Project Report – Edison RESHKETA

-Project Co-Leader

- As a project co-leader, most importantly I came up with the idea of the game and the way how the game should work. I worked on organizing the group into sub-teams. Moreover, I made sure that each sub-team had clear goals and targets. I scheduled meetings in order to keep good communication between the sub-teams as well as I frequently checked on the each of the sub-teams to make sure that the project was going on the right direction. I also scheduled brainstorming sessions when a big decision had to be taken for the development of the game and its continuation. Moreover, making sure that everybody had an objective to work on, was one of my duties.

-Structure Team Leader

- As the structure team leader, I decided to divide the team in pairs (Me and Assia, Sannya and Mathilde). As our project was extremely object oriented, having pairs working on different objects was a perfect solution. I worked and helped on the establishment of the general structure of the project. I made sure that the structure was efficient and served all our goals and necessities. Moreover, I was in close touch with the Algorithm team(GameEngine Team) in exchanging our plans, as well as GUI team to make sure that the structure team was providing all the needed tools. On the same time, I made sure that the structure was being well established during the project.

-Other contributions

- I worked with Assia on the Player Class. We created the player class with its attributes and constructors.
- I helped on creating the gitignore file.
- I coded all the User class.
- I made the connection between the User and the League.
- I made the connection between each window by creating constructors which take the League and the User by reference.
- I designed and created the training functions (4 different training methods), and also connected them to their respective buttons on the training window. I added different features to the training window, such as asking the user if he is sure to take the training.
- I helped in the calendar algorithm with Sannya and Mathilde.
- I designed and created the Market.ui and coded the sell and buy functions and all the interections when buttons are clicked. I made it possible for the user to sell and buy players. I added to this window several features and constraints, such as asking the user if he is sure to buy or to sell the player. Moreover, the user can't be left with less than 7 players and can't have more than 12 players in his team.
- I updated the League class by adding a market attribute which keeps the players which are available to be bought.
- Moreover, I made sure to show the pictures selected by Assia, Mathilde and Joan in the mainwindow, training and market window. I also helped on selecting some of them.
- I did some visual modifications on mainwindow.ui.
- I added some attributes to the Team class such as points, numberofplayers and gamesplayed. I modified the updateoverall() functions of the Team and Player classes.
- I coded the function which ranks the teams in the league based on their number of points.
- I also showed in the league ranking the number of games done by each team.
- Lastly, I coded the functions to permit the user to save the game and to resume it after. I managed to organize all the information of the game in progress and write it on files when the button Save is clicked. These files are read when the user requests to resume the game and when read they bring

the game to the previous saved state. Moreover, I took care of the case when the user saves its team and has less than 12 players and other conditions which might cause a bug.
- I also helped on progression after the user finished its game and the way how the results of the matches between the teams update the attribute of the teams and the ranking of the League.
- I also helped other members of the team in debugging several functions.

Comment:
Through this project we practiced a lot of technical skills as well as soft skills. It was one of those projects where communication and team work are essential alongside with technical abilities. We practiced C++, got to know GitHub and also Qt. It was indeed a valuable experience which helped us progress a lot in different aspects.

Project Report – Sannya AMOIKON

In this project, I worked in the Structure Team and I was responsible for the Virtual Machine. Towards the end, I also joined the GUI team to help on the design.

As part of the Structure team:

- Helped in deciding the division of the work in the team with the Structure Team
- Implemented the Team and League classes with Mathilde
- Worked on the calendar algorithm with Mathilde and Edison
- Tested the League class
- Debugged the Calendar algorithm
- Generated the calendar function with the algorithm, calendar() (changed the original algorithm to another because of a gigantic bug)
- Generated the list of games of the user, getAllUserMatches()
- Added the getThisWeekOpponent()

As responsible of the virtual machine:

- I made sure the project ran on the virtual machine
- Corrected the bugs raised by the version difference of QtCreator with Skander
- Found out how to compile and build the project on it

As a quick addition to the GUI:

- Connected the calendar function to the interface
- Modified several backgrounds
- Provided the main menu background (Top5)
- Made some aesthetic changes (layouts and so on in preparation.ui for example)
- Helped with the different bugs as well

Project Report – Assia BENACHIR

During team project work, I was part of the game structure team. Hence, I worked alongside Mathilde, Sannya, and Edison, to design the “player”, “market”, and “training” classes, to figure out their interdependences, and complete all of their functions and attributes. In particular, Edison and I were in charge of giving specific attributes to the players (such as their names, and physical attributes, etc.), all the while making sure that our implementation was consistent with the structure of the team class. We came up with functions that modify these players’ attributes during training, and implemented a function that gives a monetary market value to each player.

I also worked on defining some basketball tactics in pseudo code in order to adapt them to our program’s framework, but we did not end up using this implementation of the tactics.

Afterwards, I worked on finding basketball players names, team names, and downloading pictures of the players, alongside Mathilde and Joan, and this task proved to be quite time consuming.

Through this group work, I learned many valuable new skills. The most important of all is that of learning to work as a group, to stay organized and distribute tasks to each. On a more technical level, I learned a lot about coding in C++, and got the chance to be acquainted with Git and QT for the first time, which proved to be very useful tools, especially while working in group.

Project Report – Mathilde JACQ

I was part of the Basketball Team Manager project, that led to the creation of the game "Top5". During this project, I was part of the Structure Team, i.e. the team who designed the various classes of objects on which our game is based.

With the Structure Team (Edison, Sannya, Assia and myself), we first had a reflexion on how we should structure our game. We decided to create several classes (Player Class, Team Class, League Class, Market Class, Training Class), each class depending on the others. Then, Sannya and I created the class Team and the class League; thus, we decided of which attributes to give to the objects we created.

The class League was especially hard to implement, as it had to include a calendar algorithm, which has as output the calendar of all the matches of a basketball season.
I had to code some useful functions for the calendar algorithm, such as the intersection function. Our first version of the calendar algorithm included the intersection function but we had to change the algorithm, as the first version did not work correctly. I also created a function "printing calendar", which was really long to implement but unfortunately it was finally not used in our project.

Moreover, I wrote several test functions to make sure the player class, the team class and the calendar algorithm were working properly. I spent some time debugging the problems which arouse when we linked all our classes.

After have implemented all our needed classes, we uploaded our work on QT.
Then, Assia and I worked on the creation of a database of 184 basketball players for our game. After we created this database of names and pictures, I uploaded these pictures on QT and worked on how to make them display correctly in the game.
Finally, I wrote a description of how to play our game Top5.

Project Report - Skander MOALLA

- As a team co-leader:
  - Divided the team into subteams: We made 3 subteams: The *GUI team*, the *GameEngine team* and the *ObjectDesign Team*. Given each one's preferences and skills we assigned each team member to a subteam.
  - Did routine check ups to make sure everybody has something to work on.
- As the `GameEngine` subteam leader:
  - Code and Design:
    - Designed the `GameEngine` class: I designed and coded the signature of the methods and attributes of the `GameEngine`.
    - Designed the interaction between the `public` and `private` methods of the engine.
    - Designed and implemented the interaction of the GameEngine with the different Objects and GUI elements `User`, `Team`, `Player`, `League`, and the `NextGame` window.
    - Coded `GameEngine::applytactic(Team& playingTeam, Team& initTeam)`.
    - Coded `GameEngine::getBacktoDefaultTactic(Team& playingTeam, Team& initTeam)`.
  - Management:
    - Designed the independent methods of the GameEngine to be able to share their coding among the team.
    - Supervised the coding of the shared methods.
    - Managed the communication with the GUI and the ObjectDesign team to make everything interact well together.
- As the Git Leader:
  - Created the GitHub repo.
  - Helped all team members familiarize with git mechanics and theory:
    - Git theory: local repo, remote repo, and push-pull requests.
    - Familiarize with GitHub desktop.
    - How to commit and solve conflicts easily on visual studio code and GitHub Desktop.
    - gitignore: add their build and user preferences in the gitignore.
  - Maintained a healthy sustainable and a runnable HEAD so everyone can test their new features.
    - Implemented different branches for the subteams to try new features.
    - Reverted commits when needed, resetted the HEAD sometimes and debugged everytime.
- Other contributions:
  - Fixed bugs in `Team` and `League` objects.
  - Designed the mechanics of a Game: implemented all `GameEngine` methods to ensure a smooth transition during the game.
  - Tried to do the make file of the project with CMake, but we eventually built the project with Qt thanks to the GUI team.

Project Report - Kevin SHEN

At the initial stage of this project, I contributed many useful and actually adopted ideas when we sketch on paper to decide which basic classes we need and the hierarchies between them, for example, league > team > player, and which key attributes and functions each class needs.

Later, as I am in the algorithm team, I spent most of my time building the class GameEngine. Before we decided to have a GameEngine class in our project, I already wrote the function updateOverall, which recomputes the overall rating of two teams after a game based upon their initial overall ratings and their game result, and the function SimulatedGame, which plays an interactive game between a player and the computer. Simultaneously, I brainstormed with Joan and Skander together on codes for the non-interactive game.

After we added the GameEngine class into our project and merge of our work with the classes team and GUI team, I modified the updateOverall function and made it a function under GameEngine. I didn't reutilize the function SimulatedGame much. I took some ideas from it when I wrote the getAttackResult function of GameEngine, which is the core part of interactive games. Also, I wrote the endOfMatchUpdate function. I wrote part of the popMessage function. During the process, when I was testing my code, I identified a few tiny bugs in other class files and I corrected them.

Project Report – Joan ODEIMI

In the beginning of the project, we divided the group into three groups working on different elements of the project. I am a member of the game engine team, as such I worked on the algorithms of the game engine and thought about the different ways of making the game work. I have worked with the other members to design and code the different algorithms.
More precisely, I coded the function that returns the winner between two teams (not the manager's team) using the attributes of the teams: the game engine GetAutomaticWinner. I then modified it so that it returns the score obtained by the two teams.
I also coded the function getAttackResult with Kevin that gives back the result of an attack depending on the characteristics of the two teams playing (the manager's team and some other random team). This function also modifies the characteristics of the teams such as energy.
Moreover, I coded the function EndOfQuarterRest that iterates on the players and add them energy, this function is only called at the end of a quarter.
I also did some debugging in the Team class. I also worked on getting images and names of players and added their names to the lists.

<u>Little personal note:</u>
I found it really interesting to work on that project and to think about the ways of making the different algorithms. I really feel that I have learned many very important skills, not just in C++ programming, but also on how to work on a project in a team. This was a very enjoyable experience that is truly useful and that I won't forget.

Project Report - Abdel-Kader KABA

During this project I worked on the Graphical User Interface.
I worked closely with Yinfeng on designing the windows on QT. We were in charge of designing the GUI. During the first weeks I created four of the first eight windows that we had. There are “Mainwindow”,”StartMenu”,”TeamInfo”, and ”NextGame”. It made me become familiar with QT. In the week that followed I was doing the basic design and making the windows interact with each other. I had to deal with storing data that I was getting from the user’s input.

I was able to use the data and functions that the other members of the group created in order to animate my windows. The first design idea was to create a big “environment” class that would regroup all the data that we need during the game. However, we found out that the better idea was to create classes and integrate the classes that are logically inside the other one. Example: There is player in a team and a team in a league, hence the files league would contain teams that contain teams which in turn contain players.

The NextGame window is the window where the most interesting functions that I did are, such as the quarter timing functions. Those functions are the one that make us play each quarter. They work with a timer that last for 1 minute and a half. Each second the function is called until the timer hits zero. They set the time, the score and the buttons to be displayed. Along with these four functions I have a function called “strat”. This function uses another timer which lasts for 3 seconds. Each 3 seconds we have an action taking place (the user is either attacking or defending), those actions are a result of an algorithm designed by the Game Engine Team. The outcome is displayed through the scoreboard but also through comments. Being an NBA basketball fan, I wrote the comments under the inspiration of NBA commentators; I believe that they make the game lively and friendly. Apart from the functions in the NextGame file I made the design of the window, from the button position to the colors and the background. It was an enjoyable experience because QT uses stylesheet to customize the look of the windows, so I had the chances to practices a bit of CSS while doing the project.

The project was a great experience both academically and humanly speaking. It was a great team effort and we collaborated a lot on multiple aspects of the game.

Project Report – Yinfeng Yu

I. Design of the structure of the game windows.

In the beginning, I took the responsibility of making the UI with Kader. I started by designing the structure and the hierarchy of the windows: which windows are connected and the logic relation between them.

II. Start to learn about the Qt: documentation on the Internet.

We have never used Qt before, thus the first step I did was read the documentations on the Internet. I spent some time looking at the samples provided by Qt and tried to know how they works.

III. Switching from one window to another. (core of the game).

Since the connection of the windows is one of the most important element of our project, the first things I coded were the links between the windows: how to create a window on the heap within the parent window and show it while the parent window is hided.

IV. Make the windows with Kader separately.

After I knew how to go from one window to another, I started to make the detailed windows with Kader (I was responsible of the main game window, the training window (Edison did most of the work), the team information window and the preparation window.) I tried to make the window according to my design as much as possible, and linked them up.

V. Try to display the data with some samples.

I spent some time understanding the data types of the objects, (QStrings, QNumbers, etc) and some methods in Qt. Then I put some samples in the windows to display as an example for other team members who had been working on the data structure of the game.

← everyone starts with Qt.

VI. Adding dynamic functions to the windows.

After some other members joined in, we could quickly display the game data in the windows. Then I started to implement a "refresh" function in the main game window, the training window and the preparation window, to refresh the data display in the window. To visualise data dynamically is the key of the project, since in our game, the data is always changing. At first, I just coded a simple refresh function, which functionally worked, but did not take the changed incoming data into account.

VII. Working on passing data between the windows.

The part which took a lot of my time, was the data transferring part. The original refresh function actually did not work with the dynamic data, so I put a lot effort and time to improve it. The idea was not complicated, but the realisation was. I adjusted many details and debugged a lot, such as adding "const " in front of the arguments of a function, specify the pointers and references with "this->", etc. With

the help of the teacher, I finally managed to refresh the windows with the incoming data, and this is a big step of our project.

VIII. ″ Substitution" window where we interact with data via UI.

Refreshing is not enough. We need to interact with the data via UI in the preparation window, where we do substitutions. I spent some time thinking how to realised that and I came up with the idea of using buttons to select different players. I also used the mechanism of the ″refresh″ function here. But since I already have enough experience on refreshing, I did this part relatively fast.

IX. Adding some additional windows in the game simulation window.

I added some small sub-windows to the game simulation window, so that we can do the substitution during the match and choose the tactics we want to apply.

X. Adding some warnings and details.

I added some warnings to the game to avoid improper operations which could crash the game.

XI. Closing the cycle.

The last thing I did was to link the game simulation window to the main game window. Since the main game window is not a direct parent window of the game simulation window, I encountered some difficulties, but I still managed to finished that. After this, the structure of the game windows is complete.