Development Report

Graded Unit – Stage 2 – WLC: Software Development

Kian Gault – Student

Simply Rugby Ltd & OOP Solutions Ltd

Contents

[Self-Research on unfamiliar libraries that will be used 2](#_Toc134020588)

[Reason for self-research: 2](#_Toc134020589)

[Integrating SQL into WPF: 2](#_Toc134020590)

[Implementation 3](#_Toc134020591)

[Libraries: 3](#_Toc134020592)

[Unfamiliar XAML objects: 3](#_Toc134020593)

[Object Oriented Principles: 4](#_Toc134020594)

[Implementation of the requirements (Business domain) 4](#_Toc134020595)

[Implementation of the User Domain(User interface) 5](#_Toc134020596)

[Testing 6](#_Toc134020597)

[Test Plan: 6](#_Toc134020598)

[Test logs: 6](#_Toc134020599)

[Documentation 7](#_Toc134020600)

[Summary 8](#_Toc134020601)

[References 9](#_Toc134020602)

# Self-Research on unfamiliar libraries that will be used

## Reason for self-research:

* The reason self-research was needed was our unfamiliarity with working with an external database on a WPF application.
  + This meant researching the different possibilities and what we could use to achieve this.
  + This ultimately boiled down to integrating SQL into our WPF application.

## Integrating SQL into WPF:

* The different options:
  + Integrating SQL into WPF is a common occurrence as SQL is our best option when programming database queries so there are many different methods that can done.
    - The main two options that are the most valuable are integrating a server based database using mySQL or using an embedded database using SQLite.
* mySQL:
  + mySQL is an open-source database management system that contains a server connected to a network.
    - Pros:
      * Great for large databases spread across a network.
      * Large storage capacity.
      * Multi-user access.
      * Often better for web based applications.
    - Cons:
      * Requires a constant connection to an external server.
      * Requires good technical skills to set up.
      * Difficult to move the application once connected to a server.
* SQLite:
  + Like mySQL, SQLite is also an open source resource and is available in the public domain.
  + SQLite is different to mySQL because it is serverless and contained within itself. This is why it is most commonly used when you need to embed a database within an application. This allows the database to run alongside the app.
    - Pros:
      * **More suitable for smaller databases.**
      * **Great for standalone applications.**
      * All file based with easy set-up.

(hostinger.in, n.d.)

# Implementation

## Libraries:

* Microsoft.Entityframework.core.SQLite:
  + This library was the primary one used for this implementation. It was easy and reliable so often other external packages were not needed.
  + I discovered the use of this through research within the planning phase. In particular a content creator called Tactic Devs provided many tutorials on the topic of using SQLite with WPF so this was very valuable. (Dev)
  + With this tutorial I was able to gain a foot ground in creating my SQLite database.
  + I expanded from this tutorial by adding in multiple entities and having many different fields within these entities.

## Unfamiliar XAML objects:

* ListView
  + Within the same tutorial to use SQLite with WPF, it included the use of a ListView to display the data. (Dev) This was instantly a seller for how we would display the data from the database.
  + With a ListView you can split the list into headings with would act as our fields.
* TextBox Binding
  + The tutorial introduced a new xaml manipulation in the form of data binding the textbox with the listview items. (Dev)
  + This discovery lead to this applications defining feature. The ability to select a record and have that record data populate in the text boxes.
    - This allows us to make editing smooth and easy to understand because the user simply has to click on the record they wish to edit and the text boxes will populate with the selected records items.

## Object Oriented Principles:

The early draft design of this application was designed in an object oriented way and for the most part the final design reflects this.

* Encapsulation:
  + All visibility modifiers are set to the correct state. I.e. all variables are private and properties and methods are public.
  + This ensures when data is being modified only those with access to the properties can update data.
* Abstraction:
  + Our two main operating classes are the Coach Class and the Admin class. Abstraction is used here because instead of the event handlers having all the implementation, it actually only performs the validation. Once the validation is passed it calls the associated operation class and that class will perform it’s duties.
  + This follows the rule of abstraction “Take what you need, discard what you don’t”.

The original class diagram for this application featured the use of inheritance and an interface. In the final build this was removed as the program felt like it had unneeded complexity to an already complex build. Perhaps in the future OOP solutions can amend these and see if inheritance could be beneficial somewhere within the program.

## Implementation of the requirements (Business domain)

Through the use of testing and referral back to the initial SRS document, I believe the implementation of this application met the needs of the requirements stated by the client.

A formal demonstration is scheduled for the 11th of May with Mr Muir the clubs chair spokesmen and here we will discuss the final build of the application and see if any improvements could be made in the future.

The key aspect of this application was its ability to retain data over sessions which is successfully does with the use of SQLite.

The initial plan was to have two users, Admin and Coach and in the final design we have one admin, four coaches and four different teams. All these users have different operations and permission access levels which is assigned during login.

## Implementation of the User Domain(User interface)

The user interface was designed with the functional requirements in mind. For example on the player details window, the user would need a way to trigger certain operations like create, update, delete, search and switch team.

This was achieved through the use of buttons, textboxes, comboBoxes, CheckBoxes, and selecting items from a list.

In this application we have a total of 5 windows, Login, hub, player details, skill profiles and email list.

In the original design, we had two hub windows, one for admin and one for coach. In the final build we simply take note of the current user and display the correct permissions to them so the need for two hubs was no needed.

A future design could see the main hub removed also and just have the application open on the player details window after login. The idea was to give the user the choice between going to the player details and player skill profiles.

# Testing

## Test Plan:

In this implementation stage, created a test plan to accompany our test logs for the system. This plan outlines the scope of our testing process and the various methods we use to test the application e.g. traceability matrix, white box testing, black box testing.

## Test logs:

Test logs were used in this application to validate different aspects of the system from event validation to data input validation.

These are important documents because without them, we wouldn’t know if our application behaves as intended. These also provide the means to ensure our application is as robust and user friendly as possible.

# Documentation

Our main documentation for this implementation stage is our user documentation which took the form as a user manual for our application.

Within this manual, a step-by-step guide is shown with verbal and visual instructions shown to the reader.

As the system is complex it’s important any potential users know how exactly to use this system.

This will be especially insightful for veteran employees of the client organization as they have used paper-based forms before the creation of this app so they will need to be trained before implementing this application as their main management system.

That is why it’s crucial this guide is simple and insightful for all types of users may it be an admin of simply rugby or a coach reading it.

# Summary

In summary I believe the implementation phase of this project met expectations and even allowed for some extended requirements to be added like many teams, many users being added.

The use of SQLite has become invaluable knowledge to know as it’s simplistic and easy to set up design is so helpful. Future OOP solutions products will surely use this method of integrating a database file with a C# project again in the future.

I also believe the user documentation and testing documents will prove helpful for Simply Rugby when it comes to assessing our performance over this project.

Each stage of the implementation has been documented and will be discussed further in our final evaluation report of this project.

# References

(n.d.). Retrieved from hostinger.in: https://www.hostinger.in/tutorials/sqlite-vs-mysql-whats-the-difference/#:~:text=MySQL%20has%20a%20well%2Dconstructed,is%20harder%20when%20using%20SQLite

Dev, T. (n.d.). #C CRUD Operations with SQLite Database using Entity Framework . YouTube - https://www.youtube.com/watch?v=fnyiDMvhJOc.