PlayConnect

(Web application)

By

Mandev Seahra

Submitted to

**The University of Roehampton**

In partial fulfilment of the requirements

for the degree of

**BACHELOR OF SCIENCE IN COMPUTER SCIENCE**

Abstract

Abstract narrative

Signed (apply signature below)

**Declaration**

I hereby certify that this report constitutes my own work, that where the language of others is used, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions, or writings of others.

I declare that this report describes the original work that has not been previously presented for the award of any other degree of any other institution.

**Date:** Enter the date here

**Enter your name here**

Acknowledgements

Acknowledgements narrative

In this report, I would like to thank and dedicate this project to my family, friends, classmates, and professors for supporting me through my final year and working toward this project. It has been a very tough year for me with the family issues I had at the start of the year. I know that this would have been a lot harder if they were not there to give me the push, I needed to complete this project.

Table of Contents

[**1.** **Introduction** vii](#_Toc98426634)

[Research Question or Problem that will be Addressed vii](#_Toc98426635)

[Aims vii](#_Toc98426636)

[Objectives vii](#_Toc98426637)

[Legal, Social, Ethical and Professional Considerations viii](#_Toc98426638)

[Background viii](#_Toc98426639)

[Report overview viii](#_Toc98426640)

[**2.** **Literature or Technology Review** x](#_Toc98426641)

[**3.** **Design or Methodology** xii](#_Toc98426642)

[**4.** **Implementation or Results** xviii](#_Toc98426643)

[Evaluation xxvi](#_Toc98426644)

[Related Work xxvi](#_Toc98426645)

[**5.** **Conclusion** xxviii](#_Toc98426646)

[Reflection xxix](#_Toc98426647)

[Future Work xxx](#_Toc98426648)

[**6.** **References** xxxi](#_Toc98426649)

[**7.** **Appendices** xxxii](#_Toc98426650)

# **Introduction**

PlayConnect, a web application, is a prototype. There are many ways to execute this project, but a web application was the best option due to the given time frame and it being just a prototype. It is also more accessible to its users. Furthermore, a web application was a good way to explore multiple computer languages.

Describe your project

This project is about creating an application for people who love playing sports and for people that want to get more involved in sports. For example, if a group of 9 people wanted to play a 5-a-side football game, they can’t. They would need an extra player for a 5-a-side game. This application would help the group find a player within the area. As stated previously, this would help the group of 9 play the sport that they love, and the new player would be more involved with sports. Additionally, it can create an opportunity for a new friendship to arise.

As expected, everyone is at a different level when it comes to playing. Therefore, PlayConnect would allow users to give themselves a rating on how well they perform at a certain game. This allows other users to invite someone at the same level of skill or if they want more of a challenge, they can invite individuals with high ratings as well. After a match, users will have the option to give a rating for the invitee. This allows the invitee to know what they can improve on.

## **Problem that will be Addressed**

The problem that is going to be addressed has already been briefly stated. Sports enthusiasts will have the opportunity to play the game that they love and meet other sports enthusiasts. It also helps a group of people find a last-minute replacement in order to play the game.

## **Aims**

1) A lot of the time, if there aren’t enough people for a game, the game will be cancelled. Despite it being planned months prior. The application will help resolve this issue by providing the last-minute replacement so, the game can go on as planned.

2) There are people that would love to play a sport but have friends that do not share the same interests as them. This app allows people to meet others with the same interests when it comes to sports.

3) In the UK, some sports aren’t as common as other sports, such as, basketball. PlayConnect allows users to find someone who is interested in the same sport as them. This creates an opportunity for people to play the uncommon sports.

4) With obesity on the rise in the UK, this app is an affordable way to stay active. Additionally, some people feel like working out to lose weight is a chore. Playing sports is a great way to have time to go by fast and burn calories without having to wait for the workout to be over.

## **Objectives**

1. Creating a homepage – letting the users know what the web application is all about. Since the title can only say so much.
2. Creating a login page – this application is based on the user profiles and determining if they are good enough to play with a selected group.
3. Database – this is where all the confidential details will be stored along with the user’s email and password.
4. Implementing a working map into a page –the map allows users, that have signed up, to see how far the nearest player is. Or how far the nearest game is. This feature will not show the exact location of each player but, it will display each player’s ratings.
5. A project management tool (external) – this is a MUST when building the webpage. This tool allows users to create an organized checklist for each task. Without the checklist, it would be very chaotic since there are so many tasks to do in a short amount of time.

## **Legal, Social, Ethical and Professional Considerations**

The main issue with creating this project is to keep the users’ data private from hackers. The only information that should be shown is the key details such as the player’s name and rating. A way to prevent this from happening is to ask for the key information that is required to make this application work. Another risk is that there could be other similar applications out there. Therefore, this project has its unique features to avoid any copyright or plagiarism issues.

## **Background**

A lot of people have a passion for sports, but some may not get the opportunity to play it as often. PlayConnect gives them the chance to do so. It also allows people to go out and improve their social skills, which is certainly needed after the pandemic. As previously mentioned, this will keep people active in an affordable way. While being aware that other applications have the same idea, this application will be different due to additional features that other applications have not discovered.

Goals Soccer Centre and Find a Player are great examples of application where this idea has already been implemented on a mobile app. Goals Football Find a Player was created by Goals Soccer Centre. Goals Soccer Centre has multiple stadiums across the UK, where people can get together and book a pitch to play football on. On the mobile application, the benefits of play sports are explicitly shown. Another unique detail is how the app mentions user experiences and how it is relates to people that are interested in using the application. To differentiate PlayConnect, weakness in similar projects will be identified and improved.

## **Report overview**

Starting with looking at different technologies such as programming languages and database management tools since these will be the two main things to build this project. Alongside this, we will be looking in depth at analysing key aspects of the code used to design the project, along with other design and different methodologies used to help with the project.

# Technology Review

One of the main scopes in this project is that this web application will be database driven. The database that is most suitable for this project would be Firebase [1]. Firebase is a backend cloud computing service and is provided by Google. When it comes to database, there were a variety of database tools that could have been used such as, MongoDB or MySQL. However, Firebase is the best option since it is a database tool that is easy to manage, and set up also, it does not require any additional memory since everything will be stored via the cloud. In addition to this, the transition of integrating it with JavaScript is easily shown by the videos provided on YouTube [3].

Table 1. Comparing different database models

|  |  |  |  |
| --- | --- | --- | --- |
|  | Firebase | MySQL | MongoDB |
| Pros | -Built-in authentication, hosting, and cloud functions  -Easy to integrate with other Google services  -Real-time database with auto-syncing across different devices and platforms | -Flexible relational database management system  -Self-hosted for greater control if required  -Widely used, with more documented resources | -Supports nested data structure  -very flexible in the sense of it being cloud-based or self-hosted meaning greater control  -High performance with the larger the database |
| Cons | -Querying capabilities limited  - Data modelling isn’t as flexible  -Limited control and ownership due to it being a cloud service | -Requires more attention when setting up  -Less performant with the larger the database  -More maintenance as time goes on | -Constructing queries can be very complex  -No built-in authentication or hosting features  -Limited resources available |

Despite these comparisons, Firebase is the best choice for this project mainly for its cloud function and the real-time database with built-in authentication and hosting. Since the project is based on obtaining information from the database, this tool is the ideal one to use. The auto syncing function is a great feature as any data can be changed at any given time - assuring clients have all the up-to-date data on their devices. Another key thing is that with MySQL there are issues with saving data depending on the size of the database. Since Firebase is a cloud service, there will be no issues with storing data, if additional space is required it can be acquired with a fee.

One of the main focuses of this project is the interactive map. This feature is vital as this is where the users will be able to find their destination, as well as the route to get to that specified destination. The two different map tool options are Leaflet [24] and the Google API Maps. Leaflet is a JavaScript library that allows you to easily create an interactive map, mainly for web applications. It also uses Google Maps as a tile provider alongside other provider. On the other hand, Google Maps API is a set of tools and services provided by Google which allows them to integrate maps into their web or mobile applications.

Table 2. Comparing the pros and cons between Leaflet and Google Maps API

|  |  |  |
| --- | --- | --- |
|  | Leaflet | Google Maps API |
| Pros | -Open source  -Lightweight and fast  -Easy to customise and flexible  -Supports a variety of different map providers | -Easy to use and integrate into existing projects  -A variety of tools and services including geocoding and routing  - A lot of research done as it’s used by the majority of users |
| Cons | -Limited with the different things that can be customised  -Less research and documentation  -Certain features will need additional plugins or services | -Limited customization in comparison to Leaflet  -Less responsive than Leaflet  -Can be costly due to high traffic since more users are using it |

By comparing the two tools, Leaflet would be the more suitable option. The option of this being an open-source library means that it can be customised, and the maps can become suitable for the project requirements without being restricted which is what would’ve happened if we chose the Google Maps API. In addition to this, Leaflet has more options when it comes to different map providers which means the feeling of the map can also be suitable to the project requirements. Finally, since Leaflet is a lightweight tool, it will make it faster to use and more customisable despite the fact you may need to add additional plugins. All in all, Leaflet is the most suitable tool since it will be better to customise the features of the maps and align it to the requirements of the project.

However, Google Maps API is a great choice for users because of the resources Google will be able to supply when developing. Even though it may not be an open-source API, it will most definitely be developed more than Leaflet which suggests that there will be a significant number of features that are already built in. For example, Google Maps API would almost certainly have a real-life traffic data and transit information which will enhance the user experience with the live updates that will be provided. However, the main reason as to why Leaflet has the upper advantage is because of Google Maps API licensing and cost consideration, alongside the limitation a developer has with this tool as stated in Table 2.

# **Design and Methodology**

Figure 1. Entity Relationship Diagram for DatabaseA picture containing text, diagram, plan, technical drawing

Description automatically generated

As you can see in Figure 1, there are a variation of different entity tables within this entity relationship diagram. The scope of this database is to find a way in which each user can have a role within the project. As shown in the top left, the entity table has the Primary Key ‘AdminID’ – this is where the program will require an admin which will be able to create the specific event that will be published. But within this database design, it is shown that the relationship between the admin entity and the event entity is many to many. Anyone with an account should be able to create an event and be able to remove it, if necessary. Alongside this, the entity with the primary key ‘UserID’ allows users to reserve their spot if they see an event. However, within the RSVP entity, there are a few options that will also need to be passed by the admin. For example, some of the fields have included the rating field. If the admin doesn’t feel as if there is match between the rating of the user and the rating that is required within the event, then they would have the option of approving it or not. The reason for the admin to have access to this part of the project is to show the reliability of how accurate the game of the sport can be. Furthermore, within the RSVP entity, there is one-to-many relation since there can be many events, but each event can only have one RSVP per user since that is the focal point of this project.

Figure 2. Display the User Interface for the Home Page  
A screenshot of a computer

Description automatically generated with low confidence

The consistent pattern that will be shown throughout the whole project, as shown in Figure 2, is the button bar that is highlighted in blue on the top. Since this is a web application, we need to make sure from a visual point of view everything remains consistent throughout so that it gives the users a good experience where they can easily use the product without needing to work out even the simplest of things. Therefore, even keeping things such as the layout of the buttons at the top alongside a search bar will give the users access to whatever they may require from the project. Alongside from a users experience, from a developer’s point of view, keeping things consistent is much easier when it comes to things like maintenance because then a solution will be manageable for all the different pages at once.

Figure 3. User Interface sketch of wireframeA screenshot of a computer

Description automatically generated with low confidence

The sketch shown in Figure 3, represents the most important pages which is going to have the most functions and interaction from users. Again, from a design point of view, the intentions are to go for something simple with a few search bars and forms alongside a big sidebar that is appearing on the left of this wireframe. These forms are going to allow the user to enter their details, and this is where the relation towards the database will come into place. Once the form is complete, it will create an event which will not only publish this event into the sidebar, but only submit the details into the database.

This database will have a relation with both the connect and maps page; the connect page is where the user will be able to enter their details that will be sent to the database. Whereas, in the Maps page [4], it should do a similar thing in which there will be a sidebar just like in the previous page but alongside the bar, it will publish this event onto the maps using a pointer. This pointer should be clickable so that when the user clicks on the point, the event should appear and allow the user to take actions – whether it’s removing the event if it’s the publisher or if the user want to join the event.

Another page that is going to be implemented is the Login page [4]. This will allow users to create an event from one end, and another user can join the event from when they sign in. The page itself was going to be left out of the project completely, since it was not a feature that was a priority, but to give it a good user experience, a login page seemed very valid, in this case, to implement this despite this just being a prototype. With the log in page, it will give users an option to sign up where the users information will also be saved onto the firebase database.

The next steps will outline the different approach that will be taken – breaking down each page. In each break down, a discussion will be taking place in which tools and languages will be most suitable for this, along with looking at other alternatives too.

1. Home Page:

The home page aims to provide a visually appealing and user-friendly interface that engages users from the moment they land on the page. The design follows a clean and minimalist approach to ensure a seamless user experience. A difficult web application will make it less appealing for the clients to use.

The page layout is structured using HTML, allowing for logical organisation of content elements. Cascading Style Sheets (CSS) is employed to enhance the visual presentation and styling of the page. Custom CSS styles are applied to create a cohesive and aesthetically pleasing design. The use of different colours and specific spacing ensures a visually pleasing and consistent user interface.

The main goal of the home page is to facilitate easy navigation and provide users with a clear understanding of the application's purpose. The navigation bar at the top of the page contains buttons that link to other pages within the web application, enabling users to explore different sections. The layout and placement of the navigation elements are designed to be intuitive, ensuring that users can quickly and effortlessly navigate to their desired destinations.

2. Connect Page:

The connect page focuses on enabling users to create events and connect with other users. The design aims to provide a seamless and intuitive experience for event creation, capturing essential event details such as event name, location, date, time and the rating of the users.

The page layout consists of a form where users can input event information. The form fields are carefully designed and organized to ensure clarity and ease of use. JavaScript is utilised to handle form submission and interact with the form elements.

In addition to form handling, the Connect page leverages the power of Firebase, a tool developed by Google, for data storage and retrieval. When the user submits the form, JavaScript captures the entered data and utilises Firebase's Firestore database to store the event information securely. Firebase's Firestore is a NoSQL database that provides real-time synchronization and offline support. It seamlessly integrates with JavaScript applications, allowing for efficient and scalable data storage. The Connect page leverages Firebase's JavaScript SDK to establish a connection with the Firestore database and perform data operations. By utilizing Firebase, the Connect page ensures that event information entered by users is persisted in a reliable and scalable database. This allows for seamless retrieval and management of events throughout the application. Users can create events and have confidence that their data is securely stored and easy to access. The integration of Firebase with the Connect page involves configuring Firebase project settings and initializing the Firebase SDK in the JavaScript code. This enables the application to communicate with the Firestore database and perform operations such as adding new events, updating existing events, and retrieving event data for display.

Overall, the Connect page demonstrates the use of modern web technologies such as HTML, CSS, JavaScript, and Firebase to showcase a user-friendly event creation experience. The seamless integration of form handling and Firebase's Firestore database ensures efficient data storage and retrieval, enhancing the overall functionality and reliability of the application.

3. Maps Page:

The maps page aims to provide a visually appealing and interactive representation of user profiles and events on a map. The design utilizes the Leaflet library, which offers a comprehensive set of tools to develop an interactive map. The page layout will consist of a map container where the Leaflet map will be place. The map is centred on a specific location and allows users to explore different areas by panning and zooming, not trying to take away the basic functions of what a map normally does.

To display user profiles and events on the map, JavaScript is employed to iterate over an array of profile and event data. For each profile or event, a marker is created at the corresponding latitude and longitude coordinates. When the user clicks on a marker, a pop-up appears, providing additional information such as the user's name, address, and preferred sports. Alongside this, each user or event will be displayed in a sidebar so that all users can get a simpler view of the events.

To expand on the sidebar, an interactive button should be adapted to each event using tools within the JavaScript programming language. This is so that the user can easily access the event, where it can expand the details if required. Moreover, a feature will also be implemented where in the sidebar, a route function will be implemented so that when a user enters two locations, a route should be drawn on the map with directions. Leaflet being an open-source tool, this should be accessible to implement as it serves a powerful tool for map rendering, marker creations, and map-related functionality.

In conclusion, the approach taken in designing these pages involves a combination of HTML, CSS, and JavaScript, along with external libraries and APIs, to create a visually appealing, interactive, and user-friendly web application. The use of clean and minimalist designs, appropriate colour schemes, and intuitive navigation elements ensures a seamless user experience. The integration of Leaflet and geocoding API enhances the functionality of the connect and maps pages, enabling event creation and visualisation on a map.

Alternative Approaches:

Maps Page:

Instead of using Leaflet, another popular option for integrating maps into web applications is the Google Maps API. It provides a comprehensive set of features and extensive documentation for map rendering and interactive functionality. The Google Maps API also offers additional services such as geocoding and directions, which could be useful for enhancing the user experience. As stated in Table 2, Google Maps can be shown as a benefit since more users have explored and researched using Google Maps API. But since the main factors is that it is not an open-source tool, trying to obtain the goals of this project outcome, it may not be as feasible as Leaflet.

Connect Page:

Since Firebase provides a suitable solution for storing data and real-time synchronization, other backend technologies could have been employed. For example, SQL database such as MySQL or PostgreSQL, being the more obvious choice, could have been used to store data. These databases provide powerful querying capabilities and offer strong data consistency and integrity. This is also due to the high number of resources that can be obtained since these are backend tools that have been available for many years.

Additionally, backend frameworks like Node.js with Express could have been implemented to handle form submissions and perform server-side processing. This approach would involve setting up a server, defining routes, and utilizing an ORM (Object-Relational Mapping) library to interact with the database.

Conclusion:

In conclusion, while the chosen approaches of using Leaflet and Firebase for the Maps and Connect pages respectively are suitable for their intended purposes, alternative methods and tools could have been employed to achieve similar or enhanced functionality. The choice of tools and technologies ultimately depends on various factors such as the project requirements, scalability needs, available resources, and personal familiarity. Exploring alternative approaches provides valuable insights into different technologies and allows for making informed decisions based on the specific needs of the project. By considering alternative methods, developers can expand their knowledge and skill set, enabling them to select the most appropriate tools for future projects. Regardless of the chosen approach, the primary goal remains the same: creating a compelling and user-friendly web application that fulfils its intended purpose effectively and efficiently.

In addition to the design, the actual implementation of the project will be done in Visual Studio Code [30]. This is a software that has many different resources since many developers tend to use it regularly. This text editor has a variety of different features including extensions that can be downloaded whenever required and the fact that it has a wide range of users that use this software, support would be much easier to get from other developers. The other benefits of using Visual Studio Code is that the GitHub repository can be cloned through this software which would give easy access to push and pull files, along with data from and to the GitHub repository, making the flow of working on this project and sharing this with other supervisors much easier and not to mention that it has a debugger which is very useful when it comes to typing large pieces of code.

As mentioned previously, this project will be using Google Firebase for the database but alongside this, the programming languages the intensions are to use JavaScript, HTML and CSS. HTML and CSS [8] are the most obvious languages to use when it comes to creating the structure, adding content in the pages and designing the layout of the page so this is why these two languages were best fit. JavaScript is a language that can create interactive effects on webpages along with the interpreted language being used for both the client-side and server-side. There are other languages like JavaScript such as Django which is a high-level Python web framework which is also as effective since it also provides a set of tools and libraries from building web applications, but again is a language that is more common than anything else so in terms of the resources. All in all, these three languages work well together and is usually what developers tend to use when coming to creating a web application.

To keep record of all the steps taken with this project, the project management tool that will be used is MeisterTask to break down this process into steps so time can be managed wisely with this project. Alongside this, adding different steps on a excel sheet which is being published on GitHub, which can be found in the Appendices of this report, for supervisors to see if required is a good additional way to check different objectives off throughout this project. Again, other project management tool such as Monday.com could have also been used but first, this wasn’t free whereas MeisterTask was, and it had all the tools needed such as creating timescales and being able to check them off when the task was completed – this was more than enough as to having this project management tool.

To conclude, this approach is manageable within the timescale that has been given. Since this project is only a protocol, focusing on making the front-end design to the potential should not be the main priority. The most important thing should be to make the main function working, which is the database communicating between the connect and maps page – creating the event and then making sure that the event is placed on the map within a pointer is the main scope, alongside allowing the users to join the event and notifying the admin of the event. If there is more time than initially anticipated, enhancing the look of the web application more so it can have that user-friendly feel to it can be an option to look in to.

# **Implementation**

The first steps that are taken was on how to get a hold of running the program on a server, locally. To do this, installing a http server would be required and can easily be done through ‘npm’ which is a package manager from JavaScript applications – it is very useful when it comes to installing libraries and so forth. In this instance, we would need to do the following which will be shown in the figure below:

Figure 4. Installing the http-serverA screen shot of a computer program

Description automatically generated with low confidence  
As shown in Figure 4, the packages such as the http-server was successfully installed. But then to start and initiate this package, a command was added into the ‘package.json’ within the ‘script’ section which would allow the server to start. The command that was written was the following:

Figure 5. Displaying the command added to the package.json fileA picture containing text, screenshot, font, line

Description automatically generated

The command that was written was in line 8, shown in Figure 5. Firstly, it is defining the script called “start” which is then followed by the command “npx http-server -a localhost -p 8000 -c-1”. This line is launching the HTTP server using the ‘http-server’ package that was previously initiated. The localhost specifies that the server should bind to the localhost interface, along with ‘p 8000’ is the port number in which the server should listen to. And finally, the c-1 command ensures that the server is always providing the latest version of the file.

Figure 6. Terminal command starting the server   
A screen shot of a computer program

Description automatically generated with low confidence

As you can see, the way to initialise the server is typing ‘npm start’ in the terminal on visual studio code and that will show the available URL link at the bottom which will allow us to either click on it or go onto an explorer URL search bar and type the following down. And with these easy steps, the project has successfully started on a server. The main reason as to why this application must be written on a server because the database tool, firebase, is something that is being run online since it is a cloud-based database and therefore, if it was being ran locally without a server or the correct ports, no packaging will be either sent or retrieved to/from the firebase database.

Previously mentioned before, the programming languages HTML and CSS will be used for the front-end design. The main design used for most of this project is displayed in Figure 7:

Figure 7. User interface displaying the Home Page  
A screenshot of a computer

Description automatically generated

A brief explanation of this design was to keep it simple. Four buttons have been created with a simple effect where it allows you to hover over the button you choose to and as you do it, a white line will appear at the bottom of the button– this animation was done in CSS using the hover tool. The idea was to go for a theme that was appealing in the sense of easier for the users to see and interact with; not showing too many colours as at times it could confuse and be too much for the user.

Initialising the Firebase Database

The initialisation of the Firebase database inside of the database can be found on the Firebase website. It consists of a variety of imports that can be added throughout the implementation stage. On the website, once the project has been created, it will give the users a pre-made configuration object that will just need to be copy and pasted inside of the JavaScript code and this will be called using the ‘firebase.initializeApp()’ method. Once this is done, you will be ready to use Firebase services such as Authentication, Realtime Database etc just by referencing the ‘firebase’ global object.

Starting up the Web Application

To start off with, the user cannot use any of the pages unless they have signed in. This was done so that the users who do create an event, can keep their privacy from the public eye and keep it strictly for the other users within the application – it can also be beneficial in terms of the user’s safety and wellbeing. The figure below will show the JavaScript function called ‘signUp()’ which is constructed for the user to create an account to access the web application.

Figure 8. JavaScript asynchronous function named ‘signUp()’  
A picture containing text, screenshot, software, multimedia software

Description automatically generated

This function employs asynchronous programming techniques to manage operations that involve waiting for responses from external resources which is shown in line 114, with ‘async’ being the word indicating this. Within the function, two variables have been declared which will have the values that have been inputted by the user in the HTML form – this information will be gathered by using ‘document.querySelector()’, and stored in the assigned variables. The ’try’ block is used to encapsulate the execution of asynchronous operations that potentially could throw errors. The function ‘createUserWithEmailAndPassword()’ is called and has the parameters ‘auth’ which is the imported authentication service, and the two variables that are inputted by the user in the sign up form. This function helps create a new user with the provided details and if there are any errors which occur during this process, the ‘catch’ will block this error which will also be logged into the console during this process. If there is no error, then the data will be successfully added to the database.

Alongside this, there is another function called ‘updateProfile()’ which passes the parameters of the display name (user input inside the form) and the ‘auth.currentUser’ object. With this function, it updates the user’s profile information and just like with the other function, it will log any errors that this part of the application may face. If all of this is successful, then the user will be redirected to the home page, meaning that the application is ready for the user to use.

For reoccurring users that have already signed up, they will just need to direct themselves to the index page. On this page, it will contain the login form which was created in HTML. The reason this is chosen as the index page for users is because of keeping other user’s information private and their safety as this can be shown as a hazard especially when time and location is being shown on the event profile. In the figure below, it will show the JavaScript function logIn() which is for the login form.

Figure 9. JavaScript function for the log in page  


This function follows the same principles as the ‘signUp()’ function where it declares to from a user input, but this time the data will be recorded from the login form. The signInWithEmailAndPassword() function contains an ‘accessToken’ which is imported from the Firebase library - this is assigned to the user credentials. When the user types in their credentials and if it meets the requirements, then it will have access to all functions within the application. This would be successful within the function when it is called and is passed through the ‘auth’ object which is also a part of the Firebase Authentication – it will scan through this section on the database and see if the credentials are there (example of test data shown in Figure 10). At first, when you enter the index page (log in form), all the buttons displayed at the top, other than the log in button, will be disabled. This disabled method is displayed in the HTML script which then will be removed, as declared in this JavaScript function when the requirements have been met. A catch is also used in this function and will also display the error message or code, if the credentials are wrong.

Figure 10. Displaying the users in Firebase Database A screenshot of a computer

Description automatically generated with medium confidence

In Figure 11, it shows the input of two post codes in the respective text box and once the correct formatting on post codes have been entered, there will be a chance for the user to press the ‘calculate route’ button. Once this is done, there will then appear two pointers showing the start point and end destination, followed by a route in the colour red and the directions will also be declared on the right of the screen. Alongside this, a sidebar has been implemented on the left of the application where all the events are being displayed. These events are created in the connect page which then gets sent to the database and then read and displayed onto the maps page. Each event is inside of their container – these events are clickable and once the user clicks the event, they will be able to see a change where the location will be published onto the ‘End Postcode’ text box. This will save them time as they would not need to type the location themselves.

Figure 11. Displaying a route in the maps pageA screenshot of a map

Description automatically generated

Figure 12. JavaScript function for the maps pageA screen shot of a computer code

Description automatically generated with low confidence

Figure 12 shows the JavaScript function named ‘maps()’. This function takes the steps into how the program will read the information from the Firebase database and be placed on the sidebar which was implemented in the HTML script. Overall, this function involves user authentication and database operations. The function itself starts with an ‘if statement’ which at the beginning, tells the program to obtain the ‘getDatabase()’ object which is referring to the Firebase Realtime Database data (shown in Figure 16). Once this is obtained, the program will make a database query using the ‘get’ function and retrieve all the data that is under the ‘/user/ path – if the data is return this will be classed as a snapshot object, whereby if the object exists, the code proceeds to iterate over the values of the snapshot using the “Object.values” function.

For each value, representing a user’s event, the code retrieves the sidebar element using the querySelector methods and appends HTML markup to it containing all the different attributes and information entered by the user in the form when creating the event.

Furthermore, there is an ‘onclick’ attribute appended in the HTML markup which will trigger the ‘handleEventClick()’ function and essentially place the location from the object, into the ‘End post code’ text box which can be seen in Figure 11, in the sidebar. If any errors occur during the database operations, a message will be logged into the console. The way in which the routing of the two locations work will be shown in the JavaScript function ‘caluclateRoute()’ which is displayed in Figure 13.

Figure 13. JavaScript function for interactive click  
A picture containing text, screenshot, font

Description automatically generated

Figure 14. JavaScript code for routing two locations  
A screen shot of a computer program

Description automatically generated with low confidence

The calculateRoute() function starts with getting user input using ‘document.getElementById()’ which are then stored into new variables. To start with, it utilizes the Leaflet’s Geocoder by which the start and end values from the user’s input will be converted into geographic coordinates. Once this is completed, the function retrieves the centre coordinates of the start and end locations from the result and then it returns the value by creating a routing control – this will outline the route using the coordinates that would’ve been converted. After adding the route to the map, the function listens for the ‘routesfound’ event, which allows it to log and collect the route that’s been calculates – provides useful information to analyse and debug. This is then followed by a marker which calculates the route and makes sure that the marker is matching the coordinates in each position so that the route is displayed correctly for the user.

Figure 15. JavaScript function for the connect pageA screen shot of a computer program

Description automatically generated with low confidence

In Figure 15, it shows the writeUserData() function which starts of by declaring the appropriate variables required for the user to create an event for the application - This is done by using the ‘document.querySelector()’ method which will take the data that the user inputted. Then the program retrieves the data for further use in the script. The getDatabase() function is called from the imports and involves using the Firebase Realtime Database. It initialises a reference to the database and uses the ‘push()’ function to create an entry which will be stored under the ‘/users’ node which will have its unique identifier (‘userId’). The entry data is declared inside of the push() function containing username, email, location, sport and players number. With this function, a new child node will be initialised every time a new event is created. At the end of the function, once the ‘push()’ has been initialised, the program will automatically refresh using the pre built-in function ‘reload()’ and this will show all the different events on the sidebar, as shown in Figure 16.

Figure 16. User Interface of the Connect page  
A screenshot of a computer

Description automatically generated with medium confidence

Figure 16 is showing the connect page, where users can create an event and see the events they have created within the sidebar. This will strictly show what the user has created but everyone’s events will be shown in the maps page. This is so that the users can be notified and see if there are any updates on whether they have enough players for the event to proceed. Alongside this, you can see a basic format to the event form, which is centred in the middle of the screen, just so that this is user friendly and easy to manage by the users.

Figure 17. Firebase database with the created events  
A screenshot of a computer

Description automatically generated with medium confidence

Figure 17 visually represent the structure of the database and displayed the stored information which was inputted by the user. The node named ‘users’ has got its child nodes which is the ‘event id’ and inside of that node, will contain the different data that has been stored, as shown in the Figure. These steps will be followed continuously and will always fresh once the user makes another input, or even if a new user has been created.

## **Evaluation**

Overall, the project was meticulously planned and executed. Although this was only a prototype, the web application has many useful qualities. The main quality was that it solved issues that were discussed at the beginning of this report. Additionally, the project applied many different tools such as data being sent to databases and data being retrieved. This was explored with JavaScript and Firebase. Compared to other data management tools, Firebase hasn’t been around for that long. So, there weren’t enough resources to learn how to use the tool efficiently.  Despite this minor setback, the aim (stated in the objectives part of this report) of merging code and databases together was successfully achieved.

However, the interactive map did not live up to its expectations. The project was mostly reliant on the interactive map. One idea that was not incorporated, due to time restrictions, was the map being fully interactive. This would have eliminated the need for a sidebar. Instead of having the events on the sidebar, there could have been a clickable box with pointers for the users to see the events on the maps. Despite being able to create a routing between two different locations, more information could’ve been amalgamated with the interactive map.

In conclusion, there are aspects of the project that performed well and other aspects that need improvement. If the opportunity to redo the project was authorized, there would be changes made. One positive was the ability to demonstrate an interaction from back-end and front-end; it managed to complete a lot of objectives that were first initiated in the plan. But the downside to this was the fact that a completely developed and interactive map wasn’t demonstrated in this project.

## **Related Work**

Applications where you find a player in the area have gained significant popularity in recent years. These applications offer a convenient way for individuals to connect with like-minded players and participate in their favourite sports. Several existing applications, such as Goals Soccer Centre [6], Find A Player [22] and other familiar apps, may have addressed the player-finding challenge. However, there is still ample opportunity to explore new avenues and tackle unaddressed issues in this domain. This technology review aims to examine the current landscape of player finding applications, identify potential areas for improvement, and propose innovative approaches to enhance the user’s experience.

Existing Player Finding Applications:

Goals Soccer Centre and other competitors have showcased the value of providing a platform where individuals can search for players, join events, and book facilities for organised games. These applications often include features like player profiles, skill level filtering and many more features. They have proven to be effective in connecting players and promoting community engagement.

Differences between PlayConnect and other existing applications:

Although PlayConnect and other player finding applications solve the real-time issue for sports enthusiasts that want to physically play a sport. There are some differences when it comes to features showcased in PlayConnect. One of the main features shown is the route calculator. One thing discovered when it comes to users is needed to know directions. This is an issue which is yet to be solved with the similar applications and for which, was a great function to implement into this application. The user can find the location and insert it into the search bar as shown in Figure 11. Followed by the user entering their own location which will result in generating a route. This is a good function to have because it can be very difficult going somewhere the user is not familiar with.

Identifying Unaddressed Issues:

While existing player finding applications have made significant steps that enhance the user’s experience and solve their issues, there are some areas to consider for development:

1. Improved Matchmaking Algorithms: Current applications rely on basic filtering mechanisms, such as, skill level and location, to match players. However, more sophisticated matchmaking algorithms that consider additional factors like playing style, preferred position, and availability could lead to more accurate and compatible player pairings.

2. Real-Time Location Tracking: Integrating real-time GPS tracking capabilities within player finding applications could enable users to locate nearby players, facilitate spontaneous matches, and reduce the time spent on coordination. Although it can be challenging since, it could alarm some privacy and security concerns.

3. Social Networking Integration: Connecting users is a great way to enhance this application. For example, implementing a chat box would be a great way for people to keep in touch.

While existing player-finding applications have paved the way for connecting sports enthusiasts, there are many different features that can be explored and enhanced. By focusing on areas such as matchmaking algorithms, real-time location tracking and many other features, developers can enhance the user experience and revolutionize the way players connect and participate in sports. By constantly innovating and pushing the boundaries, these applications can play a crucial role in promoting sports, facilitating new connections, and enhancing the experience of playing sports on a global platform.

1. **Conclusion**

Overall, the project managed to meet some of its goals but, it did not meet some of the outlined requirements. The project had many features that demonstrate a good application such as, the integration of Leaflet for an interactive and visually appealing representation of user profiles and locations. The Map function was meant to be the main feature of this project. However, the Map function only managed to fulfil some of its objectives., For example, a feature that was supposed to be included was the incorporation of all user profiles on the map rather than the sidebar. The complexity of this feature required more time and resources, which were limited. On the contrary, creating a route on the maps, using the user’s data, is something that was accomplished. As mentioned previously, this was something that has not been showcased in similar applications. So, this advantageous feature makes PlayConnect different to its competitors.

Additionally, all the goals that were related to the use of database were achieved. As mentioned in Table 1, Firebase is a tool that does not have a lot of resources in comparison to other database management tools. Despite it being a challenge Firebase was successfully integrated into the application. The application can exchange data from databases and display its data for the users, successfully.

At the beginning of the project, the log in page was not a priority, as time went on, it became more necessary. A user must create a profile in order to interact with other users. Without any users the purpose of the project would not be met. Furthermore, creating a disabled feature where the user would need to enter their user credentials before being able to use the application, shows to be a necessary too since it keeps privacy for other user’s data, as well as their own.

Lastly, a feature that should have been implemented was getting user feedback. Since this is a working prototype and not the finalised project, it would have been good to include a feedback and review form, to gather user input for future improvements. Despite creating a form on Google Forms [7], it is yet to be utilised and integrated into the application. In addition, user testing received limited attention, with only a few individuals providing feedback on the application. A more extensive user testing phase would have allowed for a broader and valuable insight.

In conclusion, this application showcased a diverse range of features, utilising various tools. The use of different tools and technologies symbolises the versatility of the project along with the exploration of multiple application developments. The project incorporated a wide array of tools, such as the use of multiple programming languages (HTML, CSS AND JavaScript), Leaflet of map rendering, Firebase for data storage, Google Forms for gathering user feedback. This displays the ability of integrating third-party services effectively and utilising their individual functions to enhance the performance of the overall application. This versatility contributes to the applications overall quality in delivering a robust and dynamic software solution.

## **Reflection**

From a personal point of view, I believe that I can take a lot of positive things from this project. What started off as good progress with this project, meeting deadlines and planning early, suddenly came to a halt when I lost my father halfway through this academic year. This sudden loss impacted me massively as it was very difficult to find the motivation to do work. I had to take a temporary leave from my studies to recuperate and adjust to a new life without my father.  As I healed and gradually submerged myself with my studies, I managed to find the determination to complete this project to the best of my abilities. Since I knew I was on a time limit and had to complete missed assignments, my focal point was to outline the key functions within the project. So, to help me get through this, I focused on what mattered the most and tried to unearth the necessary features for this project.

Aside from the personal issues, one crucial hurdle that I had to overcome was that I lacked experience with the different tools and programming languages. Coming into this project, my nescience of JavaScript, HTML, CSS, and Firebase (data management tool) was overwhelming. Out of all languages that were included in this project HTML and CSS were the easiest to understand due to innumerable resources found online. I would have said the same thing about JavaScript but, it was challenging to get JavaScript and Firebase to communicate with each other. The struggle to comprehend JavaScript added an extra layer of complexity to my project, requiring me to invest significant time and effort into self-study and seek guidance from knowledgeable resources.

The process of learning JavaScript was difficult. There were moments where I encountered errors and struggled to grasp certain programming concepts. However, I remained determined and persevered through these challenges. I sought out online tutorials, documentation, and coding forums that provided valuable insights and practical examples.

In conclusion, my computer science project was a journey marked by numerous challenges, among them my limited knowledge of JavaScript. Overcoming these struggles required dedication, perseverance, and a commitment to continuous learning. Through hours of self-study, seeking assistance from knowledgeable resources, and hands-on practice. I was able to navigate the complexities of JavaScript and successfully implement it in my project. The experience not only deepened my technical skills but also fostered personal growth, resilience, and an unwavering determination to conquer future obstacles in my computer science journey.

## **Future Work**

The aim of the project was to develop a fully, functioning web application. Since the current state of the PlayConnect isa prototype, so there is a need for future work that expands and refines the project to make it more accessible and inclusive for the expanded demographic. For example, PlayConnect can be accessible for the older and disabled population in the future. To implement this change PlayConnect will include sports such as, power hockey, walking soccer, etc. If some individuals are struggling with working out the application, we will make it more accessible to them. To accomplish this task, there will be more user research and integrating feedback from users. The research will consist of the challenges they have faced in the sporting community, or even a case of where it may be a bit difficult for them to use computer systems. Gathering this feedback will allow me to gain an insight from the focused demographic. Which allows me to expand my toolbox so I can make the target users more comfortable with the application. For example, if a user that has a disability which affects, they can have an assistant option which can help them access different features. There are tools that are already implemented with other different applications whether it’s on a mobile phone or a desktop, so this would be something I would look closely towards in future works. These technological advancements should be explored and integrated into the project to empower users with diverse abilities.

Alongside the necessary tools to help the different demographics, there are different feature that can be adjusted. From a visual aspect side of things, features such as font size or the colour of words can be a great way to make the project more user friendly. This allows the users to have the freedom to explore the application without that hindrance. Disabilities such as colour blindness can make it difficult for some people to visualise the words. Simple adaptations like this could make this experience better for users, even if it is just one.

Another aspect I would change, is the layering of each page. I had fun exploring different techniques and layouts with this web application. However, the animation with the buttons could have been better. In my opinion, the hovering effect is not enough. This would have made the project seem more professional and more visually appealing to the users. However, since this is a prototype, there is a lot of room for improvement.

My focus is to improve the privacy of the users’ data’s. Handling a user’s information in a safe and secure manner is crucial. Something I must improve for future works for the application to be lawful and ethical. Instead of asking for the user’s postal code, PlayConnect can ask the user for a region that they are interested in playing in.

To conclude, there are a lot of different considerations to be considered for future works. As stated before, since this is a prototype, a lot of work can be done to enhance the inclusivity and accessibility. Mainly trying to focus on the disabled and the older population since similar app have not achieved this and it can help them stay active as well. I am confident that this can be done successfully with the help of data research, designing and receiving feedback from the sports enthusiasts of these two demographics. This would help with allowing a wider range of audience to be using this application.

# **References**

[1] Google. (2021). Firebase website. Retrieved from <https://firebase.google.com>

[2] Successive Digital (n.d.). Best Database for Web Applications. Retrieved from <https://successive.tech/blog/best-database-for-web-applications/>

[3] YouTube. (2022). Getting started with Firebase for the web. Retrieved from <https://www.youtube.com/watch?v=rQvOAnNvcNQ>

[4] GitHub. (2023). Wireframe diagrams. Retrieved from <https://github.com/mseahra/PlayConnect/blob/main/Wireframes%20-%20Website%20blockframe%20example.jpeg>

[5] ChatGPT. (2023) Retrieved from <https://chat.openai.com>

[6] Goals Soccer Centre. (2023). Retrieved from <https://www.goalsfootball.co.uk/>

[7] Google Form - <https://forms.gle/5CnxBCgy3WHTuxGH7>

[8] W3Schools. (n.d.). HTML Tutorial. Retrieved from <https://www.w3schools.com/html/>

[9] Codecademy. (n.d.). Learn HTML. Retrieved from <https://www.codecademy.com/learn/learn-html>

[10] FreeCodeCamp. (n.d.). Learn to Code — For Free. Retrieved from <https://www.freecodecamp.org/>

[11] CSS Tricks. (n.d.). CSS-Tricks. Retrieved from <https://css-tricks.com/>

[12] Google. (2021). Delete data in Firebase Realtime Database. Retrieved from <https://firebase.google.com/docs/database/android/read-and-write>

[13] Google. (2021). Update data in Firebase Realtime Database. Retrieved from <https://firebase.google.com/docs/database/android/read-and-write>

[14] Google. (2021). Write data to Firebase Realtime Database. Retrieved from <https://firebase.google.com/docs/database/android/read-and-write>

[15] Google. (2021). Read and write data in Firebase Realtime Database. Retrieved from <https://firebase.google.com/docs/database/android/read-and-write>

[16] Random Nerd Tutorials. (2021). ESP32 with Firebase – Creating a Web Application. Retrieved from <https://randomnerdtutorials.com/esp32-firebase-web-app/#install-software>

[17] Webdeasy. (2023). 65 cool CSS Buttons – with Animations. Retrieved from <https://webdeasy.de/en/top-css-buttons-en/>

[18] Stack Overflow. (2021). TypeError: undefined is not an object. Retrieved from <https://stackoverflow.com/questions/66879378/typeerror-undefined-is-not-an-object-evaluating-app-firestore>

[19] Stack Overflow. (2021). Failed to resolve module specifier. Retrieved from <https://stackoverflow.com/questions/69230383/failed-to-resolve-module-specifier-firebase-app>

[20] Stack Overflow. (2012). JavaScript code to stop form submission. Retrieved from <https://stackoverflow.com/questions/8664486/javascript-code-to-stop-form-submission>

[21] Stack Overflow. (2011). Styling Form with Label above inputs. Retrieved from <https://stackoverflow.com/questions/6046110/styling-form-with-label-above-inputs>

[22] FindAPlayer. (2022) Retrieved from <https://www.findaplayer.com/>

[23] Routing. (2022). Retrieved from <https://developer.trimblemaps.com/maps-sdk/guide/routing/>

[24] Leaflet: Open-Source JavaScript Library for Interactive Maps (2022). Retrieved from <https://leafletjs.com/>

[25] OpenStreetMap: Collaboratively Created Free Map Data (2022). Retrieved from <https://www.openstreetmap.org/>

[26] NPM: Package Manager for JavaScript (2022). Retrieved from <https://www.npmjs.com/>

[27] W3Schools. (2022). How TO – Responsive Sidebar. Retrieved from <https://www.w3schools.com/howto/howto_css_sidebar_responsive.asp>

[28] Stack Overflow. (2021). How do I set the displayName of Firebase user. Retrieved from <https://stackoverflow.com/questions/40389946/how-do-i-set-the-displayname-of-firebase-user>

[29] Smashing Magazine: Web Design and Development Articles. (2023). Retrieved from <https://www.smashingmagazine.com/>

[30] Visual Studio Code: Source Code Editor. (2023). Retrieved from <https://code.visualstudio.com/>

# **Appendices**

GitHub – Repository - <https://github.com/mseahra/PlayConnect>

MeisterTask Tool - <https://www.meistertask.com/app/project/b0Z3q3H7/fyp>