GUI Export XML
- Figure 13. Constructor for the player class
- Figure 14. Create player method in PlayerGUIController
- Figure 15. Export Match list to an XML file
- Figure 16. Import Match list from an XML file to the website
- Figure 17. HTML code where all the matches will be populated

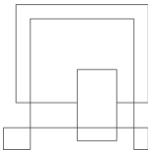


Abstract

VIA Club needs a management system that can handle the data for its football team and a website that can display their match details. The main purpose of the project is to deliver a system that can handle the data of the VIA Club and to document the different stages of development.

The development was initialized with the phase of analysis. To plan the technical performance of the software, requirements, use case diagrams, use case descriptions and activity diagrams were created. The findings of those details help us to initialize the design phase. A class diagram has been created and helps us to initialize the project in Java. The GUI was created by using the JavaFX library and Scene Builder. The implementation was followed by a test phase before being hand-in to the customer.

The result is a program that will allow the manager to store in binary files all the data. The functionality of the system enables the manager to add, edit and remove data, even export in an XML file functionality that can be used on the website. The project aims to contain all the VIA Club's requirements.



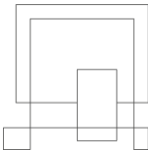
1. Introduction

Bob Oldenuff is the manager of the VellCity Indoor Athletics club, also known as VIAClub, and needs a program to make his work easier and more manageable. The manager has worked with managing the club for the last 31 years by himself. VIAClub is a relatively small club and unlike what the name suggests only has had an outdoor football team for the last 57 years. The club was founded almost 100 years ago by the legendary Robert Sixpack, Roberts's great-grandson is currently one of the club's biggest sponsors.

The manager's main job is to make lists of which players to use in every match, who will start on the pitch, and who will start on the bench, including the name, number, and position of each player. It is difficult to keep track of suspended players as well as injured ones and sometimes, new players are bought, and others are sold. The manager needs to be aware of all these changes.

The manager had a problem a while ago when his hot-headed striker ate his squad formation that was on a piece of paper. Keeping track of all the things on a piece of paper has turned out to be a burden nowadays so he decided to try the digital version of the game plan. Also, the manager would like to improve the online image of the club.

Because of the previously experienced accidents and happenings, VIAClub needs a digital system that makes the manager's job easier and a website to attract new supporters. For more details on the background description, look at Appendix A.



2. Analysis

The customer wants an application that can easily manage the team and prepare it for future matches.

The extracted keywords from the interview have been used to create functional requirements which represent what the system should be able to do and works with the customer's request.

2.1 Functional Requirements

Critical Priority

1. As a manager, I need a system to make matches and upcoming matches.

2. As a manager, I want to be able to write the date of a match and, and who the opponent will be.

3. As a manager, I need a system to store a list of players that I want to include in the squad for the next match.

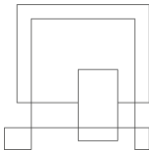
4. As a manager, I need to be able to make League and Cup matches that suspended or injured players cannot play.

5. As a manager, I need to be able to make Friendly matches where injured players cannot play.

6. As a manager, I need to be able to update and keep track of the player's details.

7. As a manager, I need to keep track of the injured or suspended players.

8. As a manager, I need to be able to add and remove players when they are sold or bought.



9. As a manager, I need to have an overview of the player to have a first name, last name, number, preferred position.

10. As a manager, I need to have an overview of players that are injured and try to also keep track of when and how long they might be suspended (Status).

11. As a manager, I need to be able to edit a player's position after it gets re-trained.

12. As a manager, I need to be able to edit matches if plans change.

13. As a manager, I need to be able to keep track of previous matches to be able to go back and see who has participated in which matches.

14. As a manager, I need to keep track of the number of substitute players for a specific type of match.

15. When an issue occurs, the problem pop-up a message.

High Priority

16. As a manager, I need to send the match details on the website.

17. As a manager, I need a responsive website with more details regarding the matches and the club.

18. As a supporter, I need to keep tracking the match history and the match details on the website.

2.2 Non-Functional Requirements

19. Every update in the system includes writing to a file.

20. The program is in the java language.

21. To have an interface that can be accessed with a mouse and a keyboard.

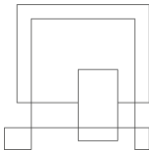
2.3 Use Case Diagram

The use case diagram has been made based on the customer requirements. This diagram represents the actions in relation to the system.

Figure 1. Use case diagram

2.4 Use Case Description to manage player details

Based on the use case diagram, a use case description has been made for each action - Manage player details, manage match details, assign player to a match, export match list to the website, and view website.

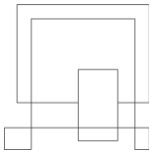


The use case for Manage player details consists of creating, reading player data, editing a player, and removing a player from the system.

Based on the requirements, “manage player details” is an essential use case because it will serve for another use case, such as Assign player to a match.

For adding a new player, the process consists of adding personal information and saving it. For everything else, the data will be displayed when the operation of reading and editing will be called.

The use case of Manage player details is presented in a step-by-step description in the following use case. More details regarding the other use case descriptions will be in Appendix B.



Use case	**Manage player details**
Summary	The manager will have access to the player list where he can edit, create, or delete a player from the system.
Actor	Manager
Precondition	Only applies to GET (step 5-8), EDIT (step 9-14), and REMOVE (step 15-20) and is required to have a player stored in the system.
Postcondition	The player will be created, updated, or removed from the system and the list will be updated after the action was made.
Base sequence	CREATE: 1.Create a new player 2.On the player list A. Insert the First Name B. Insert the Last Name C. Insert player number D. Insert position 3.Save the player info 4.If something missing on the player create, the program will pop up a warning message • The player status will be by default "available". GET: 5. System will show a list of all the players stored in the system. 6. Optionally use player status as a filter. 7. Select the player from the list. 8. Show player details: A. first name B. last name C. player number D. preferred position E. status (suspended, injured, or available) EDIT: 9. Select player from the list 10. Optionally to enter a player name, number, or status as a filter. 11. Select the player from the list 12. Edit player details 13. Got a warning message if any of the player details are not filled. 14. Save player details REMOVE: 15. From the player list 16. Optionally to enter a player name, number, or status as a filter 17. Select a player from the list 18. Delete the specific player 19. When confirming the deletion, a warning message will pop up to ask if those changes can be saved. 20. After the process begins, the player will be removed from the system.
Exception sequence	
Note	Any warning will appear on the screen when you want to create/edit a Player, in that way the system will verify that all the required fields are filled out.

Figure 2. Use case description for Manage player details

2.5 Activity Diagram for export the match list to the website

The activity diagram shows how the process of exporting a match-list to an XML file is handled. The process will read every detail from the matches list and all the details about every match will be exported in an XML file that can be used on the website.

After the process has been successfully finished a confirmation message will appear and will inform the file has been exported.

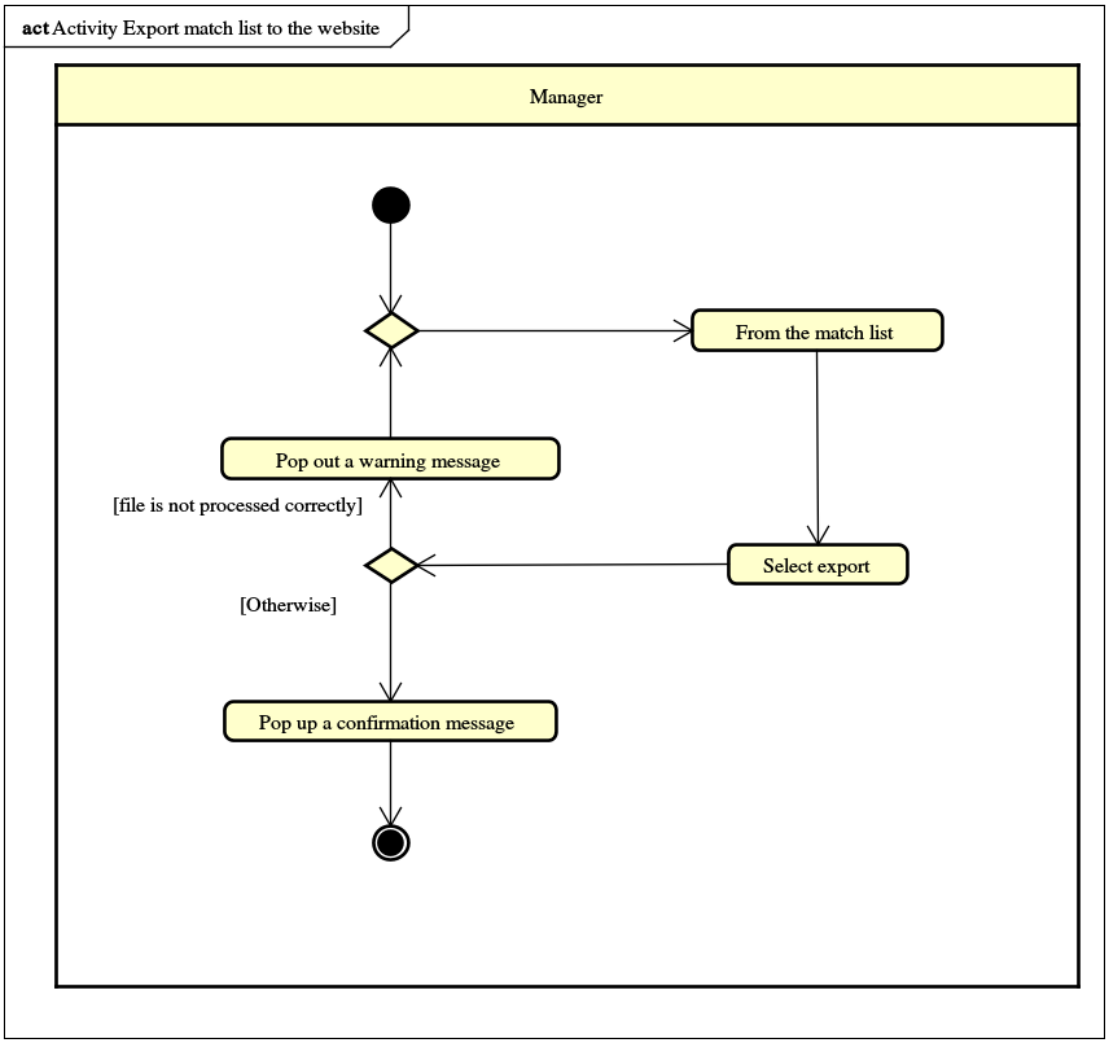
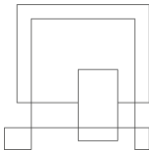


Figure 4. Activity diagram for Export match list to the website



2.6 Domain Model

A domain model has been created based on the functional requirements representing the relationship between classes. This model was generated based on the previous analysis steps.

The VIAClub application will contain a list of players and a list of matches, and for a match, it will be mandatory to have a date and location where the match will take place. The relation between Match and Player is referring to what player will be assigned to a specific match.

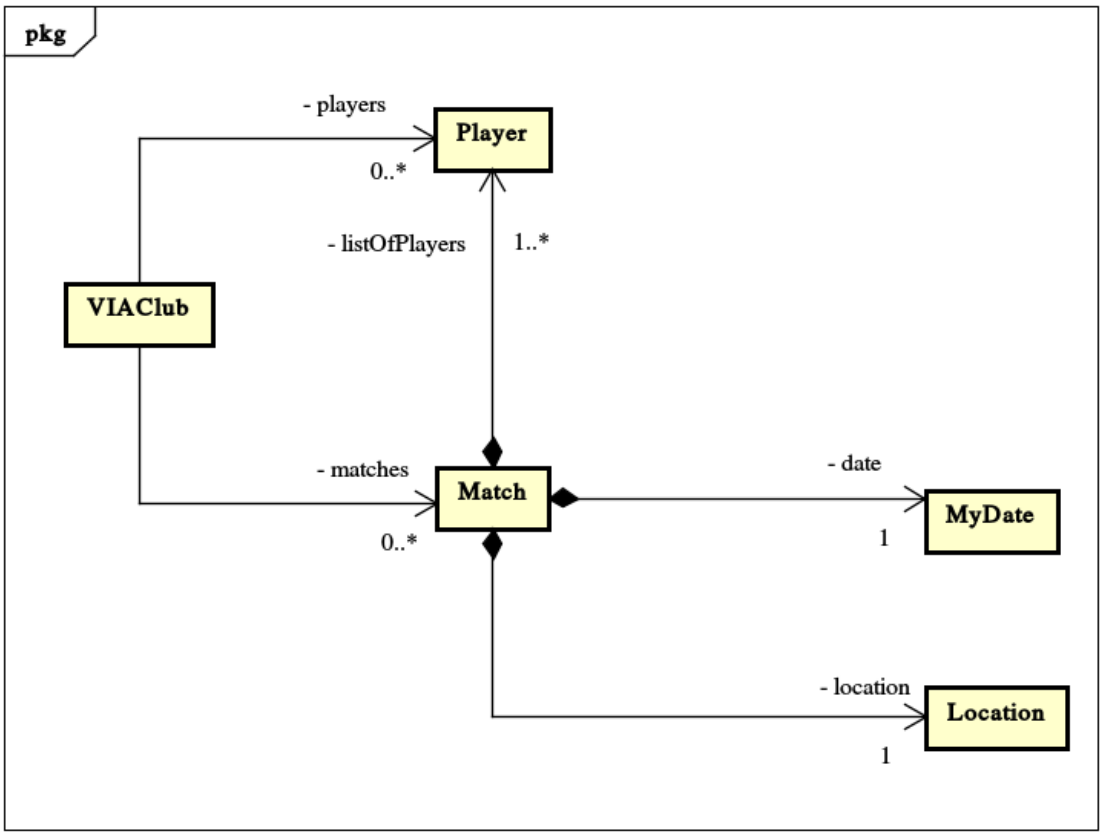
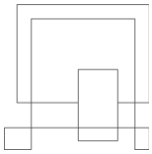


Figure 5. Domain model



3. Design

3.1 Class Diagram

The class diagram was created based upon the Domain Model and serves as the foundation for implementing the source code.

In ***Figure 6.*** We depict the Composition relation between classes. For example, the class “Match” is made up of one or more players, among other data from other classes.

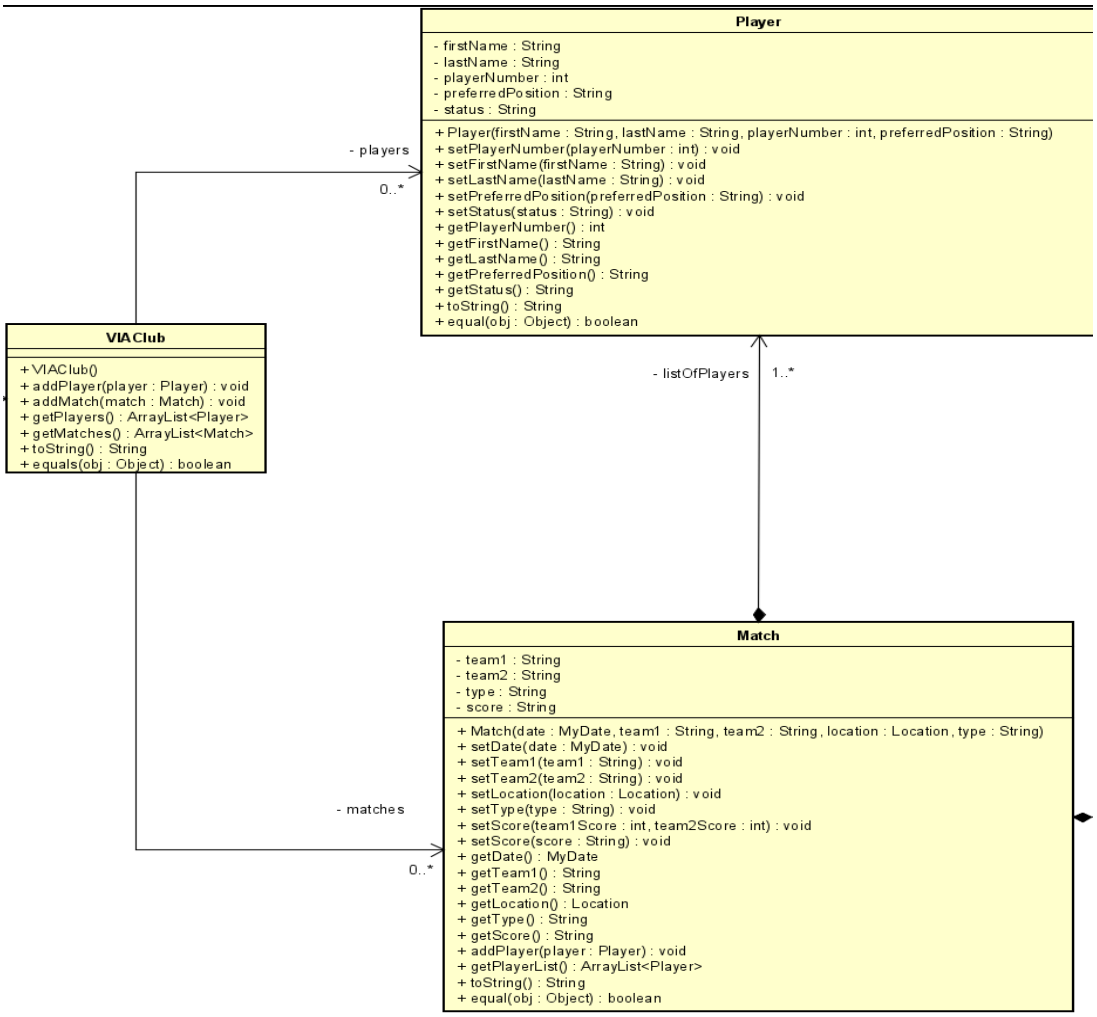
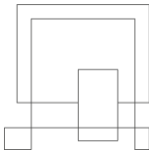


Figure 6. Relationship in Class Diagram



The “VIAClub” class has knowledge about “Match” and “Player” classes so we can see the Association relationship with multiplicity. To see the class diagram, look at Appendix E.

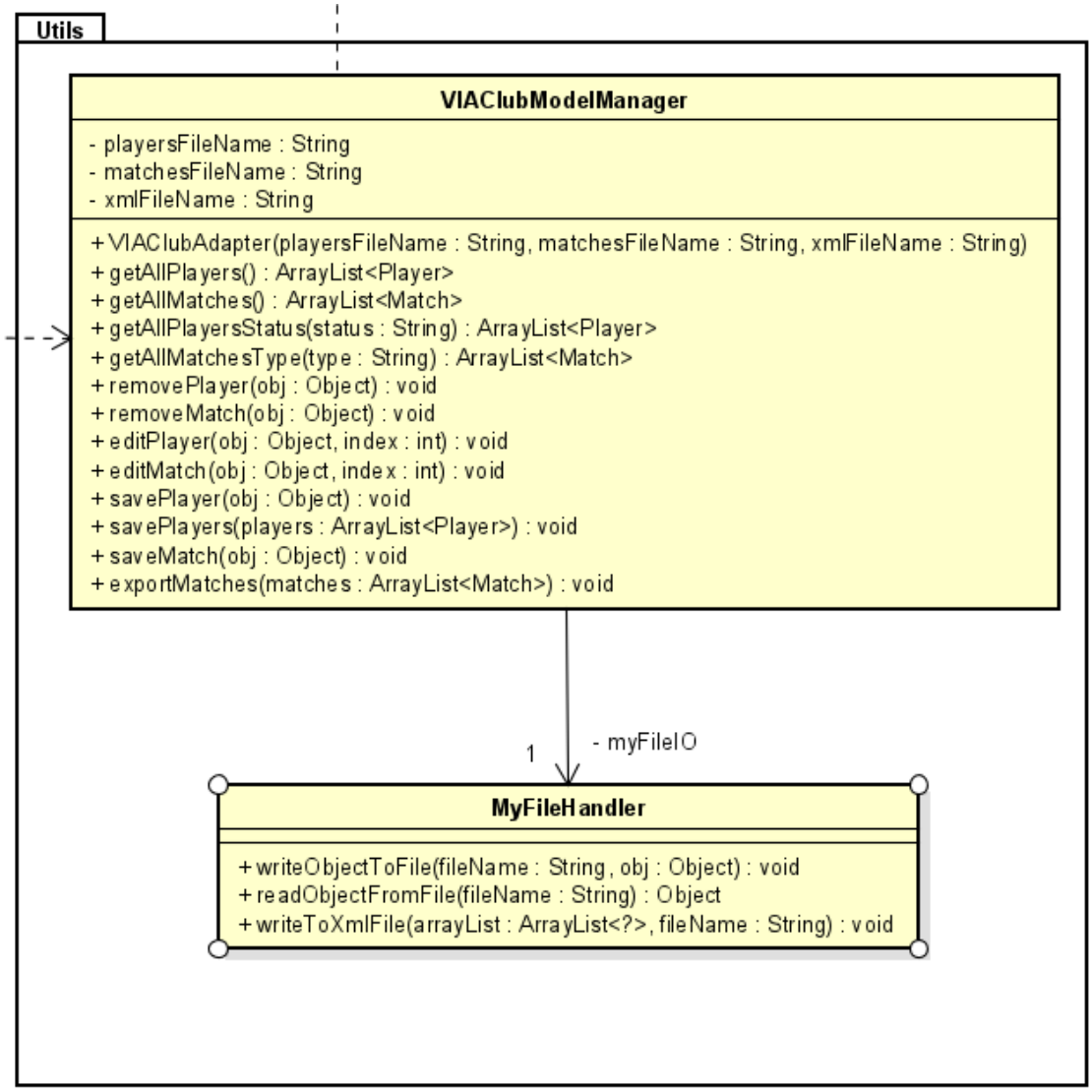
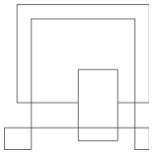


Figure 7. Utils class diagram

The class “MyFileHandler” is used for writing and reading data from a binary and to a binary file, like matches and players. Since we have learned that



binary files are more useful to store class objects rather than a text file. This class can also write a text file that will export in an XML format using an external library. (Appendix C) In order to import all the match details for the website.

3.2 Sequence Diagram

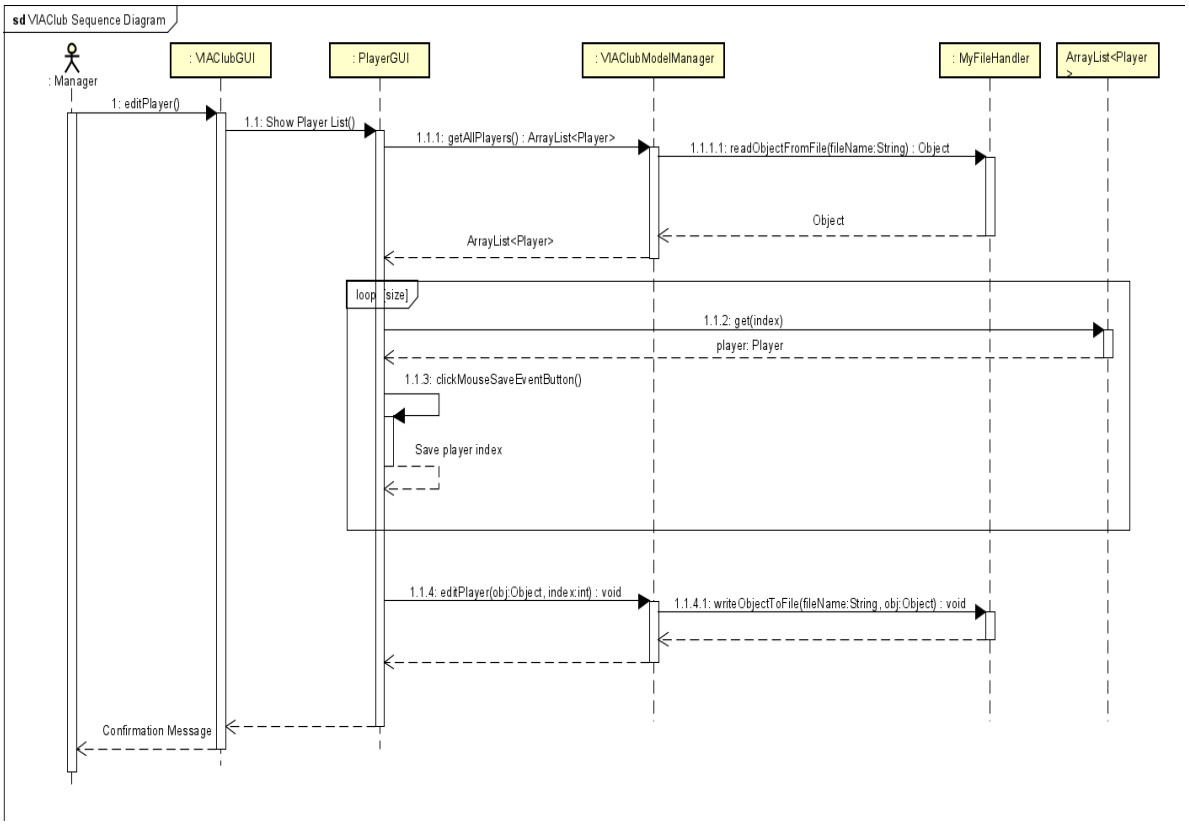
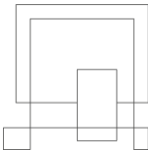


Figure 8. Sequence diagram

The edit player sequence diagram explains and describes the steps that the manager will do to edit a player and the functionality of it.

Firstly, we run the VIAClubGUI, to edit a player the tab Players is called in the PlayerGUIController which will trigger the method getAllPlayers() from the VIAClubModeManager, then the list of all the players at VIAClub will be shown(ArrayList<Player>) and a specific index will be stored on a global variable.



The method readObjectFromFIle() is called to MyFileHandler with the purpose of reading the existing players from the binary file.

The method gets (index) will return the object that is located on the specific index. After the object is received, the clickMouseSaveEventButton() function can save the new data about the object created. The new modifications of a player object will be saved to the binary file by the writeObjectToFile() function that is called from the “MyFileHandler” class.

Finally, a confirmation message will be displayed in the PlayerGUIController for the manager. To see the sequence diagram for editing a player, look at Appendix F.

To see the original files of the diagrams, look at Appendix G.

3.3 VIAClub GUI

The first page presented to the manager consists of three tabs, the Player tab, the Matches tab, and the Export XML tab.

Each tab comes with a lot of features like the player tab in which the manager can add a new player to the team, can edit the details of existing players, and remove also, but the most important thing is the search button that can search the players by their actual status.

The player tab displays the name and the status of each player.

Also from the home view, the manager can navigate through tabs.

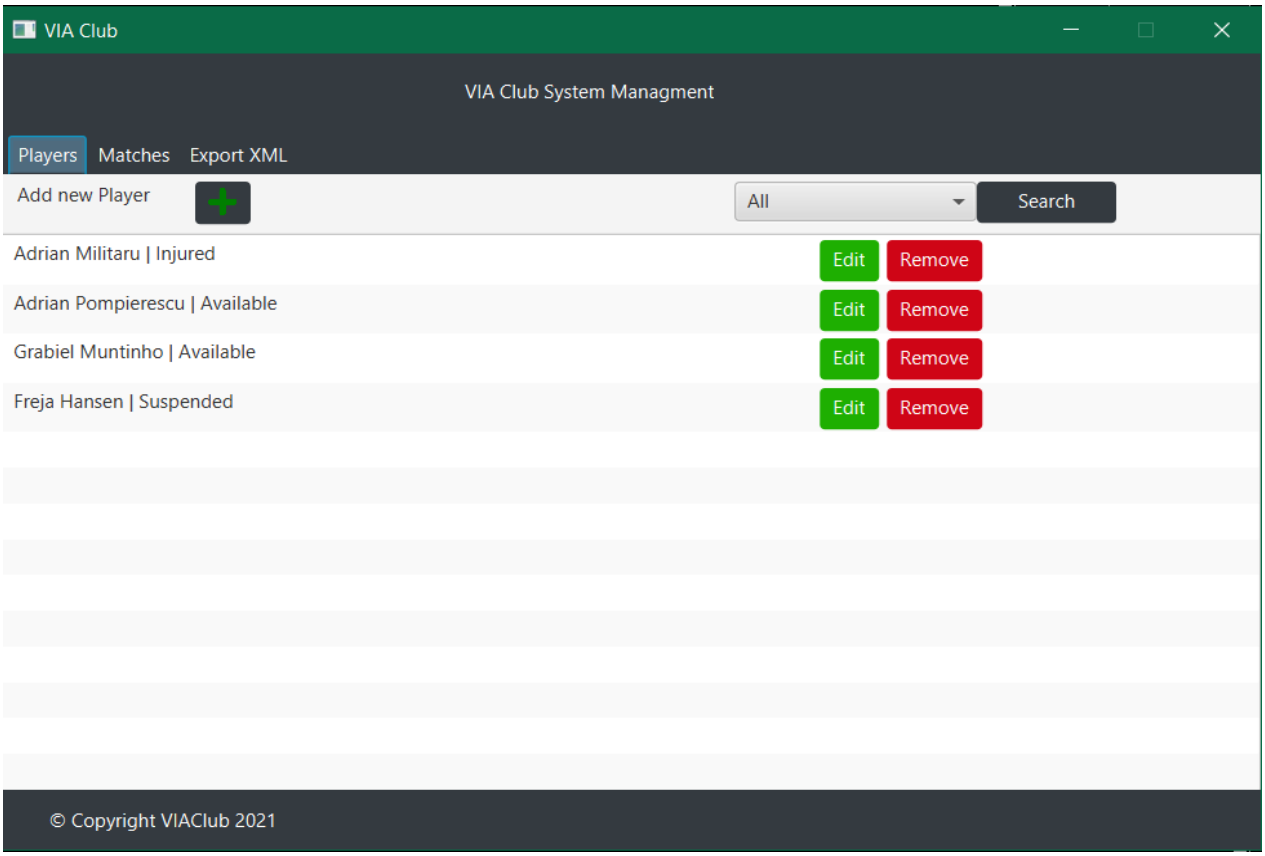


Figure 9.GUI home screen

The first page presented to the manager consists of three tabs, the Player tab, the Matches tab, and the Export XML tab.

Each tab comes with a lot of features like the player tab in which the manager can add a new player to the team, can edit the details of existing players, and remove also, but the most important thing is the search button that can search the players by their actual status.

The player tab displays the name and the status of each player. Also from the home view, the manager can navigate through tabs.

The edit button from the Player tab will display a pane where the manager can change all the information about the player including the first name, last name, number, preferred position, and status.

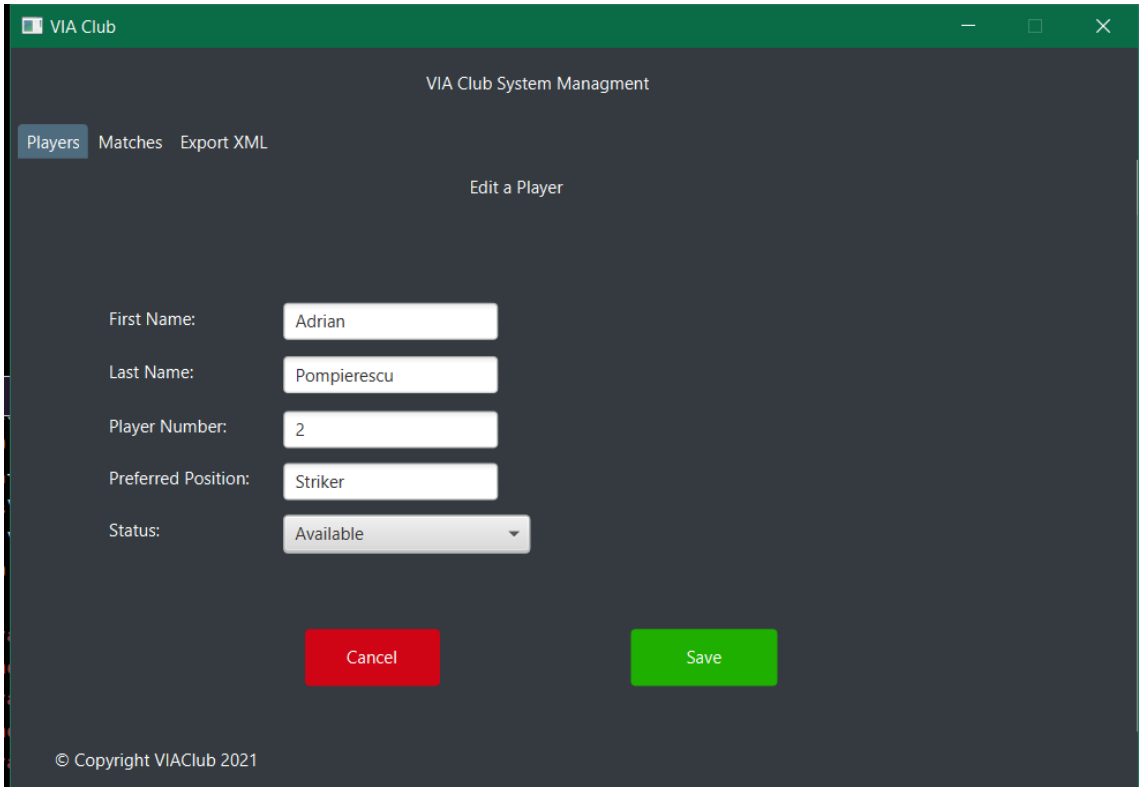


Figure 10. GUI edit player screen

The Matches tab is the same as the Players tab, in essence, of course, the manager can use this tab to add or remove a match or edit a match, and when the list will get crowded the search function will be useful.

The edit pane helps to build the team and keep track of the data about the opposing team, match type location, date, time, stadium, and score.

Also editing all the fields as the manager requires and save them in the system.

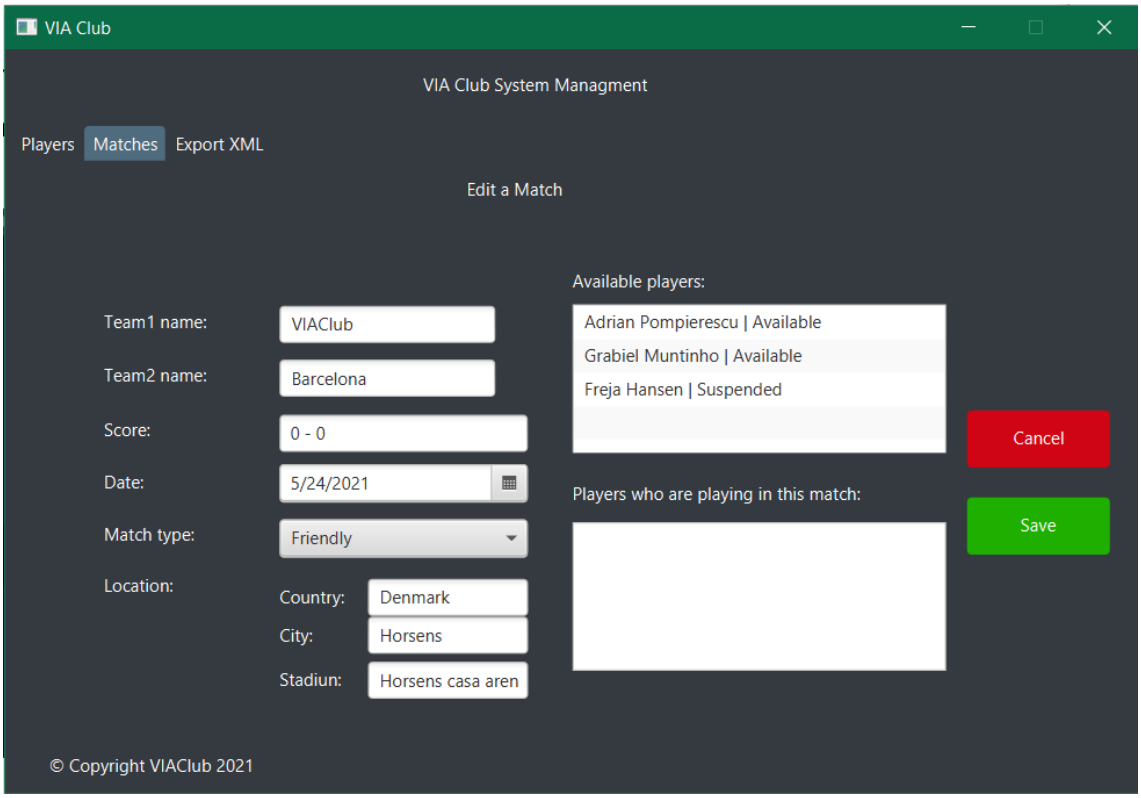


Figure 11.GUI edit match screen

To make it easier for the manager to export information about the team on the website, on the Export XML tab there is a button to Export MatchList as XML.

For more information about the graphical user interface, see Appendix H.

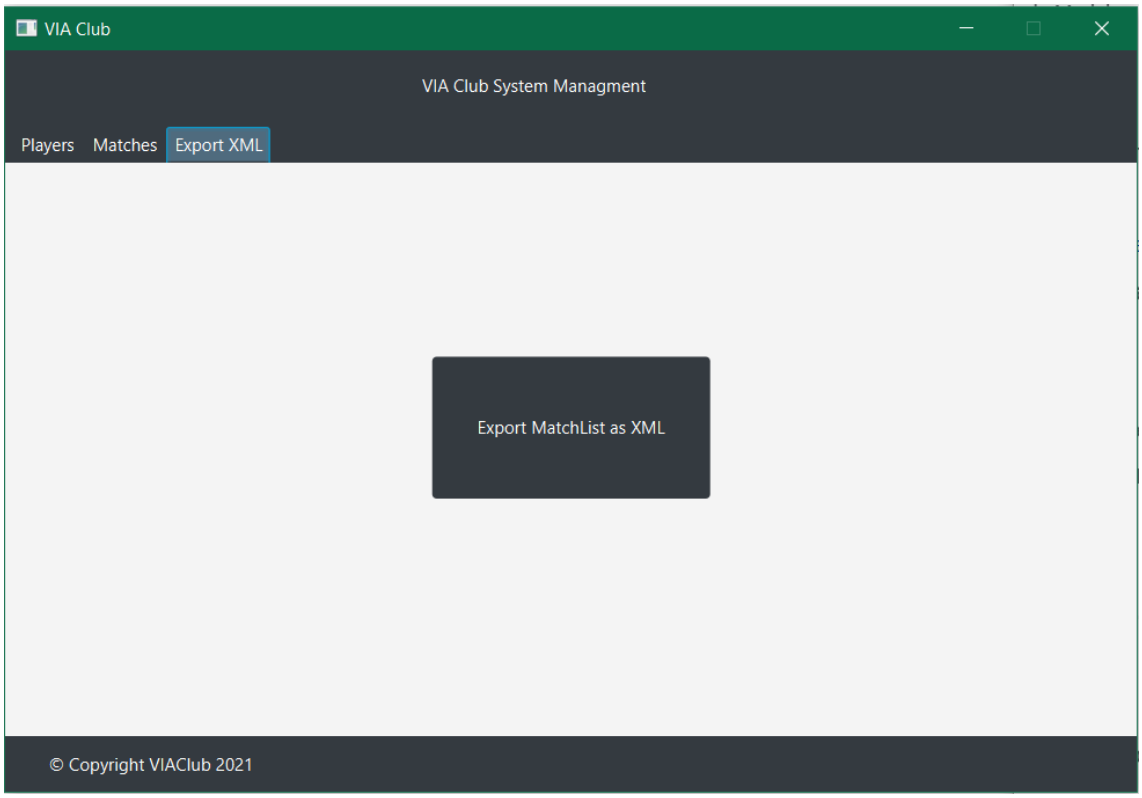
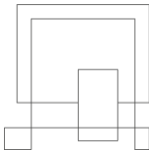


Figure 12. GUI Export XML

3.4 Website

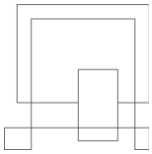
The website for VIAClub should be available for supporters and fans. The desktop version of the website that contains a navigation bar on the top of the window, the body of the page, and a footer. The navigation bar has been split into 4 big categories that the user can navigate through with easy access to the different topics or pages. The logo comes as an extension for the navigation bar, and they are fixed so the user can access a different topic every time. The footer matches the heading in colour and contains different information.



Every page has its design and functionality based on its content. The Upcoming Events category is more special because it was specially designed to support importing external data.

The website is responsive so it can be accessed from every device with different screen sizes. On devices that have a smaller resolution, the navigation bar be collapsed and hide the other pages and reveal it at the action of the user.

For more information about the website, see the website responsive images in Appendix I.



4. Implementation

Based on the design outputs, the implementation process starts. The implementation process began by implementing the basic classes from the class diagram.

4.1 GUI

4.1.1 Create a player

The player creation includes all the player details such a first name, last name, player number, preferred position, and player status.

```
public Player(String firstName, String lastName, int playerNumber, String preferredPosition, String status)
{
  this.firstName = firstName;
  this.lastName = lastName;
  this.playerNumber = playerNumber;
  this.preferredPosition = preferredPosition;
  this.status = status;
}
```

Figure 13. Constructor for the Player class

A method called “clickMouseCreateEventButton(MouseEvent event)” that are located in the “PlayerGUIController” class will perform the player creation reading all the data stored in the fields and create a new “Player” class stored in the “player.bin” file, after the action was successfully made, a confirmation message will appear to inform the manager that the player has been stored in the system.

```
@FXML void clickMouseCreateEventButton(MouseEvent event) {
    if(labelFirstName.getText().isEmpty()||labelSecondName.getText().isEmpty()||labelPlayerNumber.getText().isEmpty()||labelPreferredPosition.getText().isEmpty())
    {
        Alert alert = new Alert(Alert.AlertType.ERROR,  s: "All fields must be filled!");
        alert.showAndWait();
    }else {
        Player localPlayer = new Player(labelFirstName.getText(), labelSecondName.getText(), Integer.parseInt(labelPlayerNumber.getText()),
                labelPreferredPosition.getText(), comboPlayerStatus.getValue());
        manager.savePlayer(localPlayer);
        Alert alert = new Alert(Alert.AlertType.INFORMATION,  s: "The player have been created!");
        alert.showAndWait();
        dialogPop.setVisible(false);
        //update the player list with the new player
        setListDetails(manager.getAllPlayers());
        clickMouseCancelEvent(event);
    }
}
```

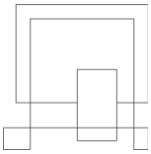
Figure 14. Create player method in PlayerGUIController

4.1.2 Export to XML

The method “exportAction(Mouse event)” that is located on the “ExportGUIController” class will allow the manager to call the function “writeToXmlFile” that is located in the “MyFileHandler” class and it will use an external library “xstream-1.4.17.jar”, this library is located on Appendix C, to export the ArrayList of matches in an XML format.

After the export was successfully done, a confirmation message will appear to inform that the match list has been successfully exported.

To see the source code for the software application and the javadocs generated, look at Appendix J and K.



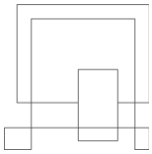
```java
public void writeToXmlFile(ArrayList<?> arrayList, String fileName)
{
  XStream xstream = new XStream(new DomDriver());
  String xml = xstream.toXML(arrayList);
  PrintWriter writeToFile = null;

  try
  {
    FileOutputStream fileOutStream = new FileOutputStream(fileName,  append: false);
    writeToFile = new PrintWriter(fileOutStream);
    writeToFile.println("<?xml version=\"1.0\" encoding=\"UTF-8\"?>");
    writeToFile.println(xml);
  }
  catch (Exception e)
  {
    System.out.println("File not found exception");
  }
  finally
  {
    if (writeToFile != null)
    {
      writeToFile.close();
    }
  }
}
```

Figure 15. Export Match list to an XML file

4.2 Website

The JavaScript code has been used to import the XML file on the website and represent it under a table format in HTML code.

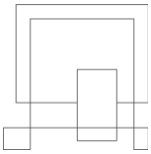
The function “showTables()” will read all the matches that are located in the XML file and base on the parameters will read a specific type of match and



will assign it to a specific table id. To see the source code for the website, look at Appendix L.

```
function showTables(client,TableID, MatchType){
    var xmlDoc = client.responseXML;
            var matches = xmlDoc.getElementsByTagName("Model.Match");

            var container = document.getElementById(TableID);

            var tableString = "<table class='table'>";
            tableString+="<tr class='head'>";
            tableString+="<th>Date</th>";
            tableString+="<th>Location</th>"
            tableString+="<th>Home vs Visiting teams</th>";
            tableString+="<th>Score</th>";
            tableString+="</tr>";
            for (i = 0; i < matches.length; i++) {
                if(matches[i].getElementsByTagName("type")[0].childNodes[0].nodeValue == MatchType){
                    tableString += "<tr><td>";
                    tableString += matches[i].getElementsByTagName("date")[0].getElementsByTagName("day")[0].childNodes[0].nodeValue+"/"+
                    matches[i].getElementsByTagName("date")[0].getElementsByTagName("month")[0].childNodes[0].nodeValue+"/"+
                    matches[i].getElementsByTagName("date")[0].getElementsByTagName("year")[0].childNodes[0].nodeValue;

                    tableString +="</td><td>";
                    tableString += matches[i].getElementsByTagName("location")[0].getElementsByTagName("country")[0].childNodes[0].nodeValue + ", "+
                    matches[i].getElementsByTagName("location")[0].getElementsByTagName("city")[0].childNodes[0].nodeValue+", "+
                    matches[i].getElementsByTagName("location")[0].getElementsByTagName("stadium")[0].childNodes[0].nodeValue;

                    tableString +="</td><td>";
                    tableString += matches[i].getElementsByTagName("team1")[0].childNodes[0].nodeValue + " vs "+matches[i].getElementsByTagName("team2")[0].childNodes[0].nodeValue;
                    tableString +="</td><td>";
                    tableString += matches[i].getElementsByTagName("score")[0].childNodes[0].nodeValue;
                    tableString += "</td></tr>";
                }
            }
            tableString += "</table>";

            container.innerHTML = tableString;
}
```

Figure 16. Import Match list from an XML file to the website



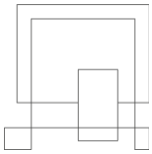
```
<div class="container-fluid">

    <div class="row my-1">
        <p><strong>Next events on the FIFA league</strong></p>
    </div>
    <div class="row my-1">
        <div class="col-lg-12 col-xl-3 text-center my-1"> <img src="../images/upcoming/fifa.png" class="img-fluid"
                alt="fifa logo"> </div>
        <div class="col-lg-12 col-xl-7 text-center my-1">
            <div class="table-responsive" id="cup-match">
                <!--Populate with XML import-->
            </div>
        </div>
    </div>

</div>

<div class="container-fluid">
    <div class="row my-1">
        <p><strong>Next events on the UEFA league</strong></p>
    </div>
    <div class="row my-1">
        <div class="col-lg-12 col-xl-3 text-center my-1"> <img src="../images/upcoming/uefa.png" class="img-fluid"
                alt="uefa logo"> </div>
        <div class="col-lg-12 col-xl-7 text-center my-1">
            <div class="table-responsive" id="league-match">
                <!--Populate with XML import-->
            </div>
        </div>
    </div>
</div>

<div class="container-fluid">
    <div class="row my-1">
        <p><strong>Next events on the Friendly League</strong></p>
    </div>
    <div class="row my-1">
        <div class="col-lg-12 col-xl-3 text-center my-1"> <img src="../images/upcoming/friendly-match.png"
                class="img-fluid" alt="friendly match logo"> </div>
        <div class="col-lg-12 col-xl-7 text-center my-1">
            <div class="table-responsive" id="friendly-match">
                <!--Populate with XML import-->
            </div>
        </div>
    </div>
</div>
```

Figure 17. HTML code where all the matches will be populated

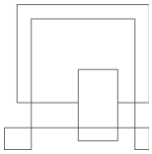
5. Test

After the implementation of the website and GUI, the system is tested, and it is ensured that all the use cases are fulfilled.

Use Case	Test Result	Comments
Manage Players	Works	The manager can read all the data of a player, create a new player, edit an existing player, or remove a player.
Manage Matches	Works	The manager can read



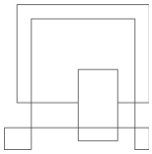
		all the data of a match, create a new match, edit an existing match, or remove a match.
Assign player to a match	Works	The manager can assign a player to a specific match, based on the match type and the player status
Export match list to the website	Works	The manager can export all the matches that are stored in the system in an XML file that can be imported into the VIAClub website.
View website	Works	The supporter can visit the VIAClub website and see what the next matches under the upcoming events tab.



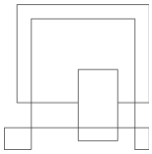
6. Results and Discussion

The following table has been made to represent all the functional requirements used in the analysis phase.

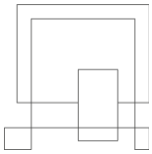
Priority	Requirement id	Requirement	Status
Critical	1.	As a manager, I need a system to make matches and upcoming matches.	Working
Critical	2.	As a manager, I want to be able to write the date of a match and, and who the opponent will be.	Working
Critical	3.	As a manager, I need a system to store a list of players that I want to include in the squad for the next match.	Working
Critical	4.	As a manager, I need to be able to make League and Cup matches that suspended or injured players cannot play.	Working



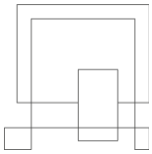
Critical	5.	As a manager, I need to be able to make Friendly matches where injured players cannot play.	Working
Critical	6.	As a manager, I need to be able to update and keep track of the player's details.	Working
Critical	7.	As a manager, I need to keep tracking of the injured or suspended players.	Working
Critical	8.	As a manager, I need to be able to add and remove players when they are sold or bought.	Working
Critical	9.	As a manager, I need to have an overview of the player to have a first name, last name, number, preferred position.	Working



Critical	10.	As a manager, I need to have an overview of players that are injured and try to also keep track of when and how long they might be suspended (Status).	Working
Critical	11.	As a manager, I need to be able to edit a player's position after it gets re-trained.	Working
Critical	12.	As a manager, I need to be able to edit matches if plans change.	Working
Critical	13.	As a manager, I need to be able to keep track of previous matches to be	Working
Critical	14.	As a manager, I need to keep track of the number of substitute players for a specific type of match.	Working



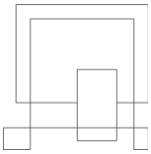
Critical	15.	When an issue occurs, the problem pop-up a message.	Working
High	16.	As a manager, I need to send the match details on the website.	Working
High	17.	As a manager, I need a responsive website with more details regarding the matches and the club.	Working
High	18.	As a supporter, I need to keep tracking the match history and the match details on the website.	Working
Low	19.	Every update in the system includes writing to a file.	Working
Low	20.	The program is in the java language.	Working



Low	21.	To have an interface that can be accessed with a mouse and a keyboard.	Working

The implementation process started by implementing all the methods and fields from the class diagram and fulfil all the requirements from the activity diagrams base on every use case.

After the implementation process has been successfully made, the team realizes that some methods of the “VIAClub” class are not used, and these modifications have been reflected on the class diagram.



7. Conclusions

The manager contacted the group with the issue he had while trying to create matches. He either lost the paper with his work or was unable to remember all the details, he looked for a solution to the problem.

To summarize, the goal of the project was to create a software application that can manage a football team, which will help VIAClub with storing data for players and matches and to export it to a newly designed website that will work on every device in order to make the manager's job easier.

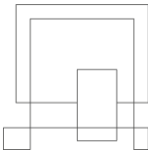
The Analysis started with getting all the requirements that the customer provided during his interview. While going through the design and implementation of the software, every aspect of the project was formed based on the Analysis phase in order to fulfil all the customer's requirements.

From the project description the requirement list was made, which made it possible for the Class Diagram was created from which it was possible to create the program. The website was created and functioning.

To make sure the client's requirements were fulfilled a test was done and the conclusion is that the project is fulfilling the requirements and solves the client's issue.

8. Sources of information

Duckett, J., 2011. *HTML and CSS : Design and Build Websites.* First Edition ed. Indianopolis, IN46256: John Wiley & Sons, Inc..



Duckett, J., s.f. *Javascript & jQuery, Jon Duckett.* First Edition ed. s.l.:s.n.

Gaddis, T., s.f. *Starting Out with Java: Early Objects, Global Edition.* 5th Edition ed. s.l.:s.n.

LaGrone, s.f. *HTML5 and CSS3 Responsive Web Design Cookbook.* s.l.:s.n.

9. List of Appendices

Appendix A: Project Description

Appendix B: Use Case Description

Appendix C: External library used for application

Appendix D: Activity Diagrams

Appendix E: Class Diagram

Appendix F: Sequence Diagram

Appendix G: Astah File

Appendix H: User Guide

Appendix I: Website responsive images

Appendix J: Software Application Source Code

Appendix K: JavaDocs

Appendix L: Website Application Source Code