APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 5 No. 2 (2024) 388-406
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Design and Development of a Freelance Sport Coach Mobile Application

Lieu Yee Zhe[1], Norfaradilla Wahid[2]*

[1,2] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: faradila@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2024.05.02.021

Article Info

Received: 13 June 2024
Accepted: 28 September 2024
Available online: 15 December 2024

Keywords

Freelance Coach, Coach, Trainee, Sport, Scheduling

Abstract

The growing interest in sports has led to a surge in demand for sports coaches, resulting in popular mobile applications for finding virtual courses. However, there are limited application for booking physical courses. This application fills the gap by offering a physical coach and trainee matching platform specifically for Malaysian residents. It accelerates the matching process, allowing users to find coaches based on preferences or specialties. Features like chat, online payments, and payment history enhance the user experience. Developed using Java via Android Studio and utilizing Firebase for data management, the project employs the Prototype modeling approach. Rigid functionality tests during the testing phase ensured all requirements were met, effectively addressing challenges faced by sport coaches and trainees in Malaysia. The application streamlines the process with an intuitive interface and robust scheduling capabilities. However, it faces limitations, including a lack of flexibility in managing sport categories and locations, and no refund policy implementation.

1. Introduction

Sports coaches prioritize mental, emotional and physical development, which is essential for the development of trainees at all levels [1]. Trainees receive guidance and direction from coaches who develop resilience and discipline [2]. They maintain the health of the trainees and create the conditions for them to perform at his or her optimal level. Sports coaches play a vital role in helping trainees achieve their maximum potential and overall development. Sports coaches have a significant impact on both the individual athlete and the wider sporting community. However, there are several issues that highlight the need for the proposed application, where the lack of a centralized platform prevents efficient matching between coaches and trainees. Coaches and trainees are matched with trainees by posting on social media such as Facebook, but there is no limit to the number of posts within the platform, resulting in the need for repetitive posting and thus slowing down the matching process. In addition, the fixed schedules of coaches limit the flexibility in choices because trainees are not able to make choices based on their personal needs such as time and preferences.

The Freelance Sport Coach mobile application aims to provide a safe and convenient platform for Malaysian residents. It focuses on providing access to a wide range of popular sports, making it easier for trainees to find qualified coaches, and at the same time enabling coaches to post recruiting postings, thereby increasing their customer pool. While there are countless mobile applications for them, the Freelance Coach mobile application is unique in its focus on sports. It provides opportunities for sports enthusiasts, whether they want to participate in sports or learn more about them. Coaches can also find trainees more easily. In short, the Freelance Coach mobile application fulfils a strong need for a dedicated platform that connects coaches and

trainees in the field of sports and promotes safe, convenient and diverse participation in sports for Malaysian residents.

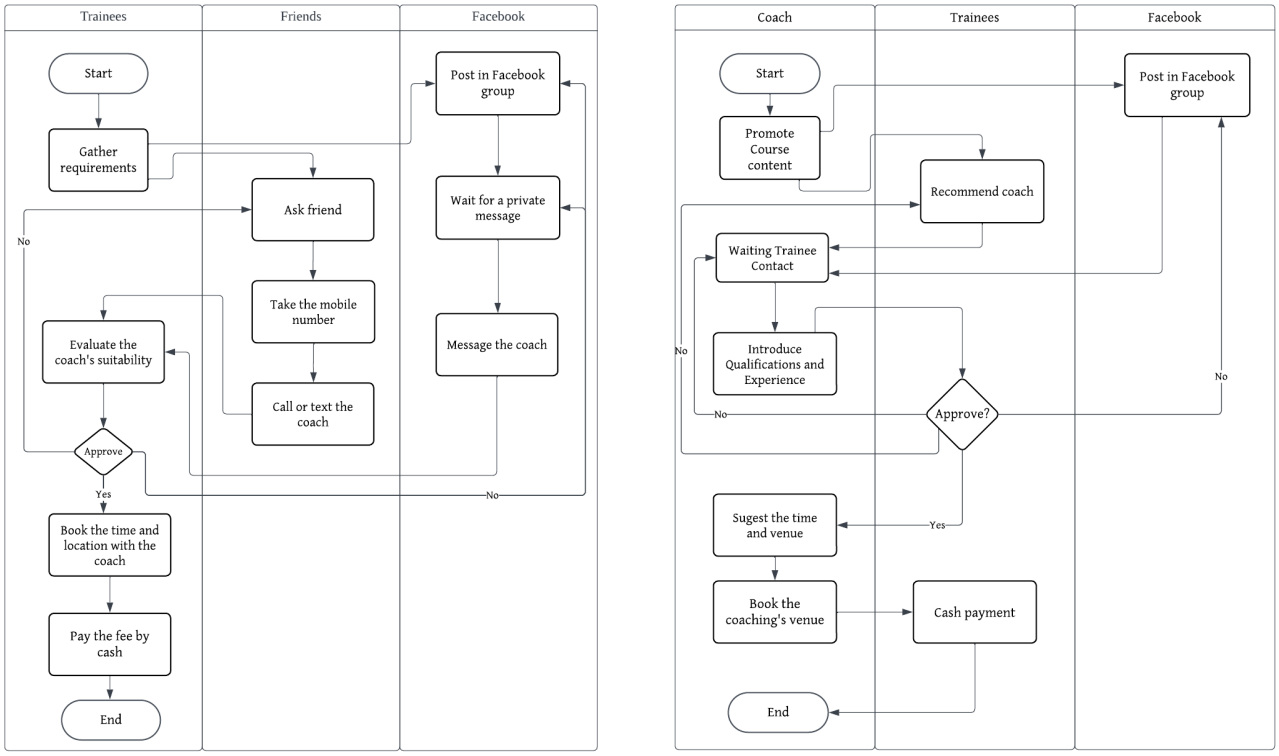
This paper is organized into 4 sections. Section 1 discusses the introduction of the proposed application followed by Section 2 which will present the related work which will explain the existing methodology and the comparison of the proposed application with the existing applications. Section 3 will discuss the methodology used, i.e., the prototype model and its corresponding phases, and the last section will discuss the conclusions of the project, briefly summarizing the archived work done, difficulties faced and future plans.

2. Related Work

The coaching industry has experienced significant growth since to cater to different needs. There are significant benefits of employing a dedicated coach who brings expertise to meet the needs and goals of a specific area. In the second quarter of 2023, the number of sports and fitness practitioners in the UK is expected to exceed 161,500 compared to 155,900 in the first quarter of 2023 [3]. This clearly demonstrates the growth in demand for coaching. Sports coaches include professional and non-professional coaches, each with different qualifications, experience and commitments. Professional coaches have recognized credentials and formal training and focus on the results, while non-professional coaches rely on personal passion or experience and prioritize the process. In addition, the price of sports courses varies, based on experience, qualifications, etc. The demand for coaching has increased due to increased awareness of the benefits of sport, the need for stress relief, and the realization that coaching is a valuable investment in personal development and boosts self-confidence through sports development.

2.1 Existing Approach

There are two existing ways for coaches to connect with trainees which are trainees looking for coaches and coaches looking for trainees. In the trainee-to-coach method, individuals look for coaches through friend referrals [4] or Facebook group postings. They contact potential coaches, check whether the qualification meets the demand and the price, time and place of the course, and confirm the course if they are satisfied. If they are not satisfied, they can seek other referrals, other PM or re-post. after confirming the coach, they can practice and pay by cash.



(a) **(b)**

Fig. 1 *Business Flow Diagram (a)Finding and Hiring Coach; (b) Coach finding Trainees*

Figure 1(a) illustrates this business process diagram that describes the ways in which trainees can find, hire and receive coaching services through referral from friends or online platforms such as Facebook. During the coach's recruiting process, the coach actively promotes his services through word-of-mouth or by posting advertisements in relevant Facebook groups. After promotion, the coach waits for potential students to express interest through referrals or direct contact. The coach then introduces his qualifications and coaching experience, allowing the student to evaluate his suitability. If the trainee agrees, coach coordinates the time and location of the session, and the chosen venue is then booked. At the end of the coaching session, the participant pays the fee in cash. In case of no participant expresses interest, the coach continues to wait for potential trainees. Figure 1(b) summarizes the business process diagram for coach finding trainees.

2.2 Existing System Comparison

This section describes the features offered by the similar applications. Three existing mobile applications are selected, i.e., Freelancer [5], Coach You [6], and Upwork [7]. At the end of the section, system comparisons will be presented in Table 1.

Freelancer: Hire & Find Jobs is a multilingual, easily navigable portal that works on mobile devices. Although it doesn't have scheduling or physical appointment features, it does have a full range of functions, including online payments, profile management, notifications, matching, and financial reporting.

Coach You offers convenient mobile access for coaching exchanges, complete with security safeguards and multilingual assistance. It does not have a single platform for scheduling in-person meetings, hiring venues, or managing coaches and coachees. It provides online payments, matching, scheduling, notifications, profile management, and thorough financial reports.

Upwork is designed for experienced users and offers multilingual assistance and global mobile accessibility. It has chat rooms, online payments, appointment tracking, matching, notifications, identity authentication, profile management, and matching functions. Real-time scheduling, actual appointments, and freelancer performance reports are not supported.

The proposed application includes features such as user-friendly interface, mobile-based accessibility, multilingual capabilities, internet connection requirements, a unified platform for coaches and trainees, identity authentication, profile management, notification alerts, matching and posting functionalities, time scheduling options, appointment tracking capabilities, online payment features, a chatting room for communication, rating & feedback mechanisms, finance report generation, and performance report tracking. The special features are venue booking, sport related platform and trust score referring system and also the platform for booking physical appointment.

The Freelance Coach mobile application differs from other platforms in a number of aspects. One of the notable aspects is that it offers an all-in-one solution for searching, booking and even scheduling sport locations. Finding and booking the ideal location for coaching sessions is another advantage offered by the solution, which is obviously not available on other platforms. In addition, the Freelance Coach mobile application is primarily aimed at sports coaches and trainees, whereas other applications are more focused on the educational sector. This particular specialization serves a specific target audience and allows users to focus on sports-related aspects on the platform, creating a dedicated environment for them.

Furthermore, Freelance Coach mobile application has added a trust score system to its referral. This technology promotes a sense of trust and openness in the coaching relationship by recommending trainees to lower-level coaches based on trust scores. Additionally, unlike other application that primarily use online booking approach, Freelance Coach mobile application offers a fully physical booking experience. This highlights the dedication of the platform to enabling face-to-face coaching sessions for those who favor or benefit from practical coaching. In summary, Freelance Coach redefines the coaching experience with its sports focused approach, revolutionary trust scoring system, and unique physical booking features. It also makes the process of choosing a coach and scheduling an appointment easier.

Table 1 *Existing System Comparison*

Features/System	Freelancer: Hire & Find Jobs	Coach You / Coach You | Coach	Upwork for Freelancers / Upwork for Client	Freelance Coach Mobile Application
User Friendly	√	√	X	√
Wed-Based	√	√	√	X
Mobile-Based	√	√	√	√
Multilanguage	√	√	X	√

Applied Information Technology and Computer Science Vol. 5 No. 2 (2024) p. 388-406

Table 1 *Existing System Comparison (Continued)*

Features/System	Freelancer: Hire & Find Jobs	Coach You / Coach You | Coach	Upwork for Freelancers / Upwork for Client	Freelance Coach Mobile Application
Internet Connection	√	√	√	√
Unified Coach and Trainee Platform	√	X	X	√
Identity Authentication	X	√	√	√
Profile Management	√	√	√	√
Notification	√	√	√	√
Matching and Posting	√	√	√	√
Time Scheduling	X	√	X	√
Referral	X	X	√	√
Appointment Tracking	√	√	√	√
Venue booking	X	X	X	√
Online Payment	√	√	√	√
Chatting Room	√	√	√	√
Rating & Feedback	√	√	√	√
Finance Report	√	√	√	√
Performance Report	√	X	X	√

Legend: √ = Yes； X= No

3. Methodology

Prototype modeling which shows in Figure 2 is the development process used for the Freelance Coach mobile application. The development process is divided into four main phases including planning, analysis, design and implementation. This method includes performing analysis, design, and implementation concurrently through an iterative process that proceeds until the system is finished [8]. The first system prototype usually involves only the main functionality or modules before it begins to receive evaluation and feedback. Afterwards that, the system goes through a series of iterative processes of re-analysis, re-design, and re-implementation in order to achieve accepted results for final system.

Prototype modeling was chosen as the development methodology because it helps to reduce the risk by correct the errors early. This is because the iterative process of analysis, design and implementation allows for the modules to be re-examined and testing at any time so that if the problems arise, it can be identified and corrected early. In addition, prototypes provide a visual representation of the concepts of the proposed application that can help the stakeholders and developers understand the proposed functionality and user interface. This helps refine the scope of the application and ensures that it meets the expectations result for proposed application.

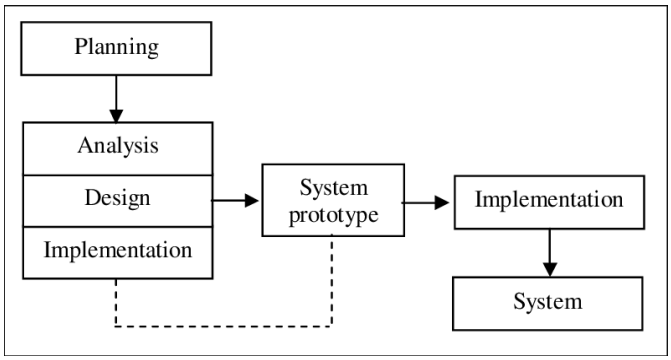


Fig. 2 *Prototype Model [5]*

3.1 Planning Phase

During the planning phase of developing a mobile application for freelance coaches, the primary focus was to establish clear objectives and define the scope and requirements of the application. The primary objective is to create a platform that seamlessly connects coaches and trainees and creates a collaborative environment conducive to create effective coaching relationship. The application was designed to facilitate scheduling and provide a user-friendly interface for coaches and trainees to effectively manage appointments. Additionally, the app aims to enable seamless communication, ensuring that coaches and trainees can easily interact within the application. In order to achieve these goals, the scope of the application was thoroughly explored, covering the features it would offer through research from the 3 similar application that are Freelancer, Upwork and Coach You applications. The project plan outlines a development period of July 20, 2023 to June 4, 2024 in the Gantt chart.

3.2 Analysis Phase

The analysis phase of the freelance coach mobile application prototyping model was a multifaceted process divided into several key steps. The initial focus was on information gathering, including interviews with stakeholders, particularly badminton coaches Lim Kheng Yang, former Johor state badminton player. These activities were aimed at gaining insights into existing coaching procedures and coaching qualification requirements. User observation techniques such as passive observation was also employed to further understand user preferences and needs. This hands-on approach enabled the developer to understand the complexity of the recruitment process. Subsequently, the data collected was carefully analyzed. The results of the analysis were crucial in identifying the functional and non-functional requirements of the freelance coach mobile application, ensuring that the subsequent development was a perfect fit with user expectations. Table 2 shows the all the functional requirements for the application.

Table 2 *Functional Requirement*

Function	Functionality	User
Register	This function allows to register an account by submitting personal information through a secure registration form.	Coach and trainee
Login	This function allows to login to the account by username and password and log out when it is not being used.	Coach, trainee and administrator
Account Management	This function allows to view and edit and delete the account	Coach, trainee and administrator
	This function allows the administrator to manage the coach account	
Posting	This function allows the coaches to create post, view post, edit post, delete post.	Coach and trainee
	This function allows the trainees to search, filter and view the posts	
Booking	This function allows the trainee to send the request from themselves or their dependent to the coaches	Coach and trainee
	This function allows the coach to accept or reject the request.	
	This function allows the coaches to register the connection coaches and to referral connection coach to the trainee.	
Chatting	This function allows the trainee and coach to communicate to each other	Coach and trainee
Payment	This function allows the trainees to pay the fee through Stripe payment gateway.	Trainee

Table 2 *Functional Requirement (Continued)*

Function	Functionality	User
Payment history and details	This function allows to view payment details, including transaction history and amounts paid.	Coach and Trainee
Rating and feedback	This function allows the trainees to rate the completed courses.	Coach, Trainee and Administrator
	This function allows the trainees and coaches view the rating and feedback	
	This function allows the administrator to remove the coaches with low rating and negative feedback	
Summary report	This function allows to generate the expenses report, activity report and performance report	Coach, trainee and administrator
Sport location	This function allows the administrator to create, update and delete the sport location	Administrator and Coach
	This function allows the coaches to view search and book the sport location	

A key aspect of the analysis phase is the design of Unified Modeling Language (UML) diagrams, which play a crucial role in visually representing the complex functionality of the application. Use case diagram for Freelance Coach mobile application visually depict the interaction between the user and the application which shown in Fig. A.1 in Appendix A. In this phase, the sequence diagram, activity diagram and class diagram have been proposed. Sequence diagrams clarify the sequence of objects interacting in each use case chronologically, providing a detailed view of each use case. On the other hand, activity diagrams are used to summarize workflows based on the administrator, coach and trainee. Meanwhile, class diagrams capture the static structure of the system, defining classes, their attributes, and relationships. In addition, the analysis phase includes the specification of software and hardware requirements. This includes detailing the technical components of the freelance coach mobile application. This step lays the foundation for the subsequent design and implementation phases, ensuring that the development team has a comprehensive understanding of the technical infrastructure required to realize the envisioned application.

3.3 Design Phase

The design phase is another stage in the development of a mobile application for freelance sport coach, where the comprehensive requirements gathered during the analysis phase are transformed into detailed system design specifications. This phase is the blueprint for the subsequent development process. The first step is to create an Entity Relationship Diagram (ERD) in Fig. B.1 in Appendix B, referred with the class diagram produced earlier. This ERD integrates classes, incorporates attributes and relationships, and lays the foundation for the development of a database system. The ERD of proposed application has 11 entities with different attributes. The database system will play an important role in storing and managing data, ensuring the functionality of the application. The data dictionary also had been done in this phase.

Once the ERD design is complete, we craft wireframes that outline the structure of the application. These wireframes include all the side of the users such as coach, trainee and administrator. These wireframes are an important reference for subsequent user interface development phases.

3.4 Implementation Phase

The implementation phase is the key stage in materializing the envisioned freelance coaching mobile application from design specifications to actual software. The developers carefully referenced the design blueprints from the previous phase, used Firebase as the designated database system, and relied on Android Studio and the Java programming language as the selected IDE for coding. The entire process included coding and prototyping of key modules such as registration, login, posting and booking. Each module was thoroughly tested to ensure seamless functionality.

Development of an iterative nature continued to be undertaken as testing feedback and inputs were incorporated, resulting in the addition of modules beyond the main functionality. Expanded functionality included tracking history, payment systems, and referral capabilities, among others. Iterative testing of modules, incorporation of feedback, and continuous refinement collectively drove the development process and ultimately the realization of the application. This dynamic approach ensured not only the utility of the core functionality, but also the incorporation of additional modules that enhanced the overall user experience.

Table 2 *Software development activities and their task*

Phase	Task	Output
Planning	1. Proposed final project title and idea 2. Identify the problem statement, objectives, project scope, expected outcomes and significance of the project. 3. Prepare a project work plan 4. Study online resources like research and journal articles related to the topic. 5. Study the functionality and features of existing applications.	1. Freelance Coach Mobile Application 2. Project proposal 3. Gantt chart 4. Business Process Diagram for coach find trainees and vice versa 5. Comparison and analysis between 3 existing applications which are Freelancers, Coach You and Upwork
Analysis	1. Interview with Coach Yang 2. Identify the specifications of the hardware and software requirements 3. Identify functional and non-functional requirements 4. Draw use case diagram, sequence diagram, activity diagram and class diagram	1. Interview questions 2. Hardware and software specification 3. Functional and non-functional requirements 4. Use case diagram, sequence diagrams, activity diagrams and class diagrams
Design	1. Design Entity Relationship Diagram (ERD) 2. Design wireframes	1. Entity Relationship Diagram (ERD) 2. Data Directories 3. Wireframes
Implementation (System Prototype)	1. Develop the system prototype module 2. Integrate the system prototype module 3. Connect to the database 4. Conduct the testing on the functions, modules and the system prototype 5. Carry out the first evaluation	1. Freelance Coach mobile application (login, register, post, booking) 2. Debug on the functions, modules and system prototype 3. Evaluation result
Implementation (Final Application)	1. Develop the entire system modules 2. Integrate the entire system modules 3. Connect to the database 4. Conduct the functional testing on the system 5. Carry out the user Acceptance	1. Freelance Coach mobile application 2. Debug on the functions, modules and system prototype 3. End-user feedback on application functionality and usability. 4. Identify any issues or inconsistencies 5. Debug required adjustments or fixes based on user feedback. 6. Final approval of system deployment

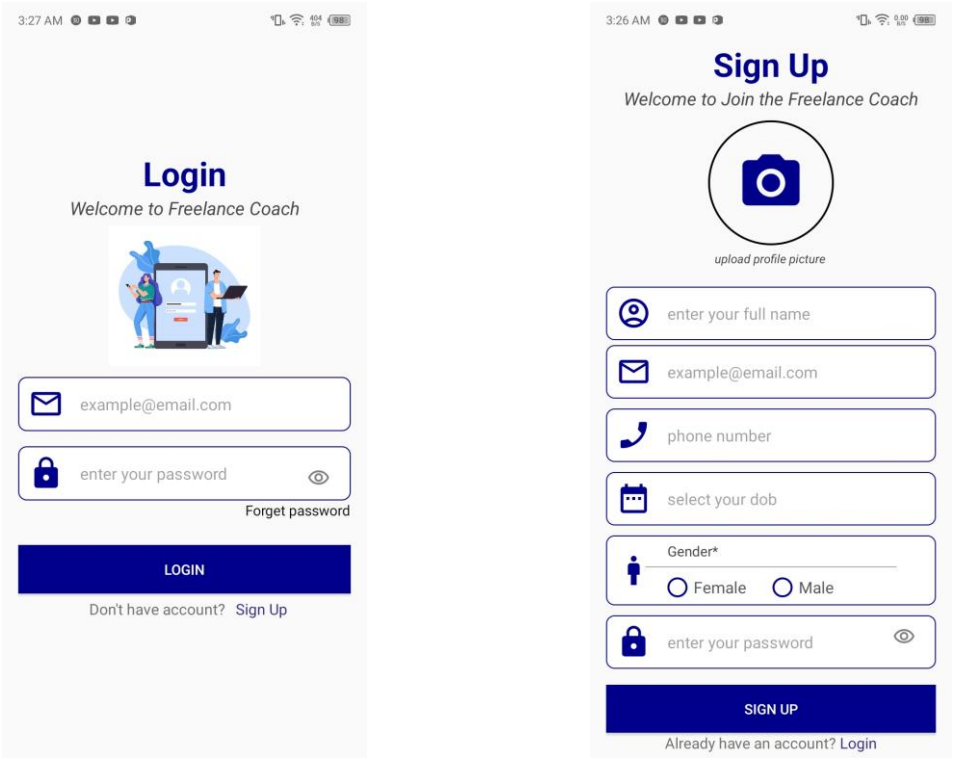
4. Result and Discussion

This section describes the system implementation in the Android Studio IDE using the chosen programming language (Java) and the user interface of the Freelance Sport Coach Mobile Application. In addition, the results of User Acceptance Testing and functionality testing will be further discussed to improve the product of the project.

4.1 Implementation

The Freelance Sport Coach mobile application is implemented using integrated libraries and frameworks with the aim of translating the stated specifications and requirements into actual code. Therefore, the application was developed using the Android Studio IDE. It uses xml to write the user interface design, java programming language to write the backend logic and integrates with firebase to manage the database.

Figure 3(a) shows the login user interface. The login user interface is the same for Coaches, Trainees, and Administrators, and allows users to log in to their accounts with an email and password. Once the email and password are valid for firebase authentication, the user can successfully log in. Figure 3(b) shows the registration form user interface. The registration user interface is the same for both coaches and trainees, and both collect data such as profile image, email, phone number, date of birth, gender, and password. Once completed, the registration is successful and takes the user directly to the login page to log into the account using the previously registered email and password.



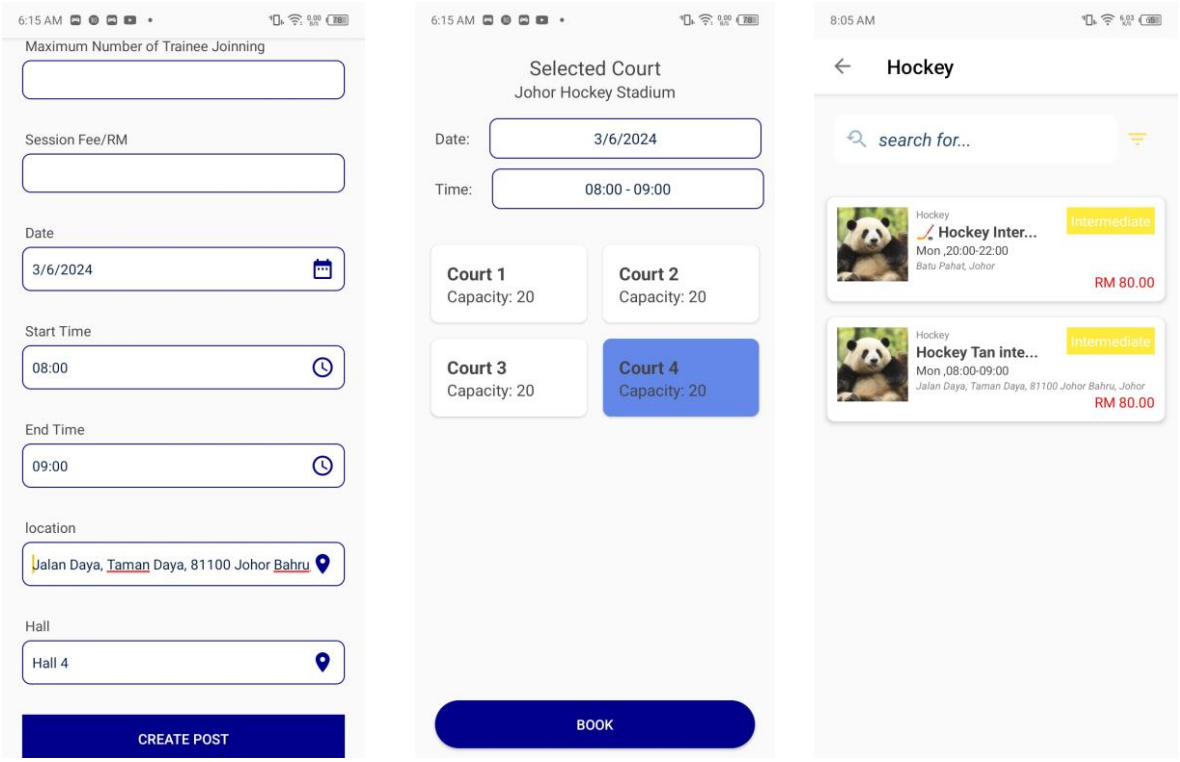
(a) **(b)**

Fig. 3 *User Account Management Module (a)Login User Interface; (b) Register Account User Interface*

Figure 4 shows the code snippet to retrieve input data from SelectCourtActivity. The code snippet in Figure 3 processes the results returned from another activity. Specifically, it is used to process the result when the user selects a sport location and its hall from a list in another activity. If the returned intent data is not null, the method extracts the hall name, capacity and location address. These values will be used to update the corresponding input fields in the CreateTask activity, which will then be saved to the database with the rest of the input. Figure 4 show the create post user interface. Figure 5(a) shows the user interface to allow the coach to input all the relevant data such as date, time and location. Once the coach selects the date and time then only be available to choose the sport location. Figure 5(b) shows the user interface for choosing the court and hall. If the hall is full based on the capacity, then it will close and the coach was unable to select. Once the coach done the selection and click book, the data will pass back to the create post activity for further action. Once the post have successfully created then the post will shown in the trainee view ad Figure 5(c).

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == REQUEST_CODE_SELECT_COURT && resultCode == RESULT_OK) {
        if (data != null) {
            String selectedHall = data.getStringExtra("selectedHall");
            int selectedCapacity = data.getIntExtra("selectedCapacity", -1);
            String locationAddress = data.getStringExtra("locationAddress");

            // Update EditText fields with the returned data
            locationEditText.setText(locationAddress);
            hallEditText.setText("Hall " + selectedHall);
            // You can also update other fields if necessary, like capacity
            // selectedCapacityEditText.setText(String.valueOf(selectedCapacity));
        }
    }
}
```

Fig. 4 *Retrieve Data Input from SelectCourtActivity*



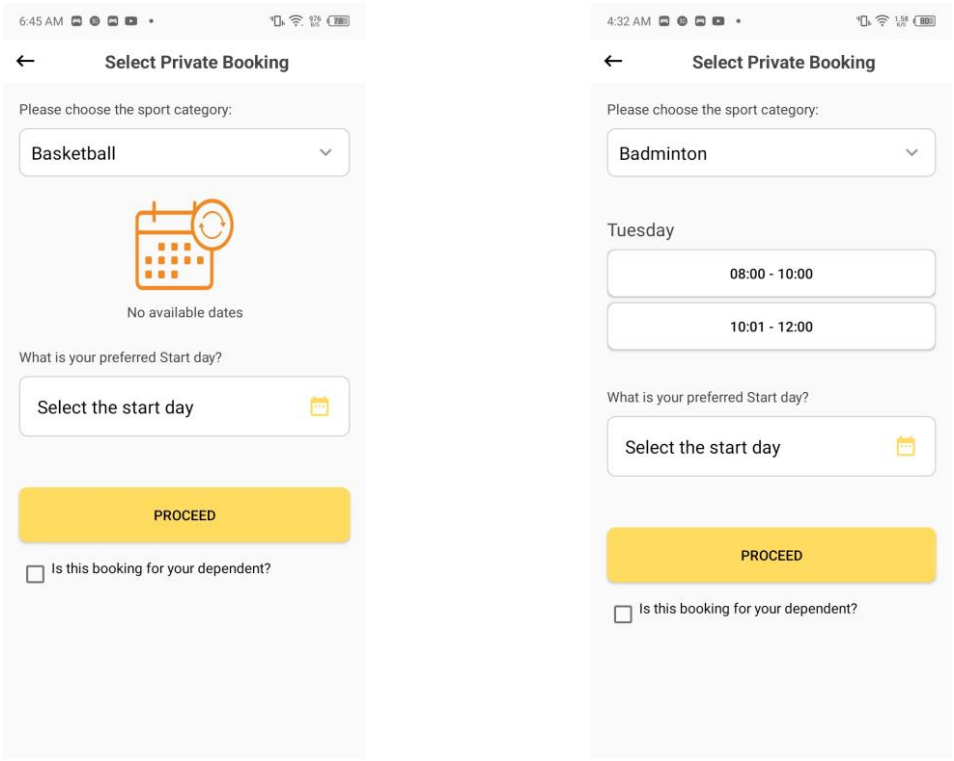
(a) **(b)** **(c)**

Fig. 5 *Create Post Data (a) Create Post Activity; (b) Select Court Activity; (c) Post View in Trainee*

Figure 6 shows the code snippet that displays the day and time of the scheduling for an individual booking. First, when a sport category is detected or selected, if an available post is found, it would be displayed by date and time to allow the trainee to select the preferred time and date. Otherwise, if no posts are found for that sport category, a no data image is displayed. Figure 7(a) show the personal booking without the schedule so the trainee was unavailable to book for the coach. Figure 7(b) show the personal booking with the available time for the coach so the trainee is available to choose the date and time and send the booking request to the related coach.

```
private void displaySchedule() {
    dynamicContainer.removeAllViews();

    if (scheduleMap.isEmpty()) {
        noAvailableDates.setVisibility(View.VISIBLE);
        noAvailableImage.setVisibility(View.VISIBLE);
        dynamicContainer.setVisibility(View.GONE); // Hide the GridLayout container
    } else {
        noAvailableDates.setVisibility(View.GONE);
        noAvailableImage.setVisibility(View.GONE);
        dynamicContainer.setVisibility(View.VISIBLE); // Show the GridLayout container

        for (Map.Entry<String, List<Post>> entry : scheduleMap.entrySet()) {
            String day = entry.getKey();
            List<Post> posts = entry.getValue();

            // Add day TextView
            TextView dayTextView = new TextView(this);
            dayTextView.setText(day);
            dayTextView.setTextSize(18);
            dayTextView.setPadding(0, 16, 0, 16);
            dynamicContainer.addView(dayTextView);

            // Add time slots GridLayout
            GridLayout timeSlotsGrid = new GridLayout(this);
            timeSlotsGrid.setColumnCount(1);
            timeSlotsGrid.setLayoutParams(new LinearLayout.LayoutParams(
                    LinearLayout.LayoutParams.MATCH_PARENT,
                    LinearLayout.LayoutParams.WRAP_CONTENT));
            dynamicContainer.addView(timeSlotsGrid);

            for (Post post : posts) {
                String timeSlot = post.getStartTime() + " - " + post.getEndTime();
                addButtonToGridLayout(timeSlot, timeSlotsGrid, post);
            }
        }
    }
}
```

Fig. 6 *Display the Schedule with the Event Personal Course*



(a) **(b)**

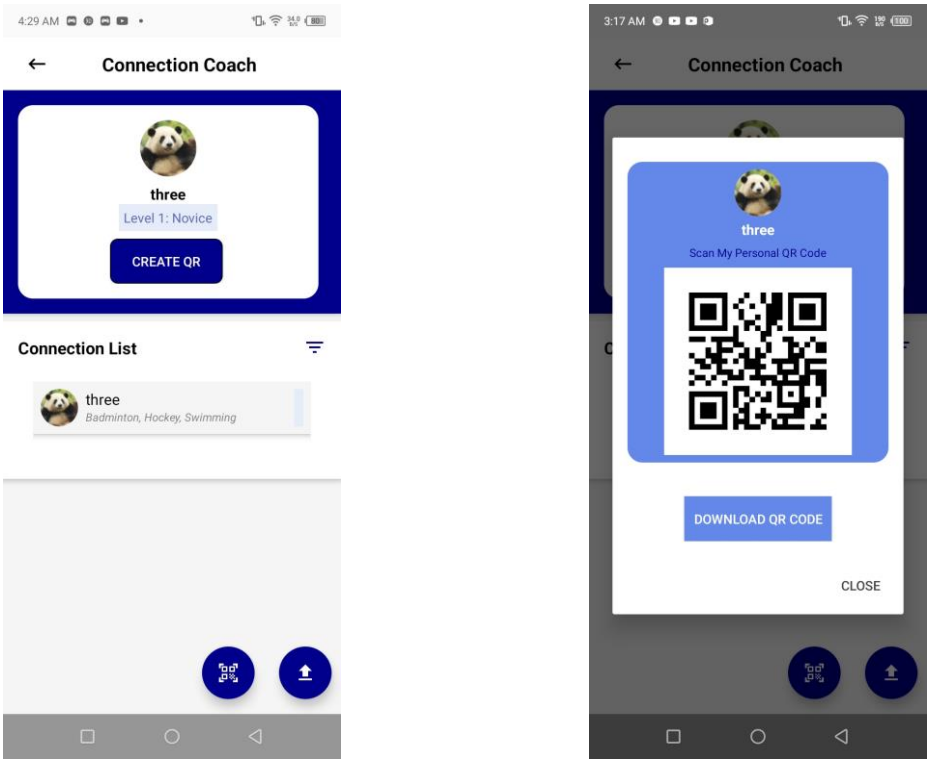
Fig. 7 *Book Personal Post Detail (a) Book for Personal Booking without schedule; (b) Book for Personal Booking with schedule*

Figure 8 shows the code snippet of generating QR connection Code for main Coach. To generate a QR code in the onClick() method of the provided Java code snippet, the application needs to rely on a Coach object that may contain a method called generateQRCode(). The task of this method is to convert the specific data associated with the coach ID into a QR code. The actual QR code generation involves encoding this data into a visual format

that can be easily scanned using a QR code reader. The method generateQRCode() uses a QR code library which was the ZXing ("Zebra Crossing") library, which must be imported and integrated into the Android Studio. This library handles the complexities of QR code generation, such as error correction, encoding and graphical representation. After the QR code is generated, it is converted into a bitmap and then displayed to the user via a showQRCodeDialog() which shown in Figure 9(B) that provides a visual representation of the QR code that can be scanned by other devices. Figure 9(a) shows the add connection coach user interface which the main coach can generate the QR code and the connection coach can scan the code and register under the main coach.

```
generateQRButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {
            if (coach != null) {
                qrCodeBitmap = coach.generateQRCode();
                showQRCodeDialog(qrCodeBitmap);
            } else {
                Toast.makeText(ConnectionCoach.this, "Coach data is not initialized", Toast.LENGTH_SHORT).show();
            }
        } catch (WriterException e) {
            e.printStackTrace();
            Toast.makeText(ConnectionCoach.this, "Error generating QR code", Toast.LENGTH_SHORT).show();
        }
    }
});
```

Fig. 8 *Generate QR Connection Code for Main Coach*



(a) **(b)**

Fig. 9 *Connection Coach Registering (a) Generate and Scan Connection User Interface; (b) QR Code Generated*

Figure 10 shows the code snippet of generate and sharing the referral link to the trainee. First, it will build the referral link based on the parameters of Coach id, trainee id and also the connection coach id. It checks for null values and logs an error if any error is found. If all parameters are valid, it builds the link and sets it to TextView. Shared referral link function is used for sharing the built referral link. It retrieves the link and associated ID and checks for null values in case of an error. If there are no errors, the referral link will be shared directly to the relevant trainee's chat room.

```
private void generateReferralLink(Connection connection, String traineeId, String connectionCoachId) {
    if (connection == null || traineeId == null || connectionCoachId == null) {
        Log.e("generateReferralLink", "Connection, Trainee ID, or Connection Coach ID is null");
        return;
    }

    String referralLink = "app://openPostDetail?coachId=" + connection.getCoachId() + "&traineeId=" + traineeId + "&connectionCoachId=" + connectionCoachId;
    urlLinkTextView.setText(referralLink);
}

private void shareReferralLink(Connection connection) {
    String referralLink = urlLinkTextView.getText().toString();

    // Log referralLink and traineeId to debug
    Log.d("shareReferralLink", "Referral Link: " + referralLink);
    Log.d("shareReferralLink", "Trainee ID: " + traineeId);
    Log.d("shareReferralLink", "Coach ID: " + connection.getCoachId());

    if (traineeId == null || referralLink == null) {
        Log.e("shareReferralLink", "Trainee ID or Referral Link is null. Cannot proceed.");
        return;
    }

    // Generate the correct chatroom ID
    String chatroomId = getChatroomId(coachId, traineeId);

    // Check if the chatroom already exists
    checkChatroomExists(chatroomId, exists -> {
        if (exists) {
            // Chatroom exists, directly share the link
            sendReferralLink(chatroomId, referralLink);
        } else {
            // Chatroom does not exist, create it first then share the link
            createChatroom(chatroomId, coachId, traineeId, () -> sendReferralLink(chatroomId, referralLink));
        }
    });
}
```

Fig. 10 *Generate a Shareable Referral Link with Coach ID Parameter*

Figure 11 shows the code snippet of scheduling the weekly Booking. This code snippet is designed to schedule weekly bookings for accepted bookings. It first gets the details of the initial booking and then calculates and schedules 10 additional weekly bookings starting from the start date of the original booking. In addition, each new booking is only saved if its date is after the current date to ensure that only future bookings are scheduled. The new booking inherits the details of the original booking such as Post ID, Trainee ID, coach ID and so on. The new booking then will be marked as the blue colour at the calender to alert the coach or trainee for the upcoming or completed classes for checking. The yellow colour indicated the current date. Figure 12(a) shows the upcoming schedule while Figure 12(b) shows the completed schedule user interface.

```
private void scheduleWeeklyBookings(String bookingId) {
    Log.d("Booking", "Scheduling weekly bookings for bookingId: " + bookingId); // Log that scheduling weekly bookings is starting
    DatabaseReference bookingRef = FirebaseDatabase.getInstance().getReference().child("bookings").child(bookingId);
    bookingRef.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            Log.d("Booking", "Inside onDataChange for bookingId: " + bookingId); // Log that onDataChange is triggered
            Booking initialBooking = dataSnapshot.getValue(Booking.class);
            if (initialBooking != null) {
                initialBooking.fetchRelatedData(new Booking.FetchDataListener() {
                    @Override
                    public void onDataFetched(Booking booking) {
                        if (booking.getPost() != null && booking.getPost().getDate() != null) {
                            Log.d("Booking", "Initial booking and post data retrieved successfully.");
                            DateTimeFormatter dateFormatter = DateTimeFormatter.ofPattern("d/M/yyyy", Locale.ENGLISH);
                            LocalDate startDate = LocalDate.parse(booking.getPost().getDate(), dateFormatter);
                            LocalDate currentDate = LocalDate.now(); // Get the current date

                            for (int i = 1; i <= 10; i++) { // Example: schedule 10 additional weekly bookings
                                LocalDate nextBookingDate = startDate.plusWeeks(i);
                                if (nextBookingDate.isAfter(currentDate)) { // Only save if next booking date is after the current date
                                    Log.d("Booking", "Next booking date: " + dateFormatter.format(nextBookingDate)); // Log the next booking date
                                    Booking newBooking = new Booking(
                                            null, // Booking ID will be generated when saving to Firebase
                                            booking.getPostId(),
                                            booking.getTraineeId(),
                                            booking.getCoachId(),
                                            "Upcoming", // Initial status for new bookings
                                            booking.getTotalAmount(),
                                            dateFormatter.format(nextBookingDate),
                                            "Unpaid",
                                            "",
                                            booking.getDependentId()
                                    );

                                    saveBookingToFirebase(newBooking);
```

Fig. 11 *Schedule Weekly Booking*

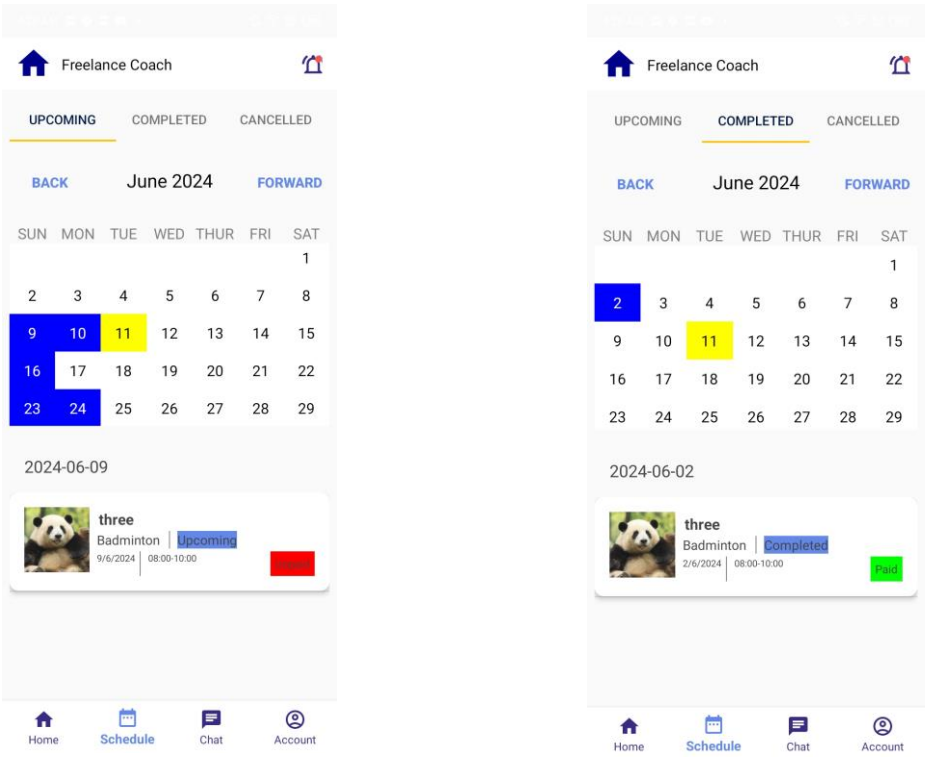


Fig. 12 *Scheduling Course event (a) Scheduling for Upcoming Event; (b) Scheduling for Completed Event*

4.2 Functional Testing

Functional testing verifies that the system performs specific tasks and functions as expected based on the requirements set by the stakeholders. Functional testing for Freelance Sport Coach Mobile Application aims to find any bugs and errors that might affect its performance, dependability, or usability in particular with the area of exception handling, the user interface, and integration with the Firebase database. The application's quality had been guaranteed by creating a test plan, which is described in Table 4, and it passed every test with no issues.

Table 3 *Test Plan for Application Functionalities*

Function	Functionality	User	Actual Output
Register	This function allows to register an account by submitting personal information through a secure registration form.	Coach and trainee	Pass
Login	This function allows to login to the account by username and password and log out when it is not being used.	Coach, trainee and administrator	Pass
Account Management	This function allows to view and edit and delete the account	Coach, trainee and administrator	Pass
	This function allows the administrator to manage the coach account		Pass
Posting	This function allows the coaches to create post, view post, edit post, delete post.	Coach and trainee	Pass
	This function allows the trainees to search, filter and view the posts		Pass

Table 3 *Test Plan for Application Functionalities (Continued)*

Function	Functionality	User	Actual Output
Booking	This function allows the trainee to send the request from themselves or their dependent to the coaches	Coach and trainee	Pass
	This function allows the coach to accept or reject the request.		Pass
	This function allows the coaches to register the connection coaches and to referral connection coach to the trainee.		Pass
Chatting	This function allows the trainee and coach to communicate to each other	Coach and trainee	Pass
Payment	This function allows the trainees to pay the fee through Stripe payment gateway.	Trainee	Pass
Payment history and details	This function allows to view payment details, including transaction history and amounts paid.	Coach and Trainee	Pass
Rating and feedback	This function allows the trainees to rate the completed courses.	Coach, Trainee and Administrator	Pass
	This function allows the trainees and coaches view the rating and feedback		Pass
	This function allows the administrator to remove the coaches with low rating and negative feedback		Pass
Summary report	This function allows to generate the expenses report, activity report and performance report	Coach, trainee and administrator	Pass
Sport location	This function allows the administrator to create, update and delete the sport location	Administrator and Coach	Pass
	This function allows the coaches to view search and book the sport location		Pass

4.3 User Acceptance Test

Functional testing verifies that the system performs specific tasks and functions as expected based on the requirements set by the stakeholders. Functional testing for Freelance Sport Coach Mobile Application aims to find any bugs and errors that might affect its performance, dependability, or usability in particular with the area of exception handling, the user interface, and integration with the Firebase database. The application's quality had been guaranteed by creating a test plan, which is described in Table 4, and it passed every test with no issues.

The Freelance Sports Coach Mobile Application evaluation had a total of 20 participants, of which 17 trainees and 3 coaches completed the Google Form. Besides that, fourteen of the participants were male and six were female. The user satisfaction scale varies from very dissatisfied to very satisfied, with a value between 1 and 5. In Figure 12 shows the satisfaction of the participants with the different features of the user interface which ranged from very dissatisfied to very satisfied. Most of the users were very satisfied with the navigation, ease of use and comprehension of the interface, which indicates that the application is well structured and user friendly. Users also expressed satisfaction with the interface layout and visual design such as the colours and backgrounds which indicates that the aesthetic elements of the interface meet the expectation. Satisfaction with textual elements such as font and size were also high which indicates that it was easy for users to view and understand the text in the application. The overall bar chart suggests that overall user satisfaction with the interface is high.

Figure 13 shows the result of application features evaluation for trainees. Based on the data collected, it shows that participants rated their satisfaction with the functionality of the application between 4 and 5. The total participants is 17 trainees. Specifically, for tasks such as logging in, managing profile account, searching and viewing posts, there were 6 participants were satisfied and 11 were very satisfied. For tasks such as registering, paying via stripe, and viewing payment history and booking details, 4 participants were satisfied and 13 were very satisfied. Additionally, for being able to request and manage bookings and being able to manage dependencies, 5 participants voted only Satisfied and 12 voted Very Satisfied. In terms of being able to manage scheduling, 7 participants voted Satisfied and 10 voted Very Satisfied. Additionally, for being able to chat, being able to provide and view ratings and reviews, and being able to generate reports, there were 3 participants voted satisfied and 14 voted very satisfied.

Figure 14 shows the result of application features evaluation for coaches. Based on the data collected, it shows that participants rated their satisfaction with the functionality of the application between 4 and 5. The total participants is 3 coaches. Specifically, for tasks such as registering, logging in, viewing, editing, and deleting coaching account details, creating, editing, and deleting posts, receiving and managing bookings, and viewing ratings and reviews, all participants indicated that they were very satisfied. In addition, for features such as logging in for the first time, recommending connecting coaches, chatting with participants, searching, viewing and booking exercise locations, and generating reports, one participant indicated only satisfied and two participants indicated very satisfied. In terms of being able to manage schedules and view payment history and receipts, 2 participants were satisfied and 1 was very satisfied.

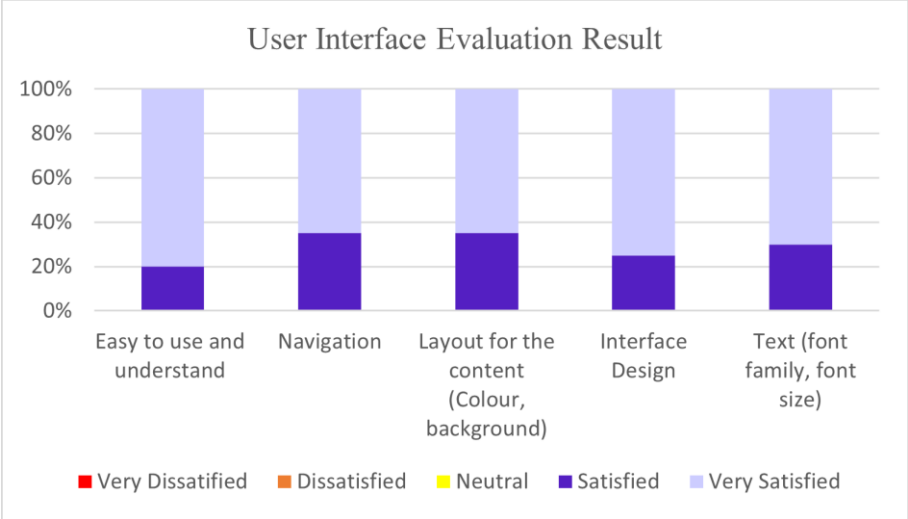


Fig. 12 *Diagram of User Interface Evaluation Result.*

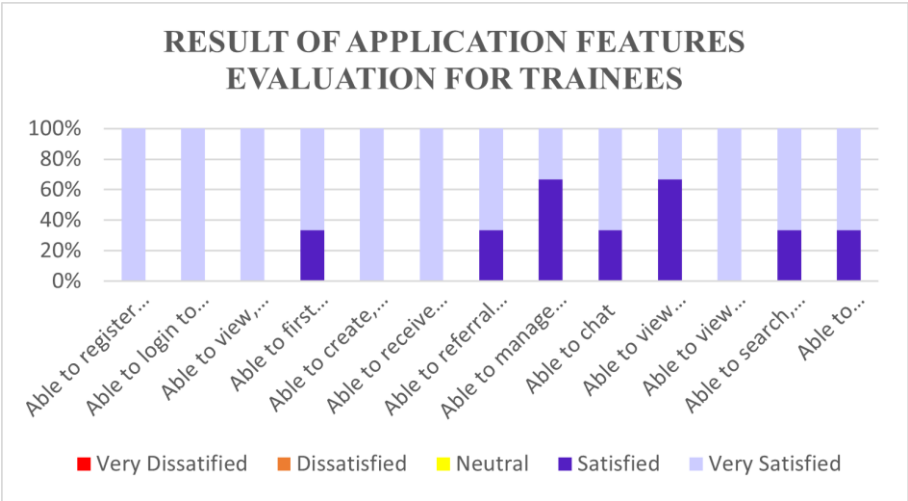


Fig. 13 *Diagram of the Result of Application Features Evaluation for Trainees*

Applied Information Technology and Computer Science Vol. 5 No. 2 (2024) p. 388-406

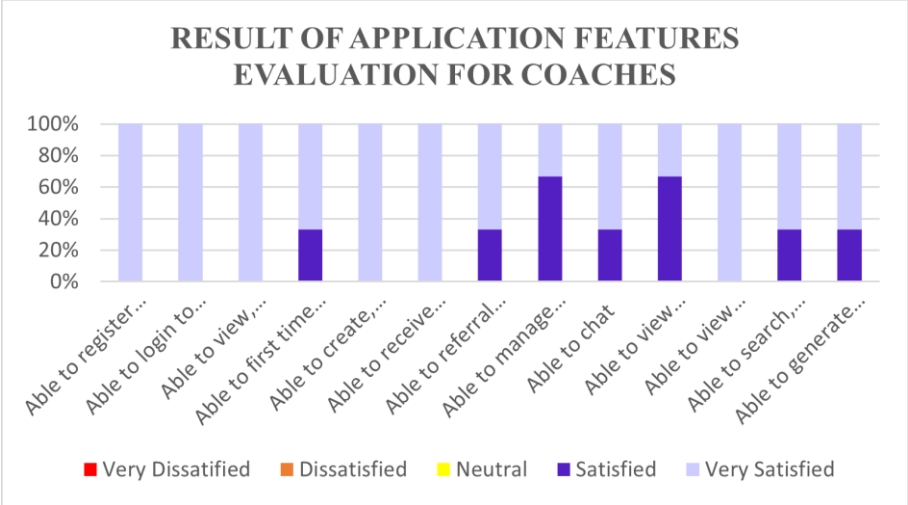


Fig. 14 *Diagram of the Result of Application Features Evaluation for Coaches*

5. Conclusion

This paper provides an overview of the development of a mobile application for freelance sports coaches which begins with a description of the objectives of the application and the current method of contacting coaches and trainees which is illustrated by a business process diagram. The paper compares three existing applications and details the process of developing the proposed application using a prototype model. The development phases are systematically presented, followed by an overview of the implementation modules, functional testing and user acceptance testing (UAT). The application provided an integrated platform for coaches and trainees to create posts, filter the post, book and view schedules. Test results showed positive feedback, emphasizing the simplicity, clearness and user-friendliness of the interface.

The Freelance Sport Coach mobile application has a number of limitations which include restricted management of sport categories which affects adaptability to a wide range of sports and user needs. Additionally, only administrators can add new sport locations and are unable to update details on time which reduces the flexibility of the application. The application also lacked a refund feature which negatively impacted the user experience when classes were cancelled.

To enhance the functionality and user experience, several improvements have been suggested. Administrators should have the features to create, update, and delete sports categories to increase flexibility. Additionally, developing a dedicated administration site for sport location owners to manage sport locations would ensure timely updates and improve accuracy. Besides that, establishing clear cancellation and refund policies along with integrating automated processes would improve transparency and efficiency.

Acknowledgement

I would like to express my sincere appreciation to the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, for their precious support during the research and development of this project. The guidance and resources provided by the faculty played a key role in the outcome of my work. My sincere thanks to the faculty for their strong commitment to academic excellence and research, which contributed significantly to the successful completion of this project.

Author Contribution

The authors confirm contribution to the development of Freelance Coach Mobile Application as follows: study conception and design: Lieu Yee Zhe, Norfaradilla Wahid; data collection: Lieu Yee Zhe; analysis and interpretation of results: Lieu Yee Zhe, Norfaradilla Wahid; draft manuscript preparation: Lieu Yee Zhe. All authors reviewed the results and approved the final version of the manuscript.

References

[1] C. Jean and T. Jennifer, "dLib.si - The dynamic process of development through sport," *www.dlib.si*, 2014. https://www.dlib.si/details/URN:NBN:SI:doc-78F5K2TW (accessed Jul. 29, 2024).

[2] C. Li, C. K. J. Wang, and D. Y. Pyun, "Talent Development Environmental Factors in Sport: A Review and Taxonomic Classification," Quest, vol. 66, no. 4, pp. 433–447, Oct. 2014, doi: https://doi.org/10.1080/00336297.2014.944715.

[3] D. Clark, "Sports & fitness occupations 2011-2020," *Statista*, Nov. 16, 2023. https://www.statista.com/statistics/319296/number-of-people-with-sports-occupations-in-the-united-kingdom-uk/ (accessed Nov. 21, 2023).

[4] N. Holt and C. Knight, "Parenting in Youth Sport," *Parenting in Youth Sport: From Research to Practice*, Mar. 2014, doi: https://doi.org/10.4324/9780203798553.

[5] "Freelancer: Hire & Find Jobs - Apps on Google Play," *play.google.com*, Apr. 03, 2014. https://play.google.com/store/apps/details?id=com.freelancer.android.messenger&hl=en&gl=US

[6] "COACH YOU - Apps on Google Play," *play.google.com*, Aug. 10, 2020. https://play.google.com/store/apps/details?id=com.technzone.coachyou&hl=en&gl=US

[7] "upwork - Android Apps on Google Play," *play.google.com*, Apr. 15, 2019. https://play.google.com/store/search?q=upwork&c=apps&hl=en&gl=US

[8] A. Dennis and Barbara Haley Wixom, *Systems analysis design*. New York: J. Wiley, 2003, pp. 9–12. Accessed: Dec. 16, 2023. [Online]. Available: http://projanco.com/Library/Systems%20Analysis%20and%20Design-An%20Object-Oriented%20Approach%20with%20UML-2015.pdf

Applied Information Technology and Computer Science Vol. 5 No. 2 (2024) p. 388-406

Appendix A: Use Case Diagram

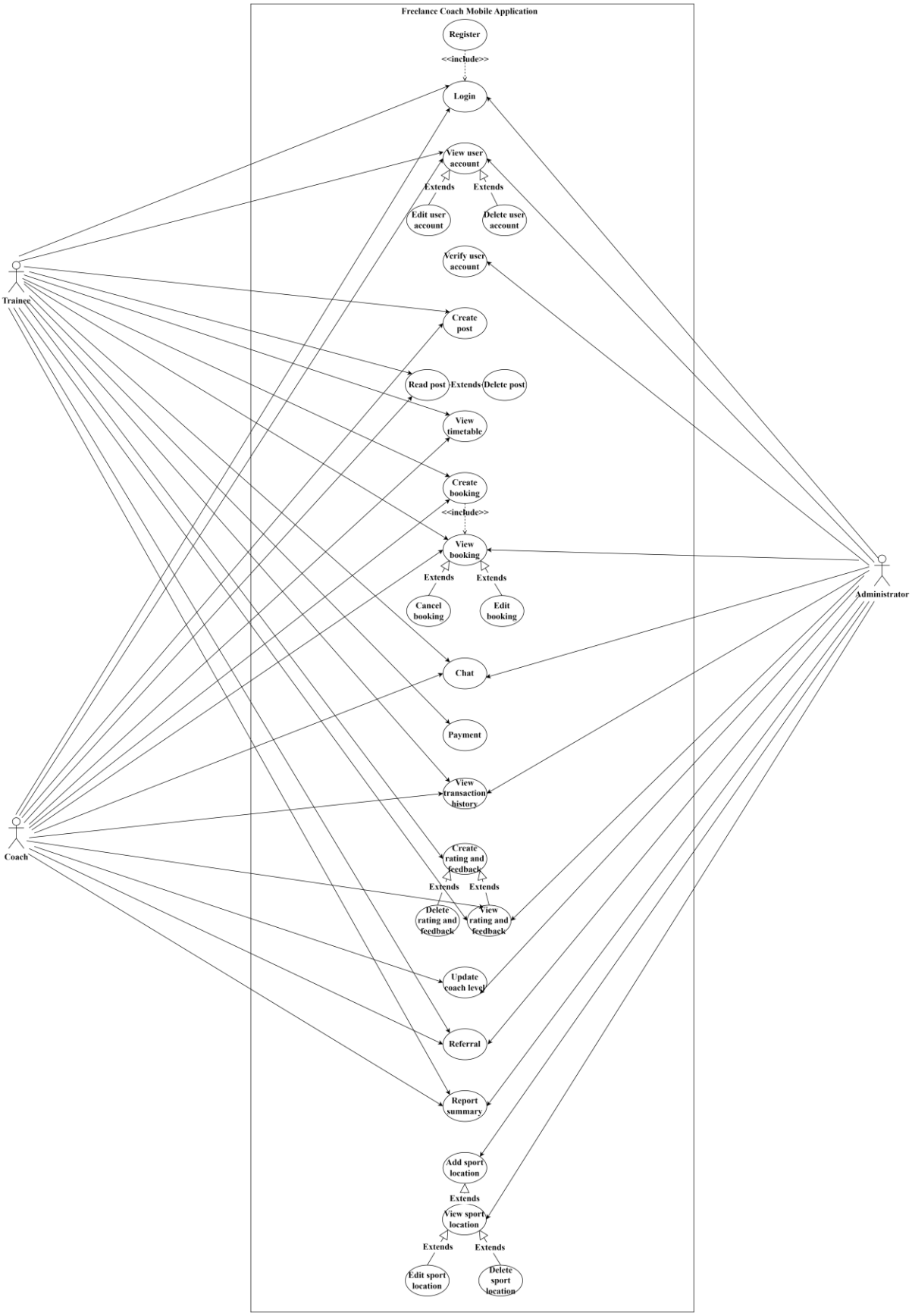


Fig. A.1 *Use Case Diagram for Freelance Coach Mobile Application*

Appendix B: Entity Relationship Diagram (ERD)

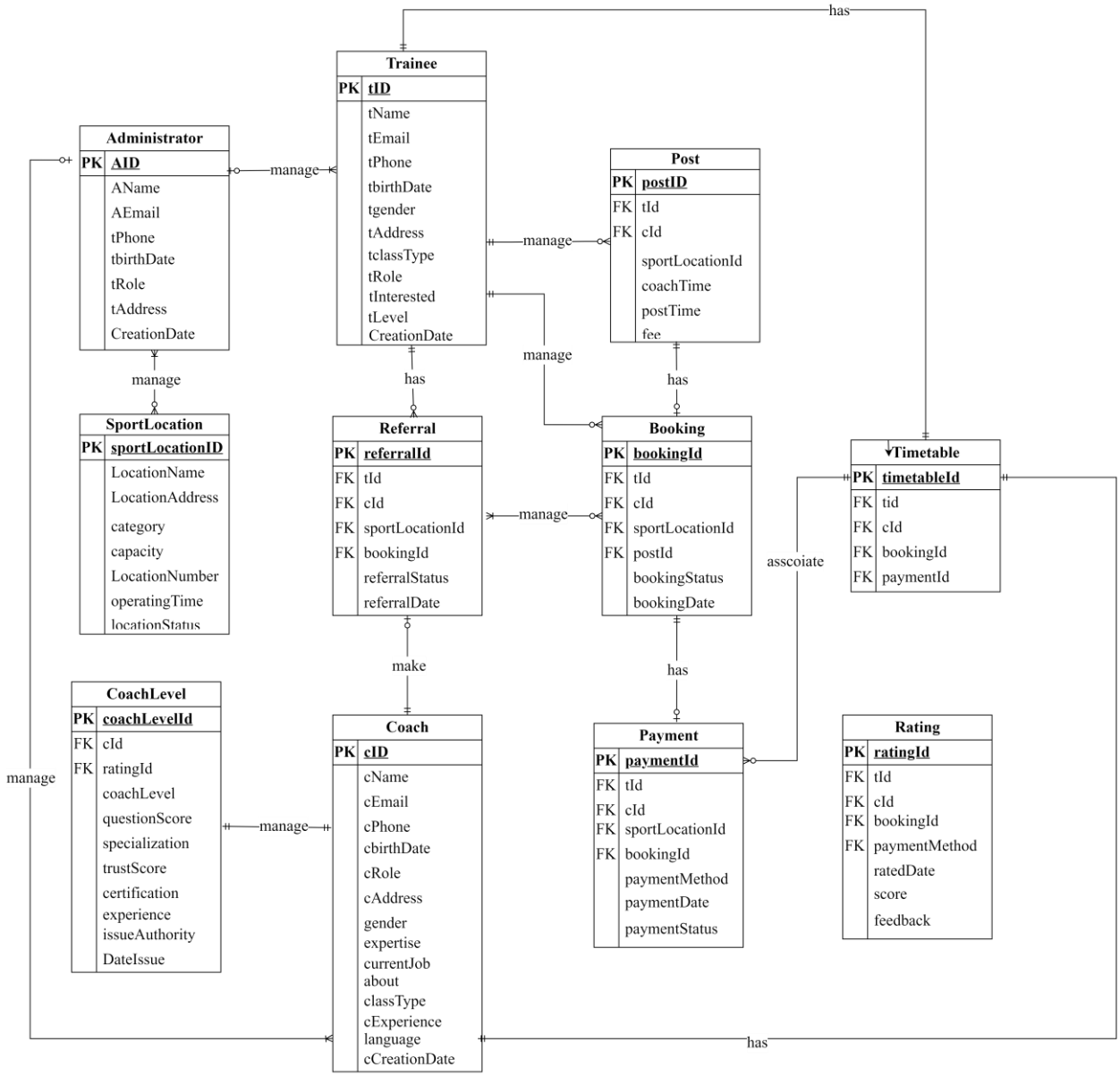


Fig. B.1 *Entity Relationship Diagram for Freelance Coach Mobile Application*