**COMP 3059 – Capstone Project I**

**Software Requirements Analysis and Design Assignment**

This assignment is an overview to gather the software needs with requirements analysis and help to proceed with the design.

The requirements analysis helps to break down functional and non-functional requirements to a basic design view to provide a clear system development process framework. It involves various entities, including business, stakeholders and technology requirements.

The design is the activity following requirements specification and before programming. Software design usually involves problem solving and planning a software solution.

To work on this assignment you could use the references and a sample template given below. The sample template can be customised to suit the nature of your project.

Reference Readings/Example:

<http://www.uacg.bg/filebank/acadstaff/userfiles/publ_bg_397_SDP_activities_and_steps.pdf>

[www.cse.msu.edu/~chengb/RE-491/Papers/SRSExample-webapp.doc](http://www.cse.msu.edu/~chengb/RE-491/Papers/SRSExample-webapp.doc)

<https://nces.ed.gov/pubs2005/tech_suite/part_2.asp>

Reference template:

[www.tricity.wsu.edu/~mckinnon/cpts322/cpts322-srs-v1.doc](http://www.tricity.wsu.edu/~mckinnon/cpts322/cpts322-srs-v1.doc)

# 1.0 Introduction

The purpose of this document is to outline and describe the high level requirements related to the “How To Train Your Dragon Boat” Application. The sections below will describe the systems features and the purpose of them, as well as describing their functionality through a use of diagrams. This requirements associated with the application are not limited to, but will include the functional and non function requirements of the system, and database requirements for the application.

## Purpose

The Main Purpose of the “How To Train Your Dragon Boat” Application is to provide the George Brown College Dragon Boat Team with an application that is intended to aid the team with the common tasks performed for the team. The team’s most important tasks, which include the management of team members, the creation of dragon boat layouts, and the monitoring of dragon boat teams during practice are currently done through manual methods, mainly through scrap paper, and as a result are of very low effectiveness and efficiency. The How To Train Your Dragon Boat application will tackle these specific problems, in order to provide a much higher level of efficiency during the completion of the Dragon Boat team’s tasks.

## Scope

* **In-Scope Specifications:** The How to Train Your Dragon Boat Application will be a mobile application supportable by both iOS and Android operating systems. It will also be supported on tablet platforms as well. The application will provide an interactable interface for the creation of customization of boat layouts, with the ability to assign users to the layout. Details about the current layout, including the weight distribution of the members, or their seating preferences, will be displayed along with the layout. This requirement replaces the current method of constructing a layout, which is performed manually, by providing a semi-automated system which will increase efficiency. A map interface will be accessible within the application, designed to assist the team with a valid method of tracking the route in which the boat pursued, as well as the performance of the boat during the current run. Details of the boat’s performance include the distance the boat travelled, the speed or velocity of the boat, and the boat layout that was used during in the current run.

The application will also provide a CRUD interface to manage the users of the dragon boat team, or any other user that may or may not be associated with the dragon boat team. Communication can also be established between users through the use of associated email addresses, eliminating the need to communicate through third party sources or face-to-face conversation. Provided within the application will be an SQLite database that will store all of the users, routes, boat layouts, and the associated information related to each item stored. This will provide a central storage that can be easily accessed by the application and its users.

* **Out-Of-Scope Requirements:** Support for the application will not extend beyond the mobile scope, and will not be supportable by any operating system beyond Android or iOS systems. While the application will use GPS and location for the functionality of the map interface, the application will not use any internet connectivity for functions of the application. The map provided with the application will be used for location tracking only, and will not include any form of custom paths for boat routes. In addition, the map will not include any form of location finding. To be specific, this application will not allow users to use the GPS to find a specific location within an area.

\*(Highlighted statements are new additions to the In Scope/Out or Scope Requirements)

# System Overview

## Project Perspective

Currently, the George Brown College Dragon Boat team does not have any current system software for the management of their tasks, forcing the team to perform these tasks manually. As a result, these tasks prove to be both unorganized and time consuming. The How To Train Your Dragon Boat application will be built upon resolving these issues, and will serve a new self-contained system for the team.

## System Context

The System Context describes the resulting software within the business case, including strategic issues in which the system is involved or which it specifically addresses.

## General Constraints

Specification Constraints:

* Current goal of production and development involves little to no budget costs or expenses. Results in a much smaller selection of available software IDEs for development.
* Deadlines set for project components / sprints are not flexible, and must be accomplished by the set due date.
* Tasks defined for the project and the division of the tasks is limited by low team size

Designing Constraints:

* Wireframes created must keep any limitations of mobile development in mind when designing components, appearance, or basic navigation

Implementation constraints:

* Implementation of database schema have the potential to vary based on the IDE selected for development
* The use of, or the implementation of the React Native Npm library will vary depending on the selected IDE for development
* Appearance or functionality of application may hold slight differences to the wireframes or diagrams depending on the software IDE’s limitations or the development work required.

Testing constraints:

* Testing GPS in real life conditions relies on access to boat and weather conditions
* Database storage testing, while unlikely, is dependant on the available storage the testing device currently has.
* The target device for testing must meet the software version requirements for the application to run
* The target device must have all required permissions enabled for all functionality to be tested

## Assumptions and Dependencies

List any assumptions that have been made during the initiation of the project. In addition, list any dependencies that may impact its success or the desired result.

Assumptions:

* Projects members are willing to assume different roles to make up for the lack of members involved in the project and to assist members in the completion of tasks
* A consistent form of communicate will be obtained across all team members despite each member’s involvement in educational activities
* Project members agree to follow the guidelines, requirements, and plans that are discussed and written in the documentation
* Project members are aware of the risks of not meeting specified deadlines.
* Project members are willing to research new languages and technologies for the project if needed
* Any sections labelled in the document including scope requirements, constraints, dependencies, features, diagrams are not final and may concur changes as further development of the project continues.
* The online marketplace will be able to support the application as it is distributed to its users
* The GBC Dragon Boat Team will have limited involved in the development of the application.
* The React Native libraries and framework being used for this project will continue to be supported by Facebook
* The chosen map API that the application will use continues to provide free support
* The database chosen for the application will be continuously provided for mobile development
* The Users of the application will meet the requirements to run the application on their mobile device.

Dependencies:

* Non-Functional Dependencies:
* Users of the application must have location enabled for the map API to function properly
* Access to an iOS device or software is needed to test iOS support
* Access to an Android device or software is needed to test Android support
* Functional Dependencies \*\*:
* Npm install packages
* React native npm library
* React native navigation npm library
* React native drag-and-drop npm library
* React native maps npm library
* React native animation npm library

\*\*For simplicity, only major library dependencies have been listed, as actual number of dependencies needed is of a much larger quantity

## 3.0 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

### 3.1 <Functional Requirement or Feature #1>

* Introduction
* Inputs
* Processing
* Outputs

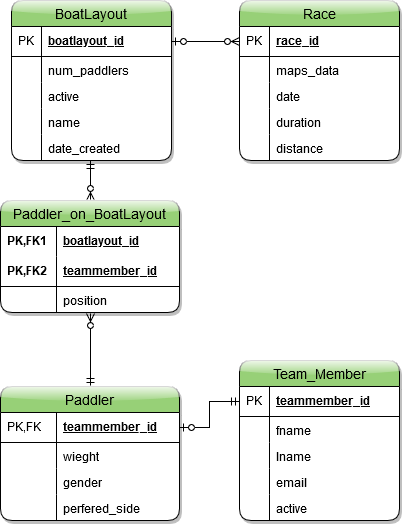
...

## 3.2 Use Cases

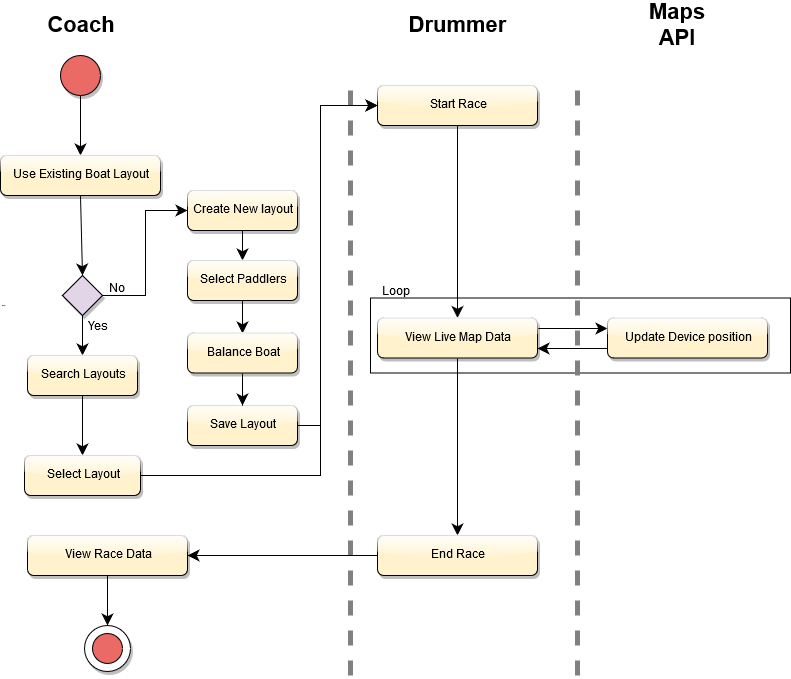
### 3.2.1 Use Case #1 ...

**3.3 Data Modelling and Analysis**

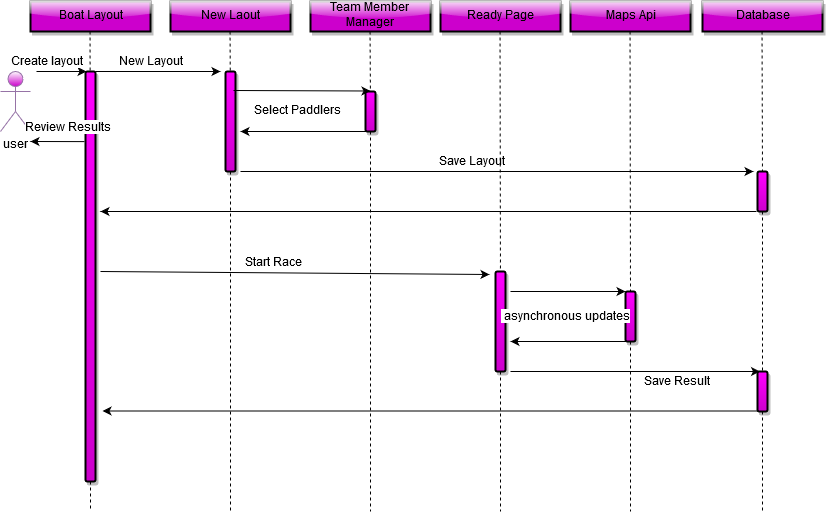
* Normalized Data Model Diagram



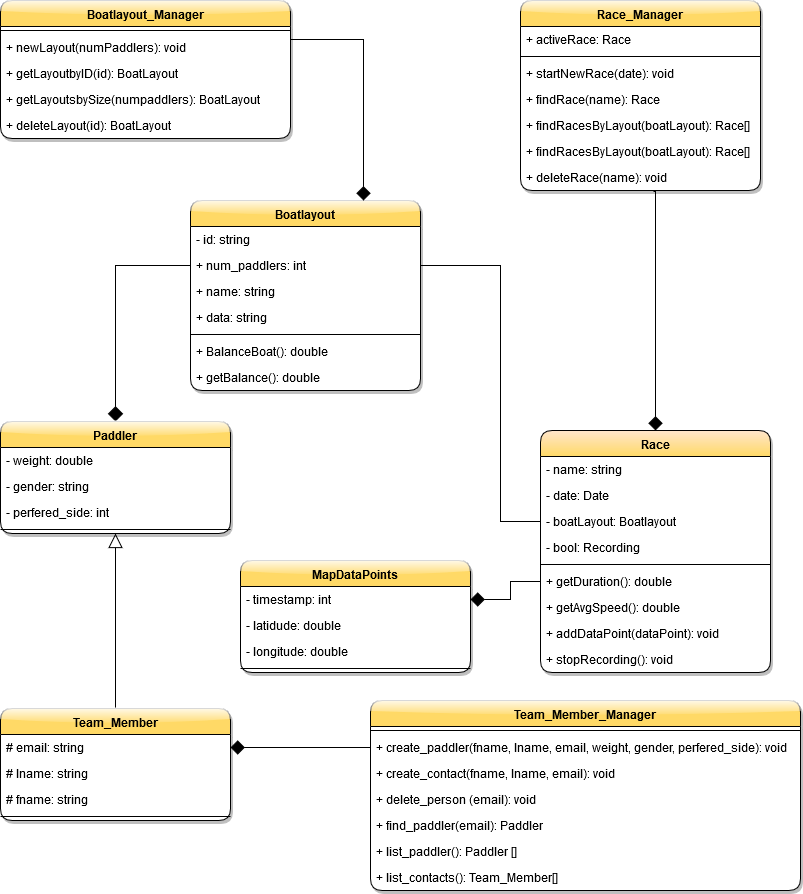
* Activity Diagram



* Sequence Diagram

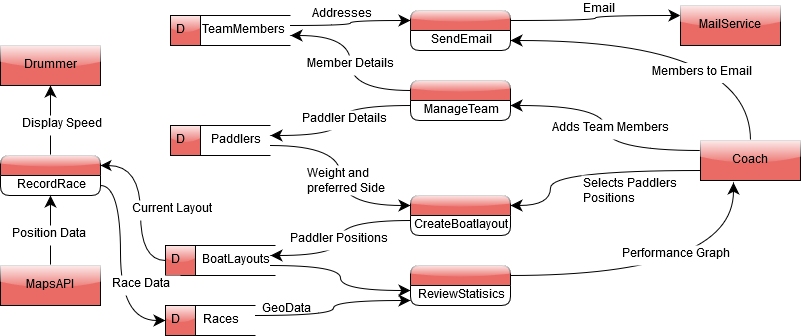


* UML Class Diagram



**3.4 Process Modelling**

* Data Flow Diagram



## 4.0 Non-Functional Requirements

The non-functional requirements for a system are typically constraints on the functional requirements – that is, not what the system does, but how it does it (e.g. how quickly, how efficiently, how easily from the user’s perspective, etc.).

### Non-functional requirements may exist for any of the following attributes – Performance, Reliability, Availability, Security, Maintainability, Portability.

Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, etc).

## 5.0 Logical Database Requirements

A local database will be used to store information on team members as well as past performance in races and practices. As this information is stored locally on the users phone, file size is a concern, as such map data will be stored in the database but other analytics (eg. average speed, speeds consistency) will not be store in the data base as they can be recalculated from the original data. The database will be required to be performant on both Android and iOS so that the user experience is both responsive and consistent across both platforms. As an internet or cloud storage is currently out of scope there will be no opportunity to create multiple or offsite backups for this data, while this is a disadvantage it is acceptable in this case as the main focus of the app is to manage the current race and give live performance feed back. If in a later release we decided to increase the focus on data analytics of this app we will also have to explore adding internet storage and backup to the app.

## 6.0 Other Requirements

Additional requirements, if any.

**7.0 Approval**

The signatures below indicate their approval of the contents of this document.

|  |  |  |  |
| --- | --- | --- | --- |
| Project Role | Name | Signature | Date |
| Lead Programmer | Andrew Cobb |  | 11/11/2019 |
| Scrum Master | Giuseppe Ragusa |  | 11/11/2019 |
| Project Manager | Arsalan Farooqi |  | 11/11/2019 |