RugBot – System Plan & URD

W.P. RUGBY ACADEMY

RugBot Development Team

Group 2

Table of Contents

Table of Figures

Table of Tables

Group number:	2
Group name:	RugBot Development Team
Members:	**Student number:** XQ9X3WV31 **Name:** Matthew **Surname:** Van der Bijl
	Student number: MB2015-0023 **Name:** Tyler **Surname:** Gray
	Student number: PXHTJDCN5 **Name:** Stefanus **Surname:** Buys
	Student number: MB2014-0695 **Name:** Abongile **Surname:** Mdleleni

Customer:	**Full Name:** Angelo Nelson **Company:** WP Rugby Academy **Industry:** Sport Science

Methodology, Development Plan & Schedule

Methodology

Each element of the project is designed and constructed separately and delivered to the client for an overview. This technique allows the client to view individual components and makes room for feedback if any user requirements have changed. This is a benefit as it decreases the time of development, where an issue can be resolved quickly and efficiently.

Agile Approach

An Agile methodology is an approach to software development which supports all unpredictable mishaps during the development process. Unlike the waterfall methodology, the agile methodology makes use of an incremental approach. Ultimately, this means that the model is designed, implemented and thereafter tested as each element is added to the project. Ultimately, it involves both processes of development and maintenance. The iterative approach is focused on conducting development in cycles rather than a sequential step-by-step process. It allows developers to create a working prototype which is compiled against user requirements and offers users the ability to add features while the cycle of development is occurring (Ghahrai, 2017).

Advantages of Agile Methodology:

Using an agile methodology will have the following advantages:

1. Supports the change of user requirements to be inserted into the project during development;
2. Easy to manage and manipulate features in the system;
3. After each deliverable is completed, agile allows the client to have an overview of the system and therefore make changes if necessary;
4. Testing is done after each deliverable and therefore makes it easy to identify bugs in a specific section of the system; and;
5. Agile ensures that the schedule of the project is kept and therefore is more reliable to meet the due date.

As clearly shown, there some clear advantages in using an agile methodology.

Incremental Approach

The development phases of this approach are done in a linear format. A developer can only start the next phase of developing the project once the initial phase has been completed. This is since each phase is dependent on the next phase. The figure below is a diagrammatic representation of the incremental model as defined by Ghahrai (2017).

Advantages of Incremental

The usage of an incremental model has many advantages.

1. Allows working software to be developed early in the life cycle;
2. Scope and requirements changes will not have a huge cost impact;
3. Risk identification occurs early in development; and;
4. Testing and debugging happen during each iteration, therefore, saving time to debug and test the entire project once it is completed (Ghahrai, 2017).

As clearly show that is various advantages to an incremental approach.

Iterative Approach

The iterative approach does not contain any set number of phases and development process which a developer must follow to achieve a successful project. Instead, the approach deals with individual cycles which do not necessarily depend on information or development done in the previous cycle. All development and information are individually processed.

Advantages of an iterative approach

There numerous advantages associated with an iterative approach

1. Changes in user requirements can be facilitated;
2. Risks are identified during each iteration;
3. More time is spent on designing and developing rather than documentation; and;
4. Prototypes are created during each iteration (Ghahrai, 2017).

By initially creating prototypes for the project, the client can thereafter manage and manipulate the features on the system. Ultimately creating more development time to ensure the completion of a successful project. the

In conclusion, this project adopts a hybrid version of agile methodology which incorporates aspects of both incremental and iterative approaches.

Development Plan

The Development Plan shows the increments during which certain requirements will be implemented. The increments each have a timeframe during which it must be completed, and the development plan shows during which deliverable it will be completed. In this development, plan sprints have been called increments and are generally each 3-weeks long.

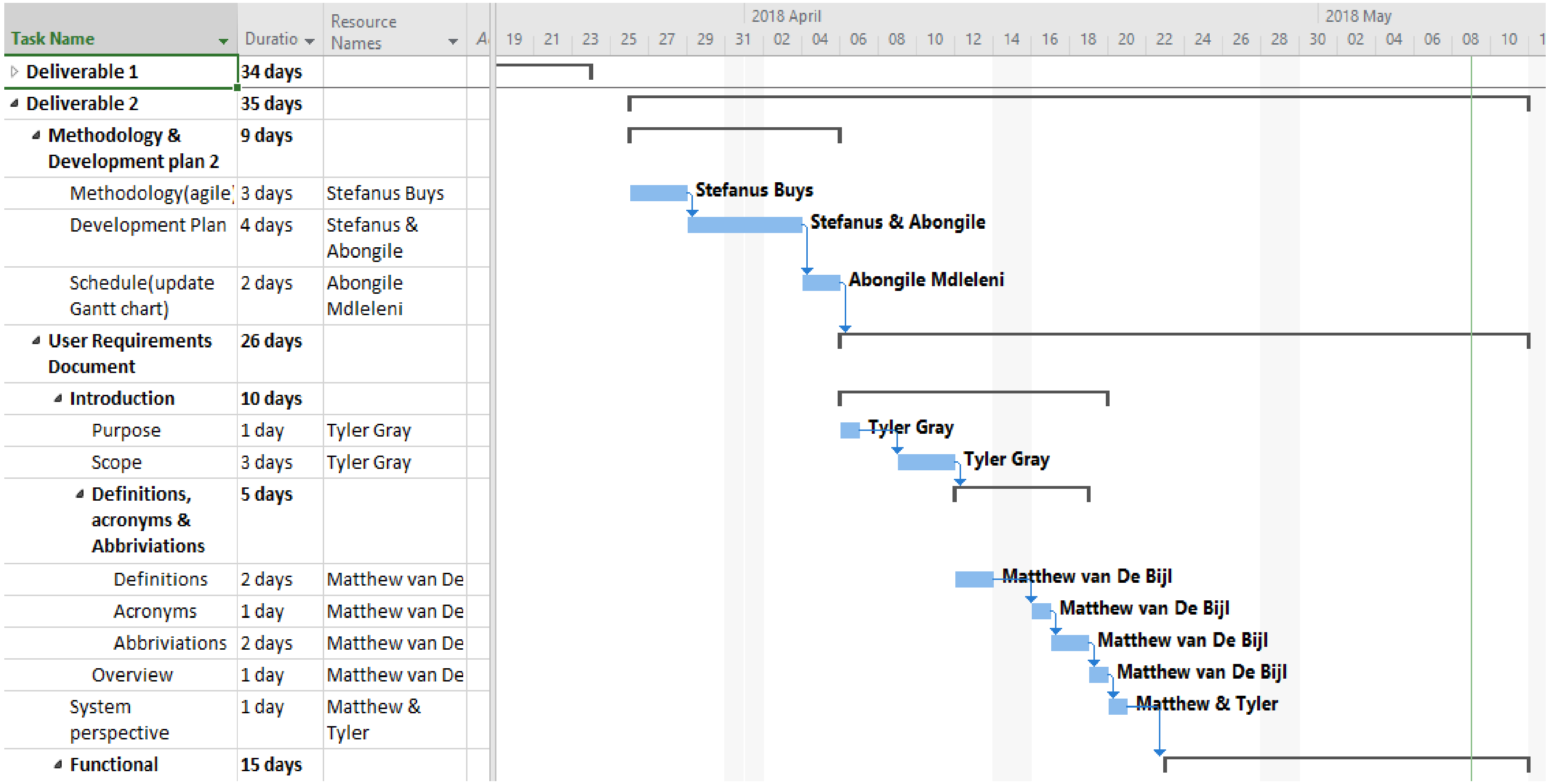
Table 1 Development plan for RugBot

D	I	Purpose	Requirements	Timeframe
	1	Create database and user profile data.		5 days
	2	Allow coaches to add or remove players; functionality for coaches to do roll call; functionality for the physiotherapist to mark players as unavailable for practise or matches and when the player will be back in action.	FR02, FR03, FR06, FR07, FR21	29 days
	3	Allows coaches to view player availability for practice and matches; allow all users to update or delete personal information.	FR04, FR18, FR19, FR20	21 days
	4	Supplies players and coaches with a calendar with match dates and times; functionality for coaches to assign jersey numbers to players for matches; coaches can schedule matches; coaches and players can view match teams and match times.	FR05, FR10, FR13, FR14	26 days
	5	Login screen; authorization and access rights; functionality for a physiotherapist to specify player injuries; coaches can view injured player details.	FR01, FR11, FR12, FR22	21 days
	6	Player notifications about practice, lunch and match information; coach notifications about upcoming matches, player injuries and player absence; coaches can view total boys at a practice session	FR08, FR09, FR15, FR16, FR17	24 days

Key:
D – Deliverable
I - Increment

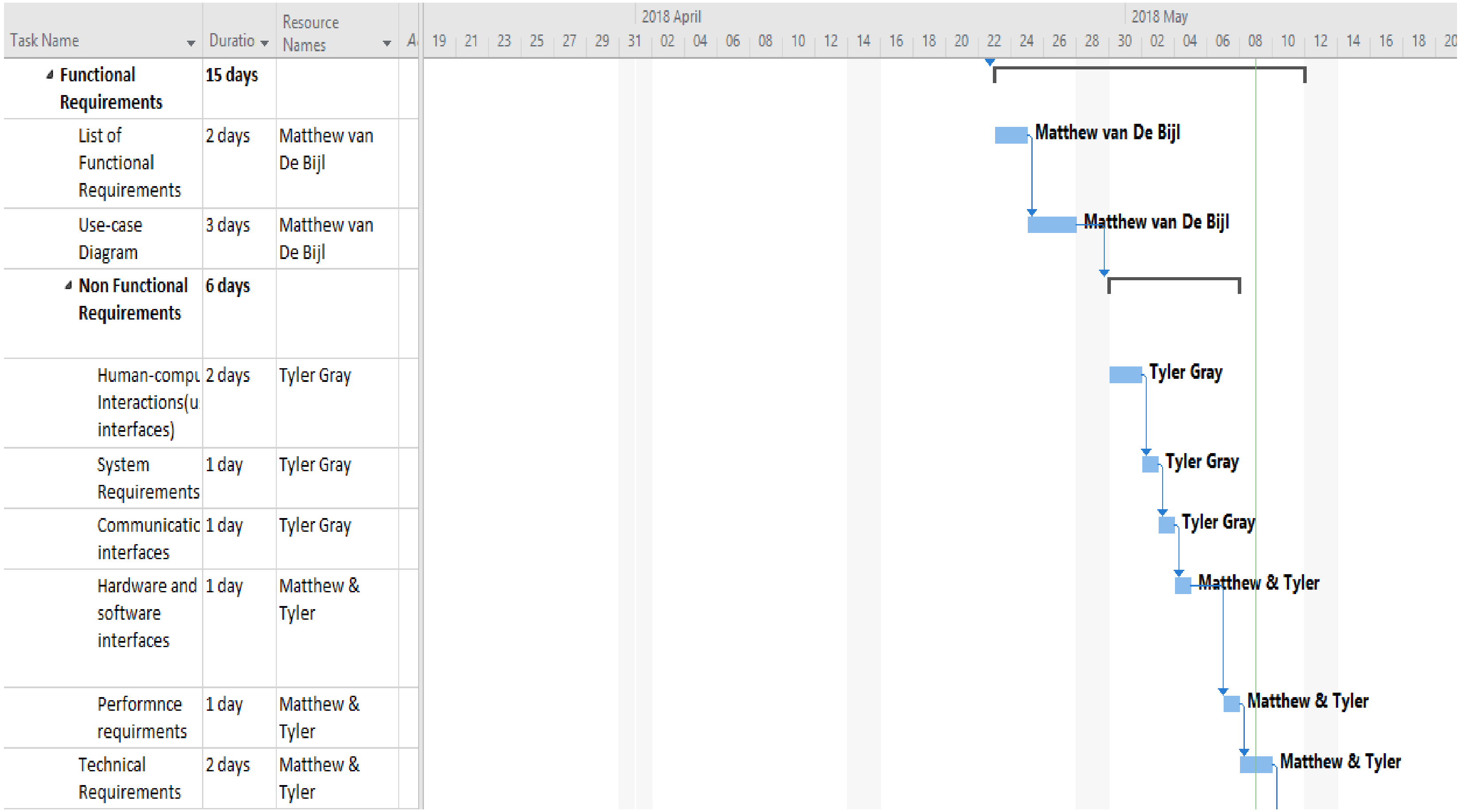
Schedule

The Gantt chart below was constructed by following examples presented by Buttrick (2013) and Schwalbe (2012). Deliverable 2 is due on the 11th of May 2018.



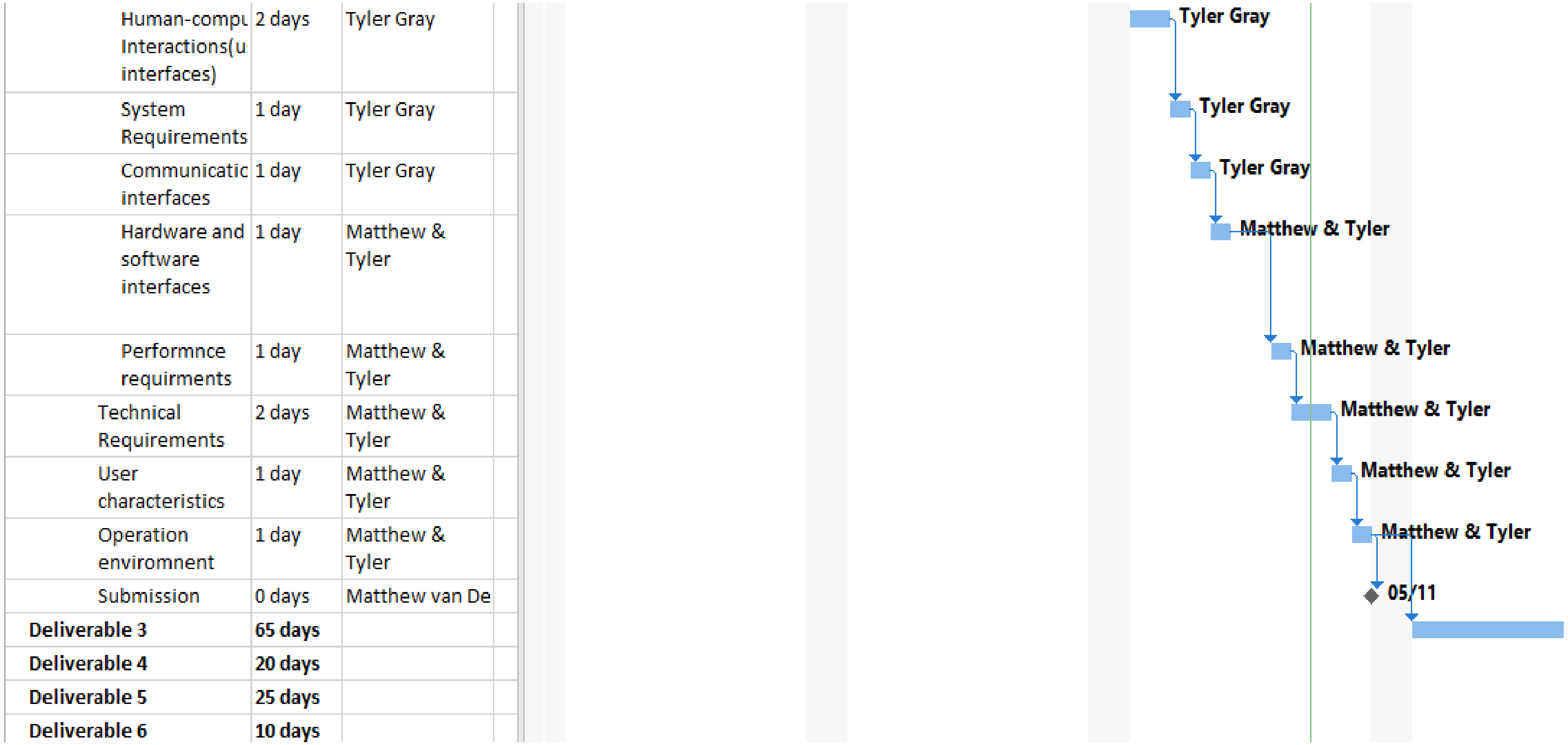
Task Name	Duratio	Resource Names
Deliverable 1	**34 days**	
Deliverable 2	**35 days**	
Methodology & Development plan 2	**9 days**	
Methodology(agile)	3 days	Stefanus Buys
Development Plan	4 days	Stefanus & Abongile
Schedule(update Gantt chart)	2 days	Abongile Mdleleni
User Requirements Document	**26 days**	
Introduction	**10 days**	
Purpose	1 day	Tyler Gray
Scope	3 days	Tyler Gray
Definitions, acronyms & Abbriviations	**5 days**	
Definitions	2 days	Matthew van De
Acronyms	1 day	Matthew van De
Abbriviations	2 days	Matthew van De
Overview	1 day	Matthew van De
System perspective	1 day	Matthew & Tyler
Functional	**15 days**	

Figure 1 Gannt Chart part 1 of 3



Task Name	Duratio	Resource Names	A
Functional Requirements	**15 days**		
List of Functional Requirements	2 days	Matthew van De Bijl	
Use-case Diagram	3 days	Matthew van De Bijl	
Non Functional Requirements	**6 days**		
Human-compu Interactions(u interfaces)	2 days	Tyler Gray	
System Requirements	1 day	Tyler Gray	
Communicatic interfaces	1 day	Tyler Gray	
Hardware and software interfaces	1 day	Matthew & Tyler	
Performnce requirments	1 day	Matthew & Tyler	
Technical Requirements	2 days	Matthew & Tyler	

Figure 2 Gantt Chart part 2 of 3



Human-compu Interactions(u interfaces)	2 days	Tyler Gray
System Requirements	1 day	Tyler Gray
Communicatic interfaces	1 day	Tyler Gray
Hardware and software interfaces	1 day	Matthew & Tyler
Performnce requirments	1 day	Matthew & Tyler
Technical Requirements	2 days	Matthew & Tyler
User characteristics	1 day	Matthew & Tyler
Operation enviromnent	1 day	Matthew & Tyler
Submission	0 days	Matthew van De
Deliverable 3	**65 days**	
Deliverable 4	**20 days**	
Deliverable 5	**25 days**	
Deliverable 6	**10 days**	

Figure 3 Gantt Chart part 3 of 3

As previously stated, each deliverable needs to be completed, reviewed and submitted before the submission dates. It is key that all components are completed on time. The Gantt chart should be updated for each submission. Add in here about Agile sprints vs Increments and Deliverables

User Requirements Document

Introduction

The following document will include necessary information concerning all user requirements needed for the system being produced.

Purpose

The type of system being produced is an application for mobile devices. The system will be supported by a database to hold all necessary information. A connection between the application and database will be established to retrieve and send information to the database. There will be three several types of readers for the system, as indicated in the table below.

Table 2 Table of Users

User	Description
Coaches	Coaches will have administrative privileges on the system.
Player	Players will only be able to see information displayed for them by the coaches.
Physiotherapist	The Physiotherapist will have certain administrative privileges.

All users will need to be able to interfaces with the system. All users will need to provide their own mobile devices.

Scope

RugBot is an application-based system. Rugbot will be supported on android and apple devices. A database will be included in the system to store and retrieve information. The main purpose of producing RugBot is to supply the Western Province Rugby Academy with an easy to use management system. The system has three parts which support the different administrative privileges for the intended users.

Coaches and physiotherapists would be provided with a username and password to login to their account. This is a once off sign in and Rugbot would not ask them for this information again unless the application is uninstalled.

Firstly, the coaches will have full administrative privileges on the system. They will be able to insert, delete and update all information stored in the database. They will also be granted access to retrieve information which is already stored in the database. The main issue which the Rugby Academy is currently dealing with will be doing admin duties in an easy manner. Rugbot will display a register of all the rugby players and allow the coaches the ability to click on a player’s name if they are present at a match or a training session. This method of doing admin will provide efficient time management for the coaches. As soon as a coach clicks on a player name it will be saved into the database as being present on that specific day. Once the register has been completed, the coach will be provided with a “Done” button and thereafter all players name’s which haven’t been selected will be saved into the database as not present.

Another issue which the coaches are having is the ability to keep track of the jerseys being assigned to players on game day. Missing jerseys and nobody taking ownership of the responsibility has been a frequent matter. Rugbot will allow the coaches to assign the relative jersey to the player and store that information into the database until it is time to retrieve the information. This functionality will provide the coaches with concrete information on all jersey assigning to the players.

The weekly schedule of the players is not a fixed schedule. Therefore, the coaches would constantly need to provide a copy of the schedule to each of the players. This can be a time-consuming process and there may be players who do not receive the schedule. Rugbot will allow the coaches to update the weekly schedule on the mobile application and it will thereafter appear on the player's interface of the application. This functionality allows for a clear communication path between coaches and players.

Match fixtures and match results have also been a strain in the admin apartment. Assigning players to the different teams (first team, second team and third team) to a wide range number of matches has been a time-consuming effort on the coaches' part. For example; a player can participate for the first time in an upcoming match but might not participate in that same team the following match. Assigning positions to players for games has also been a lengthy process. For example; a player might not play in the same position for every match. This can be due to the coach's decision or an injury occurrence. Therefore, RugBot will allow the coaches to update all match fixtures and results and it will be sent over to the player's end of the application. The positions of every player in every match will also be displayed for the player and therefore eliminates all confusion on game day.

Secondly, physiotherapists are a very important role in the rugby academy. They tend to the player's physical needs and ensures their capability out on the rugby field. Rugby is a contact sport and therefore injuries will occur. It is important that coaches and physiotherapists have a clear and open communication path. The issue at hand is that information about a player's health has not been forwarded to the coaches immediately which leaves a result of constant miscommunication. However, Rugbot will provide therapists with the ability to click on a player's name and insert any injuries or health issues which has occurred. They will have the option of stating whether and when the player can resume normal training and match games or if it is just a minor issue. They can also add any notes which will be relevant for the coaches to see. All this information will be displayed on the coaches' side of the application as soon as it has been inserted into the application.

The project has been delimited in the following ways:

1. Players will not be given privileges to insert, create and/or update any information on the application;
2. There will be no instant messaging between users on the application;
3. Physiotherapists will not be assigned the same privileges as the coaches;
4. The system will not be web-based; and;
5. The system will not be freely available for download.

Thirdly, the players will not have any administrative privileges on the application. Their side of Rugbot will only be viewing the weekly schedule, matches fixtures, match results and any other essential information which the coaches display.

Definitions, Acronyms & Abbreviations

The tables that follow outline all definitions, acronyms and abbreviations used in this document.

Definitions

Table 3 Definitions

Definition	Description
Gantt chart	A Gantt chart is a diagram used to display a project's schedule (Schwalbe, 2012).
Coach	Member of staff who instructs player.
Player	Rugby student.
Western Province Ruby Academy	Western Province Rugby Academy is to facilitate the growth of young athlete so that they may reach their full potential.
Nielsen's Heuristics	A high-level list of non-functional requirements presented by Nielsen (1994).

Acronyms & Abbreviations

Table 4 Acronyms and abbreviations

Term	Expansion
App	Application
API	Application Programme Interface
IDE	Integrated Development Environment
UI	User Interface
UX	User Experience
DBMS	Database Management System
JS	JavaScript
FR	Functional Requirements
NFR	Non-functional Requirements
GUI	Graphical User Interface

System Perspective

The system will take the form of a mobile application and a remote database. Users will interact with the database through the application. In order to achieve this configuration, the system will make use of a client-server model.

Functional Requirements

List of Functional Requirements

The table below outlines functional requirements that the final system needs to meet.

Identifier	Requirement Description	Priority	Source
FR01	Users must use a one-time login to log in to the application for authorisation purposes.	High	RugBot Development Team
FR02	Coaches must be able to take an attendance list of students at practice.	High	Coaches
FR03	Coaches must be able to view a backlog of student's attendance for past dates.	High	Coaches
FR04	Coaches must be able to view a list of all their students and their availability for practise sessions and matches.	High	Coaches
FR05	Coaches and students must have a calendar with a practise match dates and times.	High	Coaches
FR06	The physiotherapist must be able to mark a student as injured and not able to practise or play matches.	High	Coaches
FR07	The physiotherapist must be able to add an estimated date of when a student will be able to practise again.	High	Coaches
FR10	The coach must be able to assign jersey numbers to players on match dates.	High	Coaches
FR11	All users need to be able to see the injury status of a player.	High	Coaches
FR18	Players must be able to update their personal information	High	RugBot Development Team
FR13	Coaches need to be able to schedule a match.	High	Coaches
FR14	Coaches and students need to be able to see match teams.	High	Coaches

Identifier	Requirement Description	Priority	Source
FR08	If a student missed more than three practise sessions, the coach must receive a notification of the student's absence.	Medium	Coaches
FR09	The coach must be able to see the total of boys at practice.	Medium	Coaches
FR12	Coaches and physio must be able to view the medical history of a player. A player must only be able to see their own data.	Medium	RugBot Development
FR15	Players must be notified when they are playing games.	Low	RugBot Development Team
FR16	Players must be notified when they have a physio appointment.	Low	RugBot Development Team
FR17	Players must be notified when they have missed a practice.	Low	RugBot Development Team

The table has been sorted by priority. All high priority requirements need to be completed before the delivery of the final system.

Use-case Diagram

Use-case diagram creation (Figure 1) was informed by Bennet, et al. (2010).

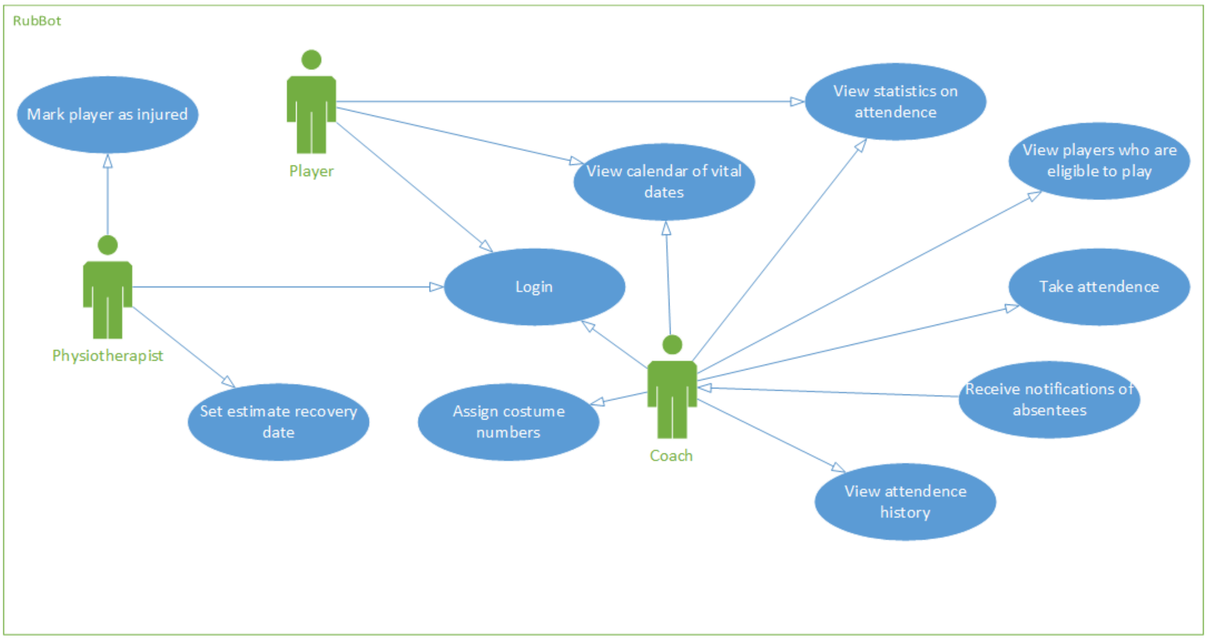

RubBot
Mark player as injured
Player
View statistics on attendence
View calendar of vital dates
View players who are eligible to play
Physiotherapist
Login
Take attendence
Coach
Set estimate recovery date
Assign costume numbers
Receive notifications of absentees
View attendence history

Figure 4 Use-case Diagram for RugBot

The system will consist of three users, or actors, namely: player, coach and physiotherapist. All users will need to be able to login to the system.

Non-functional Requirements

According to Chung, et al. (1996), non-functional requirements are also known as quality requirements. As previously stated, non-functional requirements do not directly describe what the system must do (Sommerville, 2001).

Table 5 Non-functional Requirements

Identifier	Requirement Description	Priority	Source
Performance			
NFR01	Database response times must be very quick.	High	RugBot Development Team
NR02	Quick response times in applications.	Medium	RugBot Development Team
NFR03	GUI must be quick and responsive.	Medium	RugBot Development Team
Design			
NR04	The GUI design must be minimalist and simple.	Medium	RugBot Development Team
NFR05	Navigation of the application must be sensible and straight-forward.	High	RugBot Development Team
Security			
NFR06	The database must only be accessible by authenticated users.	High	RugBot Development Team
NR07	A user must be able to access only data specific to their authorization level.	High	RugBot Development Team
Reliability			
NR08	The application should never crash and be bug-free.	High	RugBot Development Team
NR09	The application must be able to operate even when connected to the database is lost.	Low	RugBot Development Team
NR10	The database must be able to have multiple users access it at the same time.	High	RugBot Development Team
Scalability			
NR11	The system must be able to grow in terms of active users.	Medium	RugBot Development Team

In conclusion, User data needs to be kept confidential, as suggested by (Chung, et al., 1996). The system must not waste the users time, as suggested by (Chung, et al., 1996).

Human-Computer Interactions

All system interfaces should comply with design principles to provide users with a holistic user experience. All users facing interfaces should strictly comply with Nielsen's Usability Heuristics as suggested by Nielsen (1994). These include:

1. Users should be provided with feedback;
2. The system must comply with the user's model of reality;
3. The system's interfaces must remain consistent is designed;
4. Users must always be able to exit an unwanted state;
5. The system must protect against error-prone conditions;
6. Users should always be able to recognise what they need to do, rather than recall what they need to do.
7. The system needs to be efficient;
8. The GUI design must be minimalist and simple
9. The system must assist the user in overcoming errors; and;
10. Documentation, such as a user manual, must be available to the user (Nielsen, 1994).

The guidelines listed above should be a high-level set of non-functional (Rogers, et al., 2011). Stair and Reynolds (2016) suggests that all user-facing interfaces should be tested against these guidelines.

System Requirements

The final system delivered will require end users to own their own mobile devices as well as a remote server for the database. A connection will need to be made between user devices and the database users will need to provide their own mobiles devices.

Communication Interfaces

The app will need to facilitate easy communication.

Hardware & Software Interfaces

Users will interact with the system through mobile devices. Users will need to provide their own mobile devices for the app to be installed on. The system will require a remote server. This server will need to host the database that mobile devices interact with.

Performance Requirements

The system will need to maintain a high-level of performance throughout the entire system. The performance of a system has a direct impact on the user experience.

Technical Requirements

The table below outlines the technical requirements of the system.

Table 6 Technical requirements

Identifier	Requirement Description	Priority	Source
TR01	Users will need a mobile device running either Android or iOS to use the application.	High	Coaches
TR02	Users will need an internet connection to connect to the database.	High	Coaches
TR03	Firebase will be used for the database needs.	Low	RugBot Development Team
TR04	The application will be developed on the Ionic framework.	Low	RugBot Development Team
TR05	HTML, Sass and TypeScript (a superset of JavaScript) are the languages that will be used for development.	High	RugBot Development Team
TR06	Developers will need Android and iOS devices for testing.	High	RugBot Development Team
TR07	WebStorm or any modern code editor with TypeScript support will be used for writing and editing code.	High	RugBot Development Team

As previously stated, the project’s technical requirements should adapt to recent technology and market change. It is vital that the client is presented with a truly modern system.

User Characteristics

The table below outlines all users of the system.

Table 7 User characteristics

User Role	Level of computer literacy	Level of Experience	language	Technical Skills Required
Developer	Expert	Expert	English	Expert
Coach	Proficient	Novice	English	Literate
Player	Literate	Novice	English	Literate
Physio	Proficient	Novice	English	Literate

Operational Environment

The system will need to exist in two operational environments, a development environment and practical environment. The development environment will be used for developing and testing the system.

As RugBot is a mobile application, the final system will be deployed to mobile devices. The system will need to make use of a database. The database will need to reside on a remote server. The database will need to be secured to ensure that user data remain confidential.

Bibliography

Bennet, S., McRobb, S. & Farmer, R., 2010. *Object-Oriented Systems Analysis and Design Using UML.* 4th ed. London: McGraw Hill.

Buttrick, R., 2013. *The Project Workout: The ultimate handbook of project and programme management.* 4th ed. London: Pearson UK.

Chung, L., Nixon, B. A. & Yu, E. S. K., 1996. Dealing with change: An approach using non-functional requirements. *Requirements Engineering,* 1(4), pp. 238-260.

Nielsen, J., 1994. Heuristic evaluation. In: *Usability Inspection Methods.* New York: John Wiley & Sons, pp. 25-62.

Rogers, Y., Sharp, H. & Preece, J., 2011. *Interaction Design: Beyond human-computer interactions.* 3rd ed. Chichester: Wiley.

Schwalbe, K., 2012. *Infomation Technology Project Management.* 7th ed. Boston: Cengage Learning.

Sommerville, I., 2001. *Software Engineering.* 9th ed. Boston: Addison-Wesley.

Stair, R. M. & Reynolds, G. W., 2016. *Principles of Information Systems.* Boston: Cengage Learning.