



Capstone Project – Phase B

Co-Bie

An application to find people with common hobbies

Project Code: 23-1-D-15

Supervisor

Prof. Zeev Brazily

zbarzily@braude.ac.il

Raz Avraham – 314788001

Raz.Avraham@e.braude.ac.il

Amit Kempner – 206078537

Amit.Kempner@e.braude.ac.il

Table of contents

1. Abstract.....	3
2. Introduction.....	4
1. Stakeholders.....	5
3. Background.....	6
1. Related Work.....	6
2. Survey.....	7
4. Project Review & Process Description.....	8
1. Our Solution.....	8
2. Agile Development.....	8
3. Mobile Development.....	9
4. Android.....	9
5. Client-Server Model.....	10
6. Mobile Cloud Computing.....	10
7. Project Goals.....	11
8. Unique Features.....	11
9. Process.....	12
10. Engineering Challenges & Solutions.....	13
11. Evaluation / Verification Plan.....	15
12. Results & Conclusion	19
5. User Documentation.....	20
1. General Description.....	20
2. User Interface.....	21
6. Maintenance Guide.....	31
1. Product.....	31
2. Use-Case Diagram.....	34
3. Class Diagram.....	35
4. Package Diagram.....	36
5. Activity Diagram.....	37
1. Create Event.....	37
2. Join Event.....	39
3. Add Review.....	40
7. Program Structure Description.....	41
1. DB Scheme.....	41
2. System Requirements.....	44
8. References + Git Link.....	45

Abstract

The Co-Bie application is a groundbreaking solution that connects individuals with shared hobbies, addressing the prevalent challenge of finding like-minded enthusiasts to engage in hobby-related activities. Through extensive research, meticulous planning, and the utilization of agile development methodologies, Co-Bie offers a user-friendly platform that enables users to connect, share, and organize meetups based on their interests.

By conducting surveys and gathering insights from the target audience, the development team ensured that the application met the needs and preferences of hobbyists. The architecture of Co-Bie follows a client-server model, with the Android client running on devices and utilizing Firebase as the cloud-based backend. This combination provides scalability, reliability, and real-time synchronization of data.

The Co-Bie application boasts a comprehensive set of features designed to enhance user engagement and facilitate meaningful connections. Features such as personalized event recommendations, live chat functionality, and notifications based on user preferences create a dynamic and interactive experience. The integration of Google Maps allows for seamless navigation and location-based event discovery.

Engineering challenges were addressed through meticulous problem-solving, resulting in effective solutions. The application underwent rigorous testing to ensure a seamless and enjoyable user experience, maximizing its impact and value. The well-designed user interface, supported by visual diagrams, ensures intuitive interaction and a visually appealing aesthetic.

The Co-Bie application demonstrates the feasibility and effectiveness of connecting individuals with shared hobbies. It provides a convenient platform for hobbyists to share their expertise, seek advice, and organize meetups and events. By fostering meaningful connections and enriching hobby-based communities, Co-Bie empowers users to enhance their hobby-related experiences and create lasting connections.

Introduction

In today's world, there are countless individuals with diverse hobbies, many of whom struggle to find like-minded individuals to share their hobbies with. Existing platforms partially address this problem by enabling people to connect based on shared hobbies but often lack a focus on facilitating specific events. Our solution aims to bridge this gap by creating a platform that allows for spontaneous and quick meetups among individuals with common hobbies.

We propose the development of an Android app designed to connect people with shared hobbies and facilitate the organization of meetings. This app will offer a comprehensive set of features to streamline the process of setting up gatherings. Users will be able to create meetings and provide relevant details such as the purpose, number of participants, location, and time. Other app users can then register for these meetings and confirm their attendance. Additionally, a live chat functionality will be available for seamless communication between participants.

Each app user will have a basic profile containing essential identifying information such as their name, age, photo, and hobbies. To ensure the safety and trustworthiness of the community, we will implement a reporting system where users can flag any suspicious or untrustworthy individuals, which will contribute to a measure of reliability.

The app will support two main types of events: physical and virtual. Physical events will be displayed on a map interface. Virtual events, on the other hand, will be presented as a sorted list based on the user's chosen hobbies.

To actively engage users, notifications will be sent based on various triggers, ensuring they stay informed about new meetings and have the opportunity to join them.

By combining the power of technology and a focus on real-time meetups, our app aims to connect individuals with shared hobbies, fostering meaningful connections and enabling the formation of hobby-based communities.

Stakeholders:

The potential stakeholders of our system are:

- People from all the population layers who do not have partners to share their hobbies with.
- People who are looking for new partners for diversity and new friendship.
- People and organizations interested in organizing a tournament/competition in a certain field (for example, a municipality organizing a basketball tournament for the city's residents).
- People with disabilities who are unable to leave their homes and want to share their hobby virtually.

Background

Related Work:

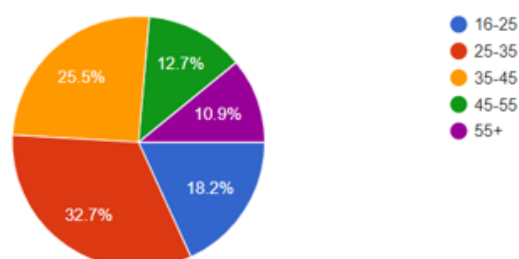
There are different platforms that offer a solution to the problem. Each platform has its own unique approach to user interaction. We will detail some of the platforms:

- **Pinterest** – Online social network similar to a bulletin board. The service allows its users to create and manage their own photo collections. The stated purpose of the site is "to connect all the people in the world through things that interest them".
The social network does not provide the option of social meeting but focuses on sharing hobbies and creating a conversation about them.
- **Meetup** – Social media platform for hosting and organizing in-person and virtual activities, gatherings, and events for people and communities of similar interests, hobbies, and professions. Meetup mainly focuses on getting to know people with similar interests but does not emphasize the existence of the hobby itself during the meeting. For example: a chess meetup focuses on getting to know people who like to play chess, but the main purpose of the meeting will not be to play the chess game.
- **Facebook** – A website operated by the American company Meta. The site is the largest online social network in the world. Facebook gathers many groups under it. There is a group for almost every topic that exists. The group is a platform where you can have conversation, get to know each other and have a joint discussion, but there is no hierarchical and organized model through which you can choose a hobby and hold a meeting.

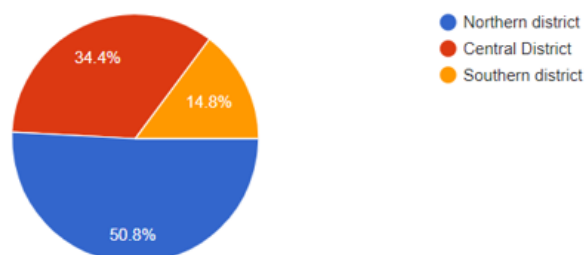
Survey:

In order to get an indication of the problem, we conducted a survey that included 73 people in a diverse age range and residential area. Below are the questions and results of the survey:

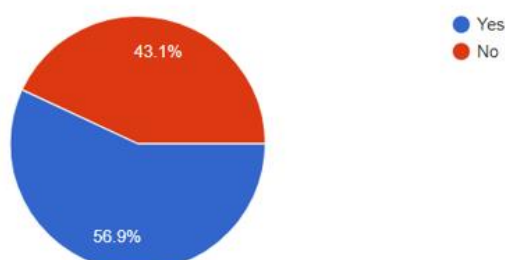
How old are you?



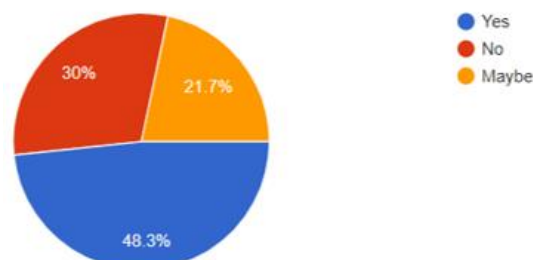
Where do you live?



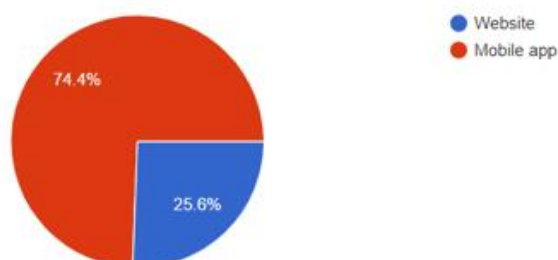
Do you have hobbies that you have no one to share them with?



Are you interested in a platform that will allow you to find partners for your hobbies?



If you are interested in the platform, would you prefer it to be on a website or mobile app?



Based on the results, we have reached several conclusions:

- The problem is prevalent among a high percentage of respondents.
- It is evident that there is a demand for a platform that will solve the problem.
- The absolute majority answered that they would prefer a mobile app platform over a website.

Project Review & Process Description

Our solution:

We offer an Android app to solve the problem. The application will include an option to create a meeting along with relevant details (purpose, number of participants, location and time). The users of the application will have the option to register for the meeting and confirm arrival. In addition, it will be possible to conduct a live chat between the participants of the meeting.

Each user of the application will have basic identifying information (name, age, area of residence, photo and interests). Moreover, there will be an option to report an untrustworthy user, these reports will be a measure of reliability.

The interface will contain two main types of events: physical and virtual.

The physical event interface will be displayed on a map. Meetings that take place in the user's area will be displayed on the map. The virtual event interface will be displayed as a list sorted by the hobbies the user has chosen.

When a new meeting that matches user's hobby is created, a notification will be sent to him with an option to join the meeting.

Agile Development

The development methodology that will guide us in the development process of the project is Agile. Agile project can be delivered quickly and offers greater flexibility. Considering that as students we have a lot of tasks throughout the semester, we decided that this is the most suitable method for us.

Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

Mobile Development:

We chose to develop our idea as a mobile application for several reasons:

- **Ease of access** – for the user, clicking on app is far quicker and easier than accessing to website, even if they've bookmarked it.
- **Communication** – it's easier to interact with users and prospects with an app including the immediacy of being able to attract their attention and maybe prompt action through, for example, push notifications.
- **Staying “front of mind” with users** – The application icon on the mobile home screen will remind the user his need of the application.
- **GPS** – The platform is designed to be implemented on a GPS-based map. Naturally, mobile development would be more appropriate.



Android:

Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets.

Android development is a form of software engineering dedicated specifically to creating applications for devices that run on the Android platform.

Client – Server Model:

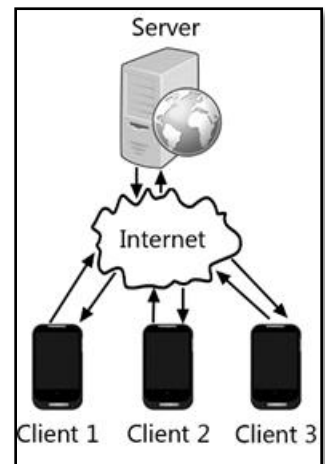
For our platform, we will require a constant connection with a server that will contains a large amount of information. For example: data about the user - identifying details, interests, etc. Therefore, we see a strong need to use a client-server model.

What is a Client – Server Model?

A client and server networking model is a model in which computers such as servers provide the network services to the other computers such as clients to perform a user based tasks. This model is known as client-server networking model.

The application programs using the client-server model should follow the given below strategies:

- An application program is known as a client program, running on the local machine that requests for a service from an application program known as a server program, running on the remote machine.
- A client program runs only when it requests for a service from the server while the server program runs all time as it does not know when its service is required.
- A server provides a service for many clients not just for a single client. Therefore, we can say that client-server follows the many-to-one relationship. Many clients can use the service of one server.



Mobile Cloud Computing:

For our platform, we will require the use of cloud services in order to get better performance and get accessible and flexible data storage. In addition, choosing a good cloud service will cover the issue of information security.



What is a Cloud Computing?

Cloud computing is the delivery of computing services such as software, databases, servers and networking, over the internet. This means end users are able to access software and applications from wherever they are.

Project Goals:

- The overarching goal of the project is to develop a friendly and easy-to-use application that will allow users to schedule meetings based on common hobbies.
- The application will allow the user to register with identification details and add personal hobbies under categories.
- The system will allow people with common hobbies to create social events that will help them develop their hobbies and enrich their social circle.
- The system will provide a convenient infrastructure for virtual meetings on the relevant platforms.
- The interface of the physical meetings in the application, will be based on a map. Each user will be able to create an event that includes the purpose of the meeting, time and location (which will be displayed on the map).
- The system will utilize a secure and efficient cloud database to store and manage the app's data. By leveraging the capabilities of a cloud-based solution, we can ensure the reliability, scalability, and security of the data storage infrastructure.

Unique Features:

- **Notifications** – When an event is created, notifications will be pushed for users whose interests match the event's purpose and their geographic location (if physics event) is relevant to the event.
- **Live Chat** – Users registered for the event will be able to chat via live chat.
- **Report** – The application will offer the possibility to report abusive users who violated ethical rules defined during the registration process.
- **Virtual Events** – There will be an option to schedule events for hobbies that happening virtually and do not require a physical meeting. For example: online games, zoom meetings, etc.
- **Adding new friends** – The app will allow the manager's event to invite friends to a specific event.
- **Event Summary** – After the event, it will be possible to add comments and intellectual achievements from the meeting.

Process:

To ensure an optimal development process for our application, we have established a systematic workflow that encompasses various milestones. These milestones guide us through the different stages of development, enabling us to create a robust and user-centric solution. The key steps in our development process are as follows:

- **Problem Analysis:** We initiated the development process by conducting a thorough analysis of the problem at hand. This involved identifying the root causes of the problem, understanding its scope and impact, and determining the best approach to address it effectively.
- **Target Audience:** By conducting surveys and gathering relevant data, we gained valuable insights into the target audience for our application. This research helped us understand the diverse population groups affected by the problem and confirmed the need for a solution in the market.
- **System Requirements:** With a clear understanding of the problem and its origins, we proceeded to define the system requirements in detail. This involved identifying both the functional and non-functional aspects that the system should encompass, based on the insights gathered during the previous stages.
- **Offering Solutions:** We generated multiple potential ideas to solve the identified problem. Each proposed solution was evaluated based on predefined criteria derived from surveys and research conducted earlier. By analyzing the advantages and disadvantages of each idea, we assessed their suitability and effectiveness.
- **Selected Solution:** After careful evaluation, we determined that a map-based Android application offered the optimal solution to meet our defined requirements. This solution provides a user-friendly interface and leverages location-based features to facilitate efficient and personalized event management.
- **Development Tools:** We selected the necessary tools and technologies to support the development process. Android Studio was chosen as the development environment, utilizing the Java programming language for coding and XML for designing user interfaces. Firebase, a real-time database platform, was identified as the ideal cloud service provider to support the application's functionality.
- **Development Methodology:** We adopted the Agile development methodology due to its flexibility and ability to adapt to evolving project requirements. The iterative and

incremental nature of Agile development allows for efficient task management and fosters rapid development and responsiveness to user feedback.

By following this well-defined development process, we aim to create a robust and user-centric Android application that effectively addresses the identified problem. Our choice of appropriate tools, methodologies, and technologies ensures an efficient and streamlined development journey, resulting in a high-quality and engaging solution for our target audience.

Engineering Challenges & Solutions:

- **Challenge:** Google Maps Integration

One of the challenges we encountered during our project was integrating Google Maps into our application. Although Google Maps is a powerful tool for mapping and location services, it presented certain difficulties that required careful consideration and problem-solving.

Solution: Overcoming the challenge of integrating Google Maps involved studying the API documentation, managing authentication and API key, handling rate limiting and usage quotas, implementing error handling mechanisms, designing a seamless user experience, and staying updated with maintenance and updates.

- **Challenge:** Identifying Unreliable Users

The application is based on physical encounters between people. It is our responsibility to develop an algorithm that will monitor user's safety and keep untrustworthy users away. The system will include an algorithm based on smart use of user reports and user status in order to alert the administrator about unreliable users.

Solution: We have implemented a system that responds to user reports by taking appropriate action, including potential account suspension. Reported users are required to contact app support for further resolution and clarification regarding the reported incident.

- **Challenge:** Personalized Event Recommendations Based on User Hobbies

One of the challenges we faced in our application was providing personalized event recommendations to users based on their chosen hobbies during registration. We

wanted to create an algorithm that would analyze user preferences and promote events that align with their hobbies, enhancing their overall experience.

Solution: To tackle this challenge, we implemented the following strategies:

Hobby Selection during Registration: During the user registration process, we included a section where users could choose their hobbies from a predefined list. This allowed us to gather valuable data about their interests and preferences.

Hobbies-Based Event Highlighting: On the map interface, we implemented a feature that highlighted events directly related to the user's selected hobbies. By filtering and displaying events that align with their interests, we ensured that users would easily discover relevant activities in their vicinity.

Sorting Virtual Events by Hobbies: For virtual events, we implemented a sorting mechanism that prioritizes events according to the user's hobbies. This meant that virtual events associated with their selected hobbies would be displayed prominently, making it more likely for users to participate in activities tailored to their interests.

Hobby-Based Event Invitations: After an event is created, event managers have the ability to send invitations to users who have matching hobbies. This targeted approach ensures that users are notified about events that are specifically relevant to their interests, increasing the chances of engagement and participation.

Evaluation & Verification:

In order to check that our system is working properly, we will perform two types of tests:

- Unit Testing – A unit test is a way of testing a unit - the smallest piece of code that can be logically isolated in a system.
- Functional Testing – Functional testing is a type of testing that seeks to establish whether each application feature works as per the software requirements. Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user's expectations.

We will perform the 2 types of tests in the Android Studio environment. For the unit testing we will use the Junit 5 framework and for the functional testing we will use the Espresso testing framework.

Unit Testing

Registration		
Number	Test Subject	Expected Result
1	Registration succeeded	User added to the users table in the DB with appropriate details
2	Registration failed (as a result of invalid fields)	User has not been added to the users table in the DB
3	Registration failed (as a result of system failure)	User has not been added to the users table in the DB - throw an exception

Login		
Number	Test Subject	Expected Result
1	Login succeeded	Username and Password exists in the DB
2	Login failed (as a result of invalid fields)	Username and Password doesn't exist in the DB
3	Login failed (as a result of system failure)	throw an exception

Create an Event		
Number	Test Subject	Expected Result
1	Event created successfully	Event added to the event table in the DB with appropriate details
2	Event creation failed (as a result of invalid fields)	Event has not been added to the events table in the DB
3	Event creation failed (as a result of system failure)	Event has not been added to the events table in the DB - throw an exception

Join to Event		
Number	Test Subject	Expected Result
1	The user joined the event successfully	User has been added to the event users list and the number of participants increased by one
2	Joining failed (as a result of system failure)	User has not been added to the event users list – throw an exception

Menu		
Number	Test Subject	Expected Result
1	Select Logout	User status have been changed to false in the DB
2	Select My Profile and edit details	Users details have been updated in the DB

Add Review		
Number	Test Subject	Expected Result
1	The user add review successfully	Review has been added to the event reviews list in the DB
2	Failed to add review (as a result of invalid fields)	Review has not been added to the event reviews list in the DB
3	Failed to add review (as a result of system failure)	Review has not been added to the event reviews list in the DB – throw an exception

Add Hobby (as Administrator)		
Number	Test Subject	Expected Result
1	The administrator adds new category successfully	Category has been added to the Categories table in the DB
2	The administrator adds new hobby successfully	Hobby has been added to the Hobbies table in the DB
3	Failed to add new category (as a result of invalid fields)	Category has not been added to the Categories table in the DB
4	Failed to add new category (as a result of system failure)	Category has not been added to the Categories table in the DB – throw an exception
5	Failed to add new hobby (as a result of invalid fields)	Hobby has not been added to the Hobbies table in the DB
6	Failed to add new hobby (as a result of system failure)	Hobby has not been added to the Hobbies table in the DB – throw an exception

Functional Testing

Registration		
Number	Test Subject	Expected Result
1	Enter a username that already exists in the system	System Displays: "The username already exists, try another"
2	Enter a password shorter than 8 characters	System Displays: "The password is too short, please enter at least 8 characters"
3	Enter an email that already exists in the system	System Displays: "The email address is already being used"
4	Enter invalid email address	System Displays: "Invalid email address"
5	According to the date of birth entered, the user is under the age of 16	System Displays: "The use of the application is only allowed from the age of 16 and above"
6	There is an empty field	System Displays: "Please fill all required fields"
7	The registration has been successfully completed	System switches to user home page screen

Login		
Number	Test Subject	Expected Result
1	There is an empty field	System Displays: "Please fill all required fields"
2	Enter a password shorter than 8 characters	System Displays: "The password invalid"
3	Enter a username that doesn't exist in DB	System Displays: "Username or Password is incorrect"
4	Enter an incorrect password	System Displays: "Username or Password is incorrect"
5	Successful login	System switches to user home page screen

Create an Event		
Number	Test Subject	Expected Result
1	Invalid time	System Displays: "Please enter valid time"
2	There is an empty field	System Displays: "Please fill all required fields"
3	No hobby selected	System Displays: "Please select the event hobby"
4	If the event is physical and no location selected	System Displays: "Please select the event location"
5	If the event is virtual and no link has been inserted	System Displays: "Please enter the event meeting link"
6	User selected physical event	System displays a map to choose a location to the meeting
7	User selected virtual event	System displays a text box to insert a link to the meeting
8	Event created successfully	System switches to the event screen

Add Review		
Number	Test Subject	Expected Result
1	The review field is empty	System Displays: "Please fill the review"
2	No rating selected	System Displays: "Please rate the meeting"
3	The review added successfully	System switches to the event reviews screen with the added review

Join to Event		
Number	Test Subject	Expected Result
1	The user joined the event successfully	System Displays: "Joined Successfully" System switches the "Join" button to "Unjoin" button

Menu		
Number	Test Subject	Expected Result
1	Select Logout	System switches to the login screen
2	Select My Profile	System switches to the user profile screen
3	Select My Events	System switches to the user events screen

Results & Conclusion:

The development and implementation of the Co-Bie application has yielded significant results and demonstrated the feasibility and effectiveness of connecting individuals with shared hobbies. Through extensive research and meticulous planning, we have successfully created a convenient and engaging platform for users to connect, share, and participate in activities related to their hobbies.

Throughout the development process, we conducted thorough research in the relevant field, including surveys to gather insights from our target audience. This research provided valuable input and helped shape the key features and functionality of the Co-Bie application. By leveraging the knowledge and preferences of our users, we were able to create a tailored experience that meets their needs and expectations.

The engineering challenges that arose during development were effectively addressed through careful problem-solving and collaboration. By anticipating potential issues and employing appropriate solutions, we ensured the smooth functioning and robustness of the application. The incorporation of visual diagrams, including use-case, class, and activity diagrams, proved instrumental in enhancing the understanding of the application's functionality and interactions. These visual representations facilitated the design process, resulting in a well-crafted and intuitive user interface. By leveraging visual diagrams, the development team ensured that the Co-Bie application offers a seamless and user-friendly experience, enabling users to navigate and interact with the platform effortlessly.

To ensure the quality and user satisfaction of the Co-Bie application, we have devised a comprehensive testing plan. This plan includes various testing methodologies to identify and address any bugs or usability issues. Through rigorous testing, we aim to provide a seamless and enjoyable user experience, maximizing the application's impact and value.

In conclusion, the Co-Bie application has successfully addressed the need for individuals to connect with like-minded hobbyists. It offers a convenient and effective platform for users to share their expertise, seek advice, and organize meetups and events. The project's results highlight the dedication and effort put into developing a user-centered application that promotes engagement and fosters meaningful connections among individuals with shared hobbies. We are confident that Co-Bie will provide a valuable resource for hobbyists, enabling them to enhance their hobby-related experiences and create lasting connections.

User Documentation

General Description:

The Co-Bie application is an Android app designed to connect individuals with shared hobbies. It addresses the challenge of finding like-minded enthusiasts by providing a platform where users can connect, share, and organize hobby-related meetups. The app allows users to create meetings, specify relevant details, and enables others to register and confirm their attendance. It also features a live chat functionality for participants to communicate.

The application follows a client-server model, with the Android client running on devices and utilizing Firebase as the cloud-based backend. This architecture ensures scalability, reliability, and real-time synchronization of data. Users have profiles containing basic information such as name, age, residence, photo, and interests. The app supports physical and virtual events, displayed on a map or in a list sorted by hobbies.

The target audience for Co-Bie is hobbyists seeking connections with others who share their interests. The app fosters meaningful connections and empowers users to enhance their hobby-related experiences. By providing personalized event recommendations, live chat functionality, and notifications for relevant meetings, Co-Bie creates a dynamic and interactive experience. The app's user-friendly interface and reliable infrastructure contribute to a seamless and enjoyable user experience, facilitating lasting connections within hobby-based communities.

User Interface:

- Our Logo:



Fig 1: Application Logo

- Login Screen

Login screen for registered users. For unregistered users there is an option to sign-up.

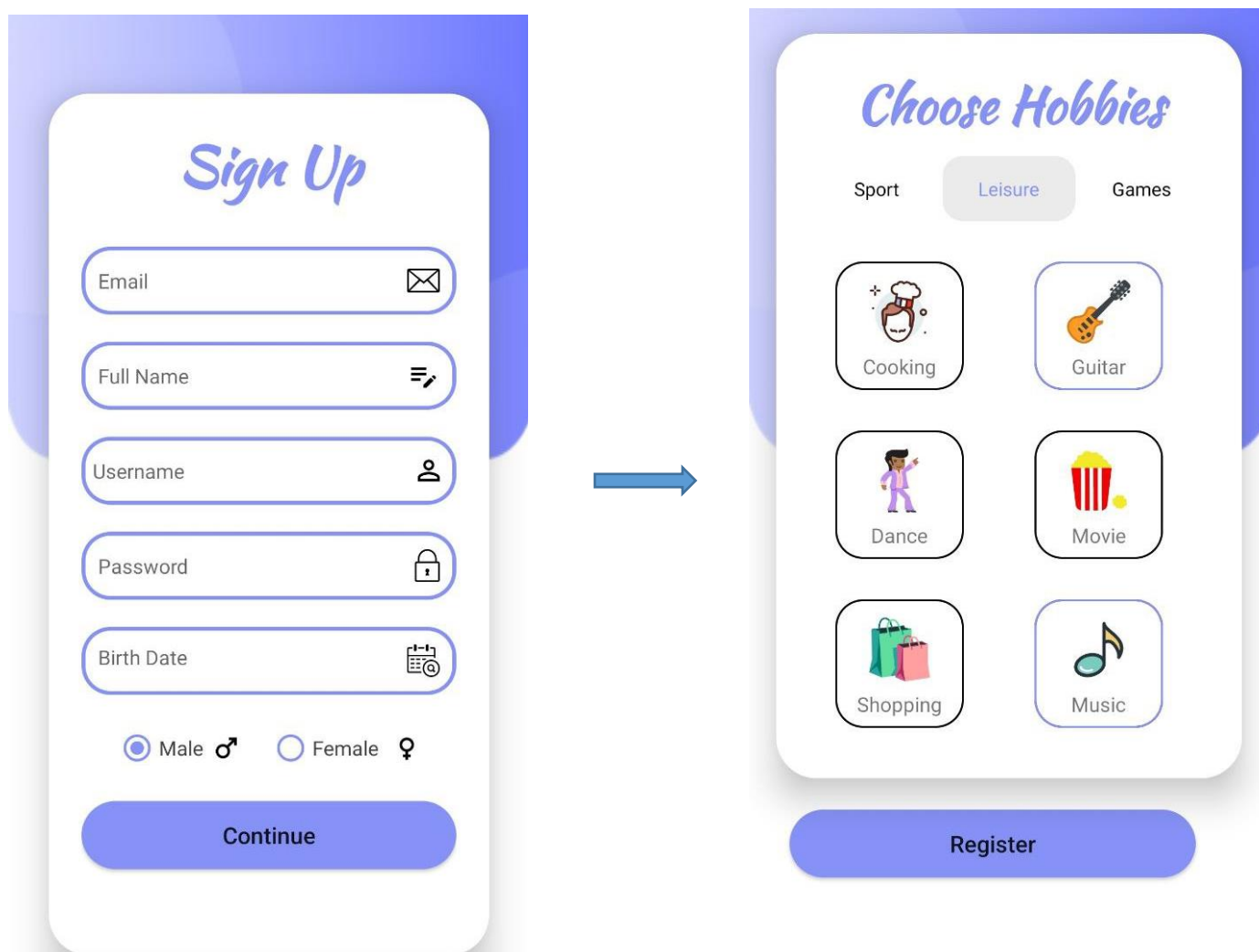
The login screen has a blue gradient background. A white rounded rectangle in the center contains the 'Co-Bie' logo in blue script. Below the logo are two input fields: 'Email' with an envelope icon and 'Password' with a lock icon. A blue 'Login' button is below the fields. At the bottom of the white rectangle are two links: 'Not yet registered? Sign Up Now' and 'Forgot password?'.

Fig 2: Login Screen

- Registration Screen

When a user clicks on "Sign-Up", he will have to enter identifying details, username and password for future login.

After entering the details successfully, he will have to select his hobbies from different categories (selection of hobbies is optional).



The diagram illustrates the registration process flow. It starts with the **Sign Up** screen, which contains the following fields and options:

- Email (with an envelope icon)
- Full Name (with a pencil icon)
- Username (with a person icon)
- Password (with a lock icon)
- Birth Date (with a calendar icon)
- Gender selection: ☒ Male ♂ and ☐ Female ♀
- Continue** button

An arrow points from the **Continue** button to the **Choose Hobbies** screen. The **Choose Hobbies** screen features:

- Category tabs: **Sport**, **Leisure** (selected), and **Games**.
- Hobby selection grid:
 - Cooking (with a chef hat icon)
 - Guitar (with a guitar icon)
 - Dance (with a dancer icon)
 - Movie (with a popcorn icon)
 - Shopping (with a shopping bag icon)
 - Music (with a musical note icon)
- Register** button

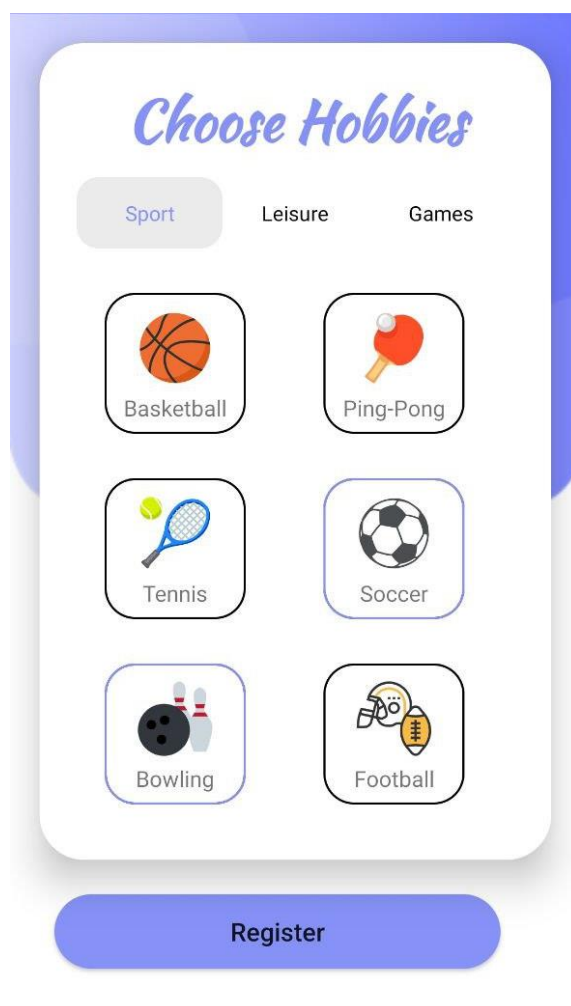
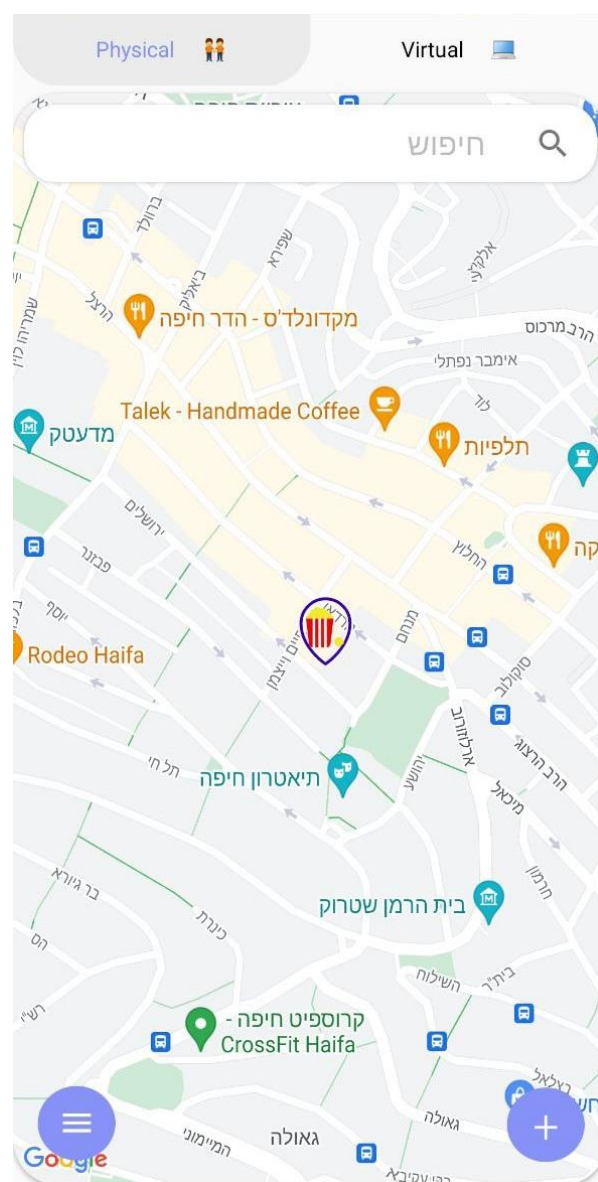
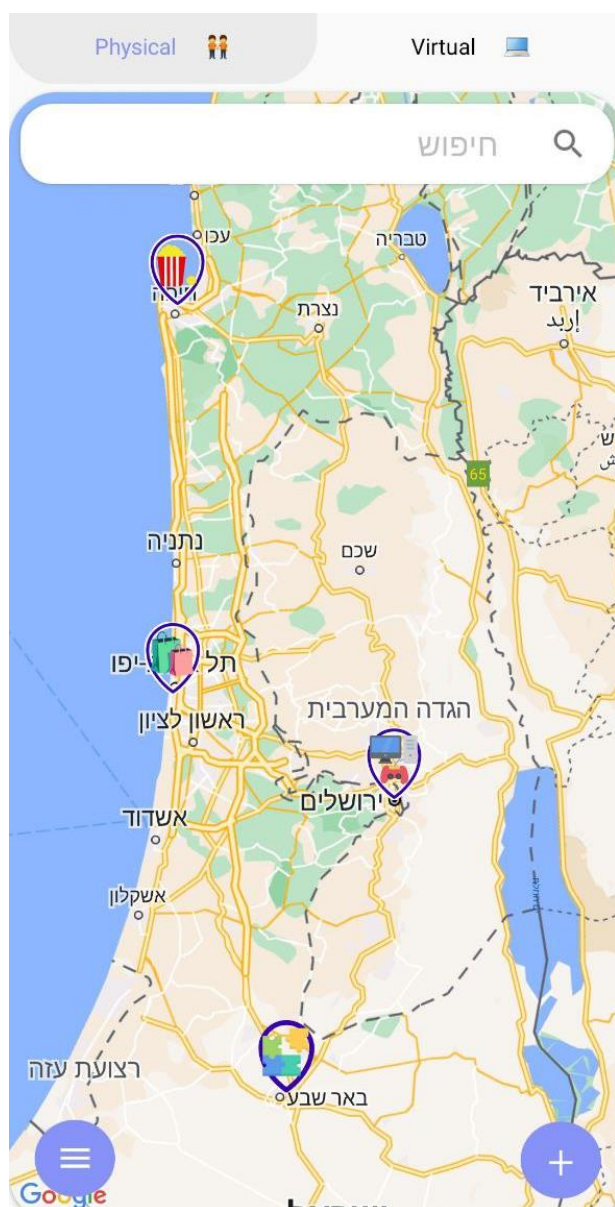


Fig 3: Registration Screens

- Home Screen:

The home page will contain the various events that exist. There will be a division into physical events and virtual events. The user will be able to click on the event he is interested in. The physical events will appear as locations on a map and the virtual events will appear in a list.



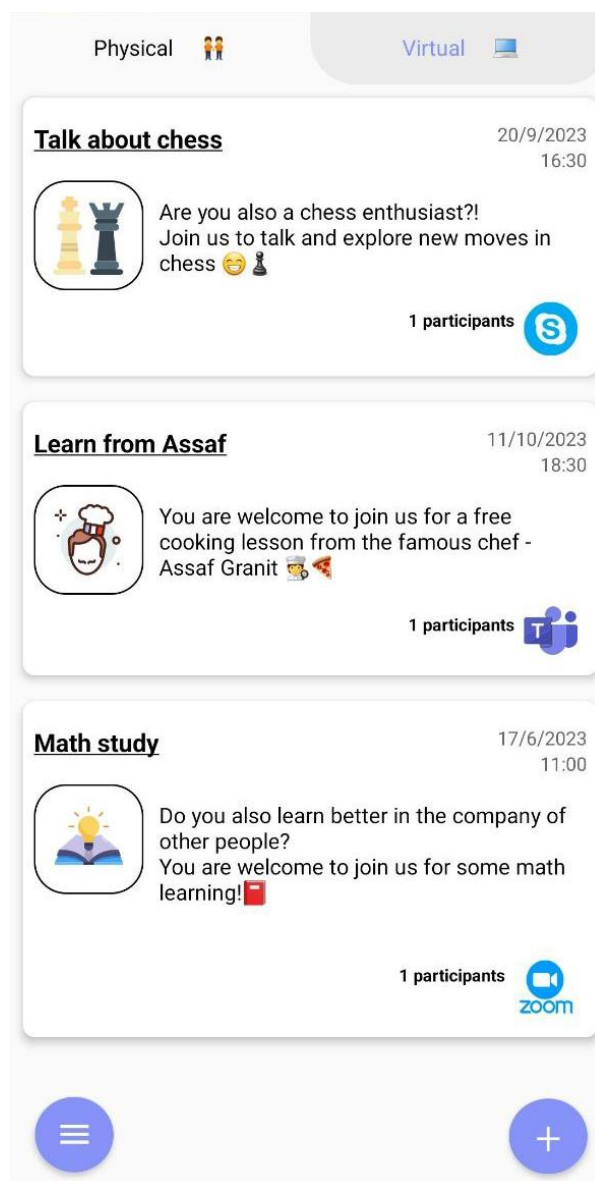


Fig 4: User Home Screens

- Join Event Screen:

When a user clicks on a certain event, a screen will be displayed with the event details and the user will be able to join the event by clicking "Join" button.

If this is a physical event, the location will be displayed.

If this is a virtual event, a link to the meeting will be attached.

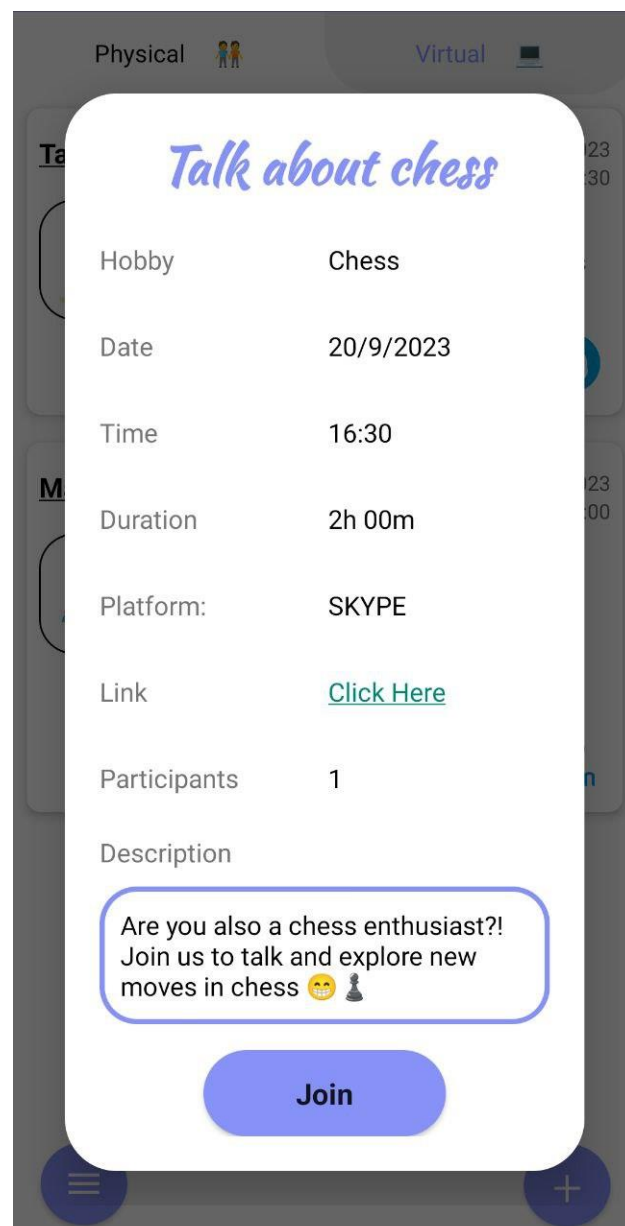
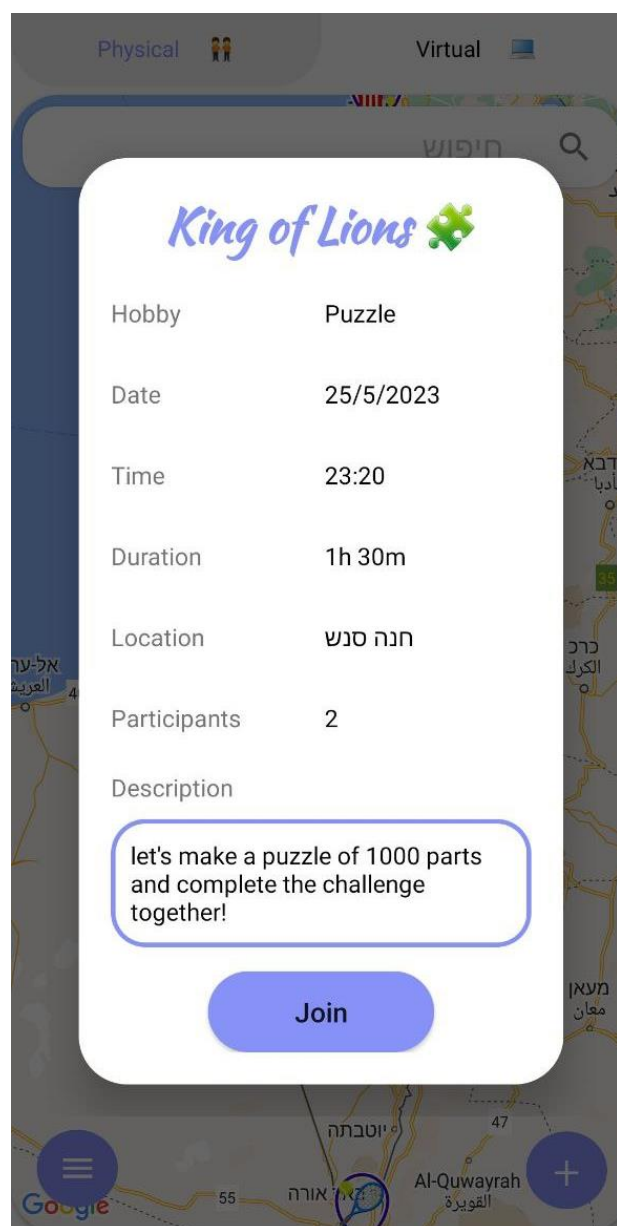


Fig 5: Join Event Screens

- Create Event Screen:

When clicking the plus button on the home screen (top right) the user will have the option to create an event.

The user will be required to enter the details of the event.

If the event is physical, the user will be asked to provide a location on a map.

If the event is virtual, the user will be required to provide the name of the platform and a link to the meeting.

If the event is private, the user will require providing the invited users.

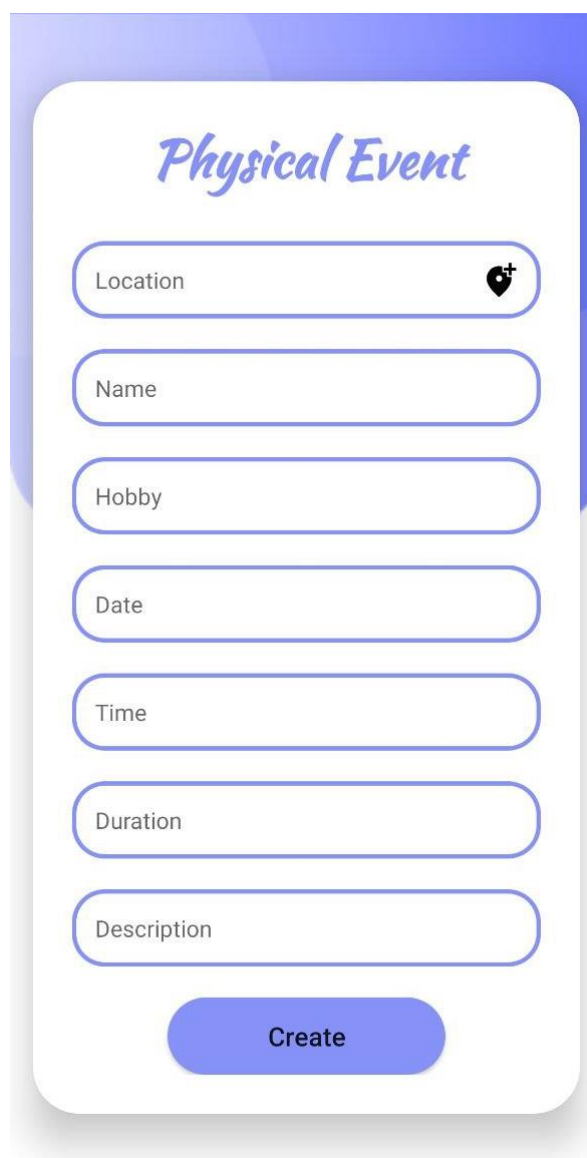
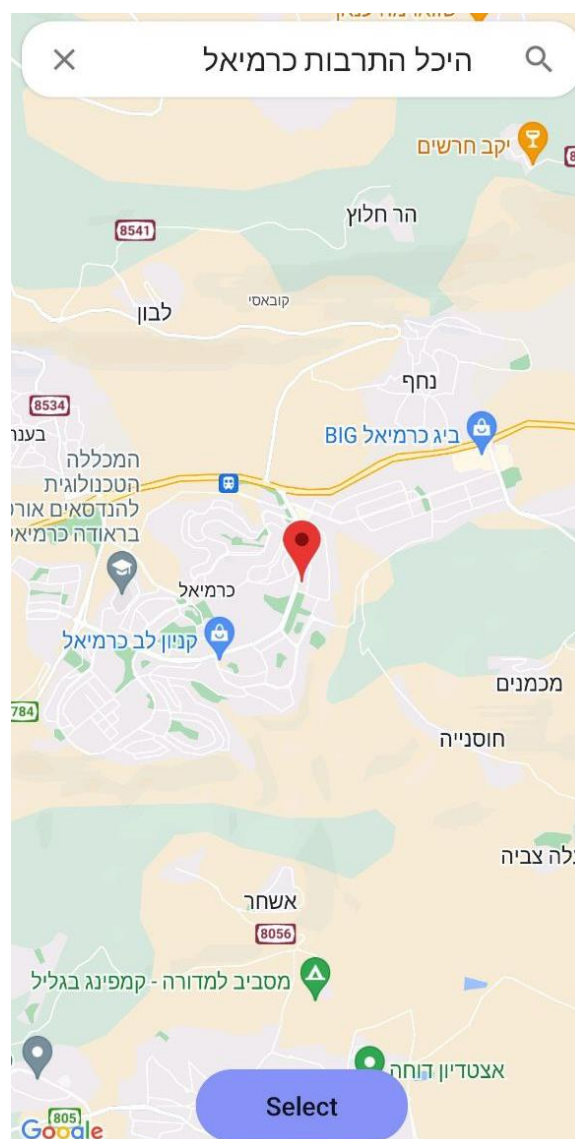



Fig 6: Create Event Screen

- Menu Screen:

On the home screen, the user will have the option to open the application menu (top left). The menu will contain the following options:

My Profile – option to enter the user's personal page, edit details and more.

My Events – option for the user to see the events he registered for. Also, events that have already passed will be displayed in order to watch/add reviews and intellectual achievements.

Logout – disconnection from the existing user.

About Us - information about the use of the application.

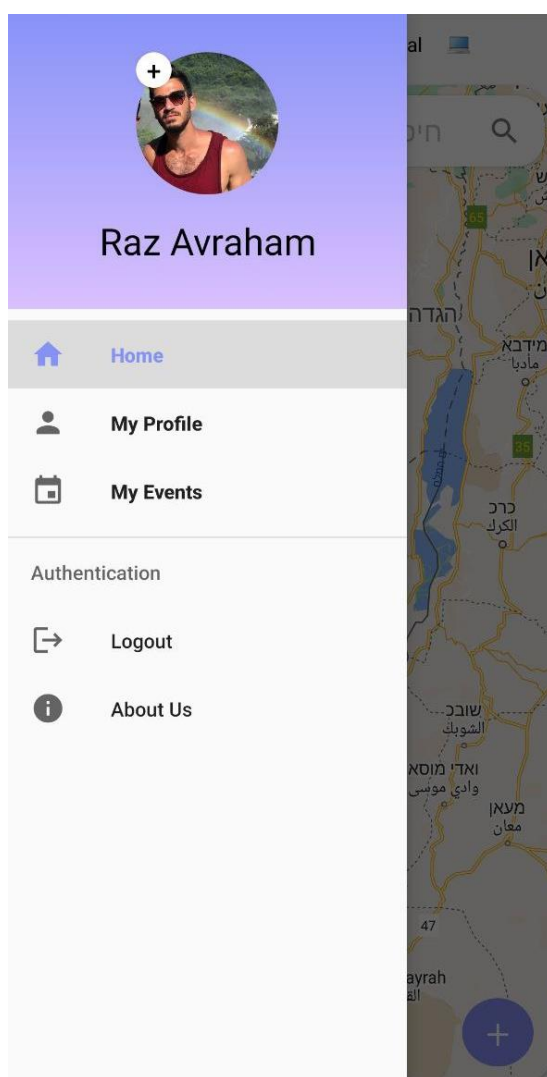
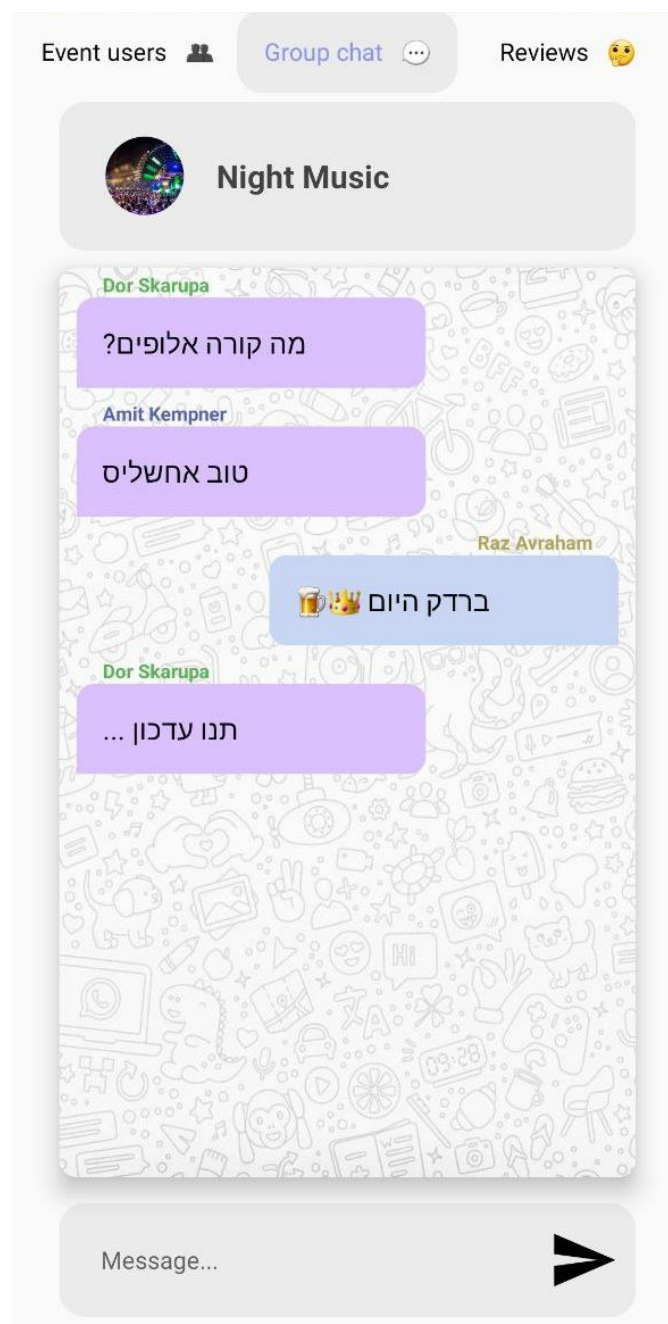


Fig 7: Menu Screen

- Event Screen:

The screen consists of three tabs that provide comprehensive event details, user participants, and a group chat functionality, along with an additional reviews tab for user feedback.



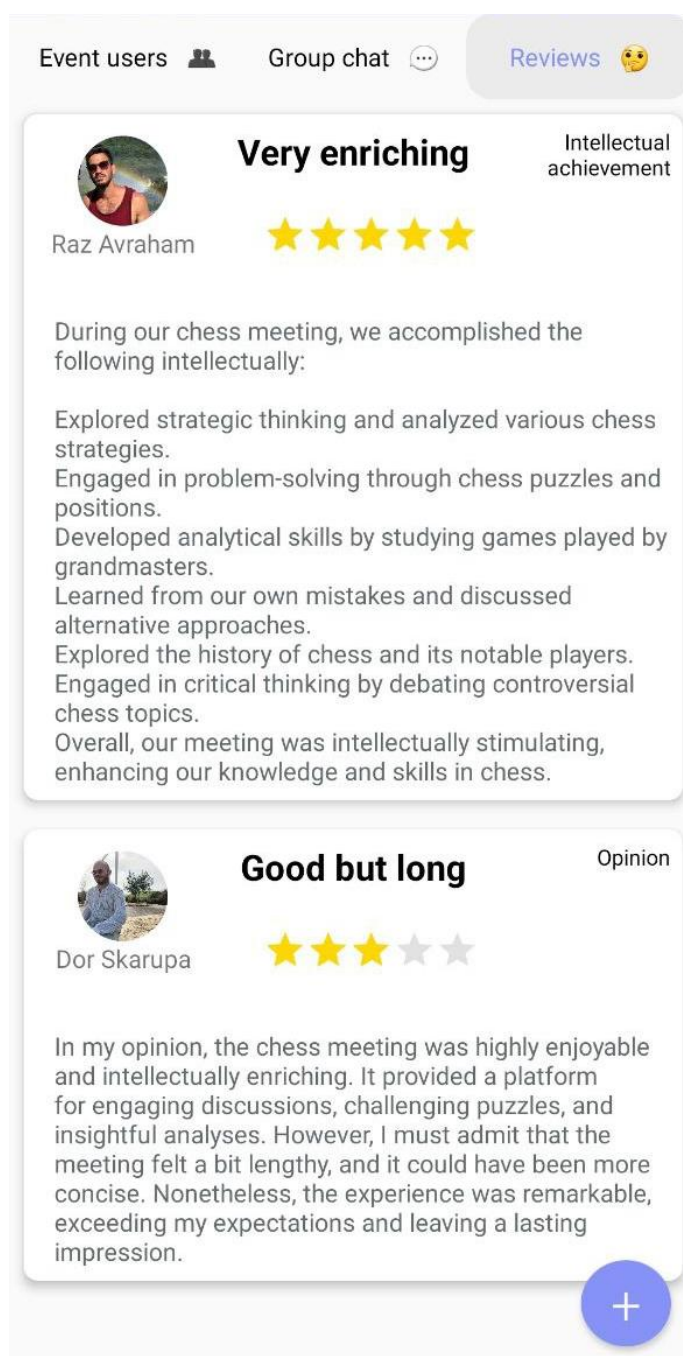


Fig 8: Event Screen

Maintenance Guide

Product:

Software Architecture Diagram:

Application

Services

Data Base

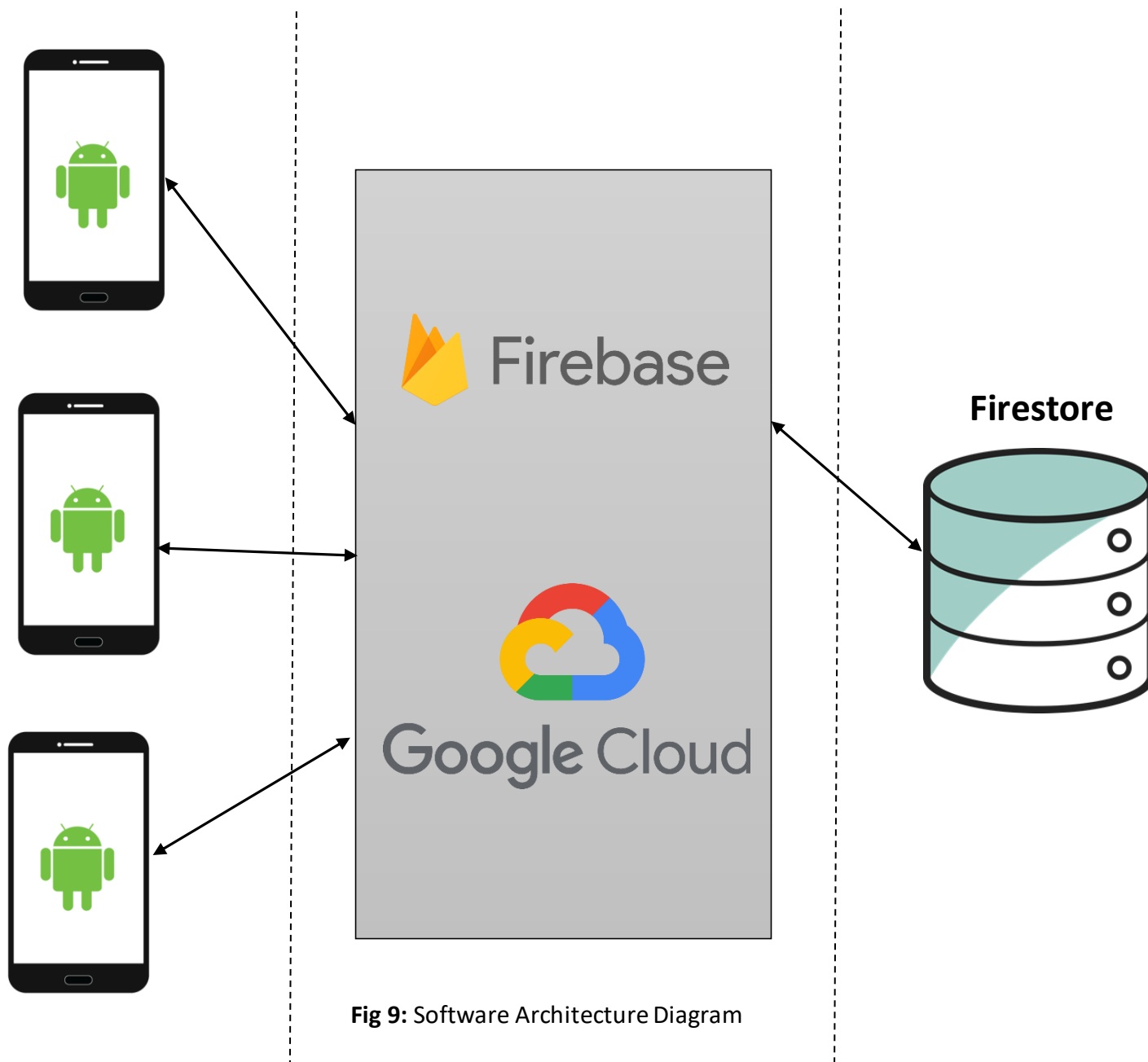


Fig 9: Software Architecture Diagram

Architecture Overview:

The architecture of your Android application follows a client-server model, with the client side running on Android devices and the server side utilizing Firebase for cloud-based services and data storage. The combination of Android and Firebase provides a robust and scalable architecture for your application.

Components of the Architecture:

- **Android Client:**
The Android client component consists of the user interface (UI) and the application logic running on Android devices. It handles the presentation layer and user interactions. The Android client communicates with Firebase to retrieve and send data, authenticate users, and handle real-time updates.
- **Firebase Backend:**
Firebase serves as the backend for your Android application, offering a range of cloud-based services that streamline the development process and provide scalability, reliability, and security. The key Firebase services that you utilize in your architecture include:
- **Real-time Database:**
Firebase Real-time Database is a cloud-hosted NoSQL database that provides real-time synchronization capabilities. It allows you to store and sync data in real-time across connected clients. Your application can read and write data to the database, enabling real-time updates for events, user profiles, and other relevant information.
- **Authentication:**
Firebase Authentication offers a secure and easy-to-implement user authentication system. It provides various authentication methods like email/password, social logins (e.g., Google, Facebook), and anonymous authentication. This ensures that users can securely register, log in, and access the application's features.
- **Cloud Messaging:**
Firebase Cloud Messaging (FCM) enables sending push notifications to Android devices. With FCM, you can send notifications to users for various events such as new meeting invitations, updates, or general announcements. This helps in engaging users and keeping them informed about relevant activities.

- **Cloud Storage:**
Firebase Cloud Storage allows you to store and serve user-generated content, such as profile pictures or event images. It provides secure and scalable storage for media files associated with the application.
- **Communication:**
The Android client communicates with Firebase through various APIs and SDKs provided by Firebase. These APIs enable data retrieval, data storage, user authentication, and handling real-time updates. The communication between the Android client and Firebase occurs over secure network connections, ensuring the confidentiality and integrity of data transmission.

Use-Case Diagram:

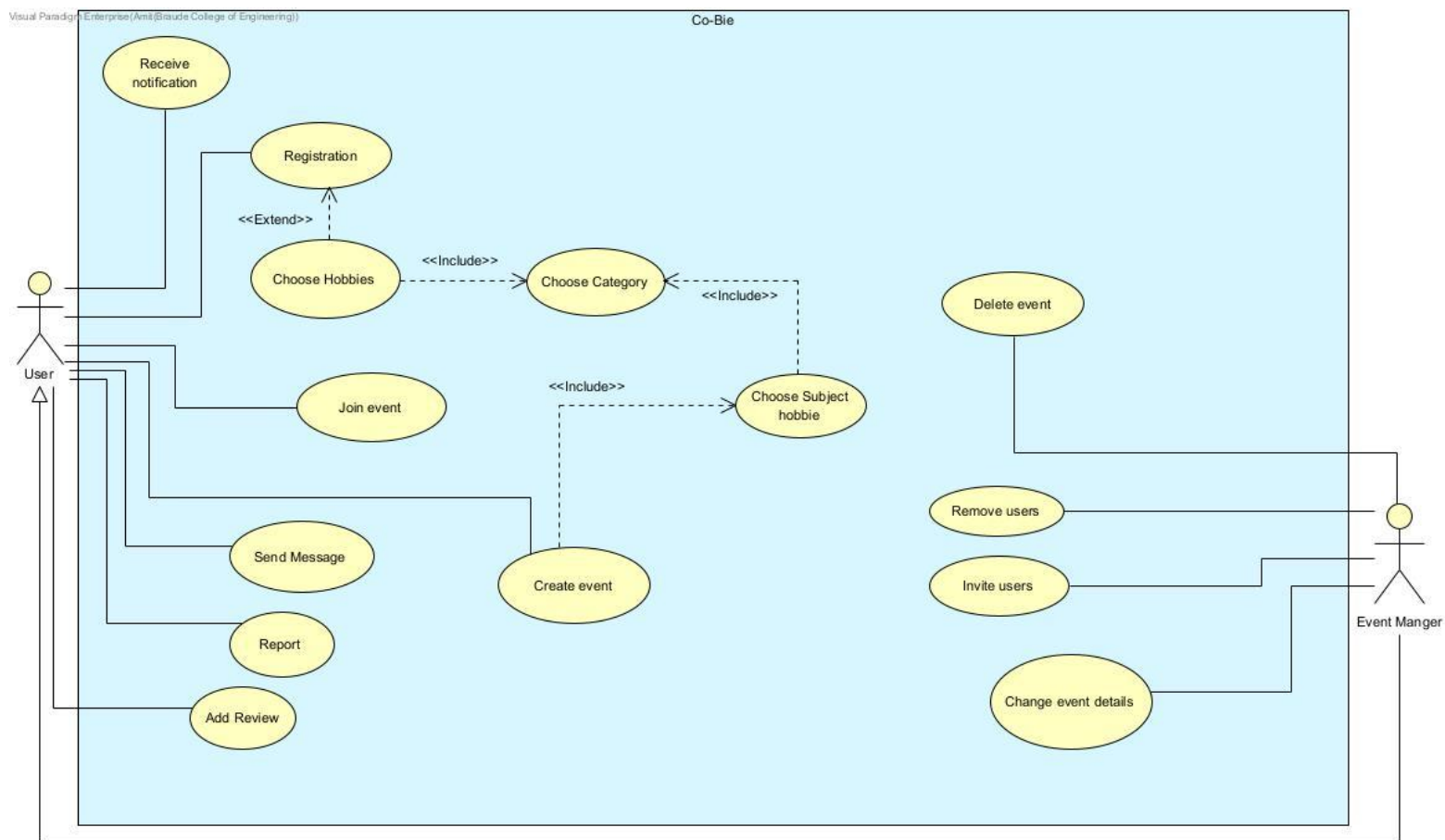


Fig 10: Use Case Diagram

Class Diagram:

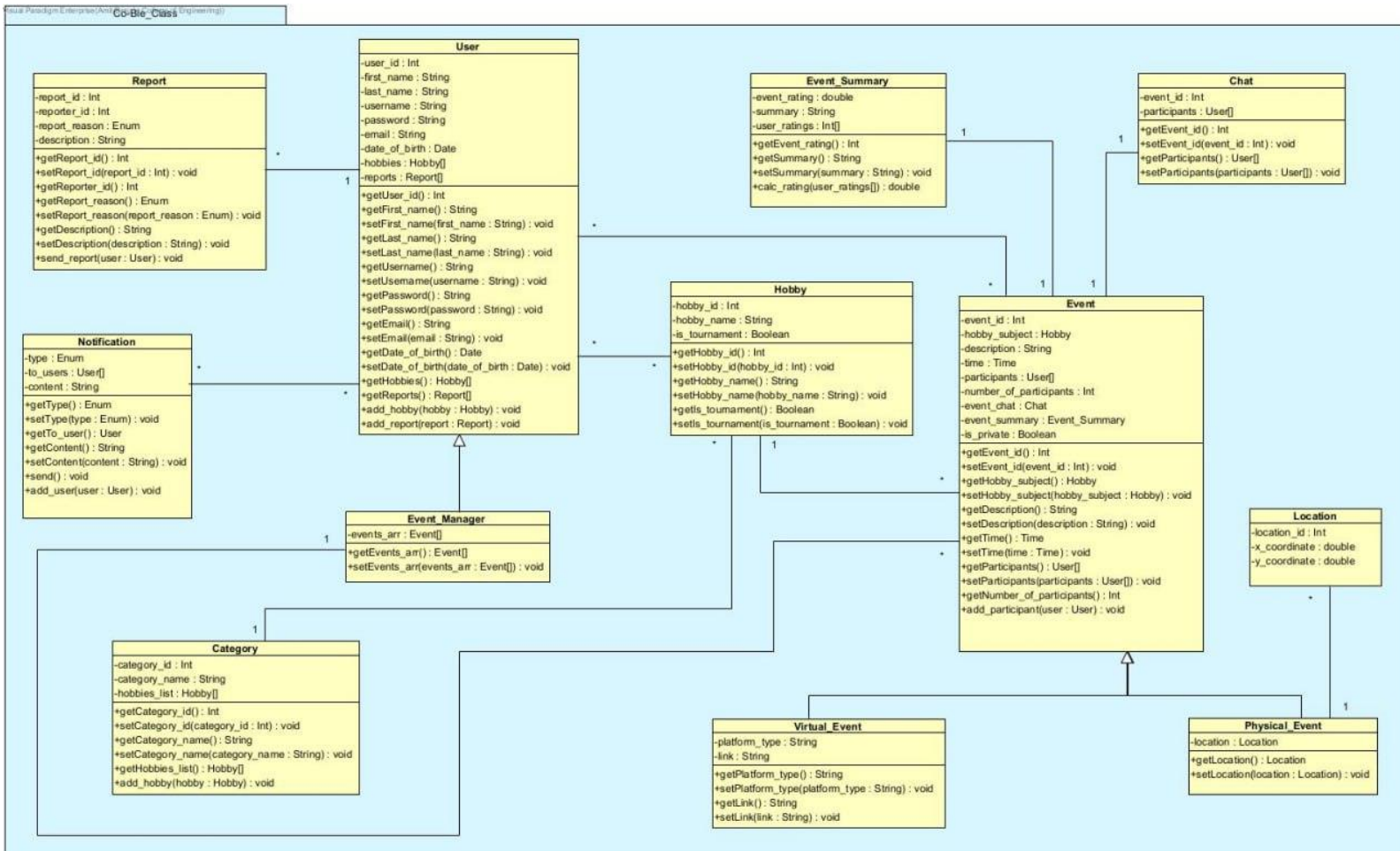


Fig 11: Class Diagram

Package Diagram:

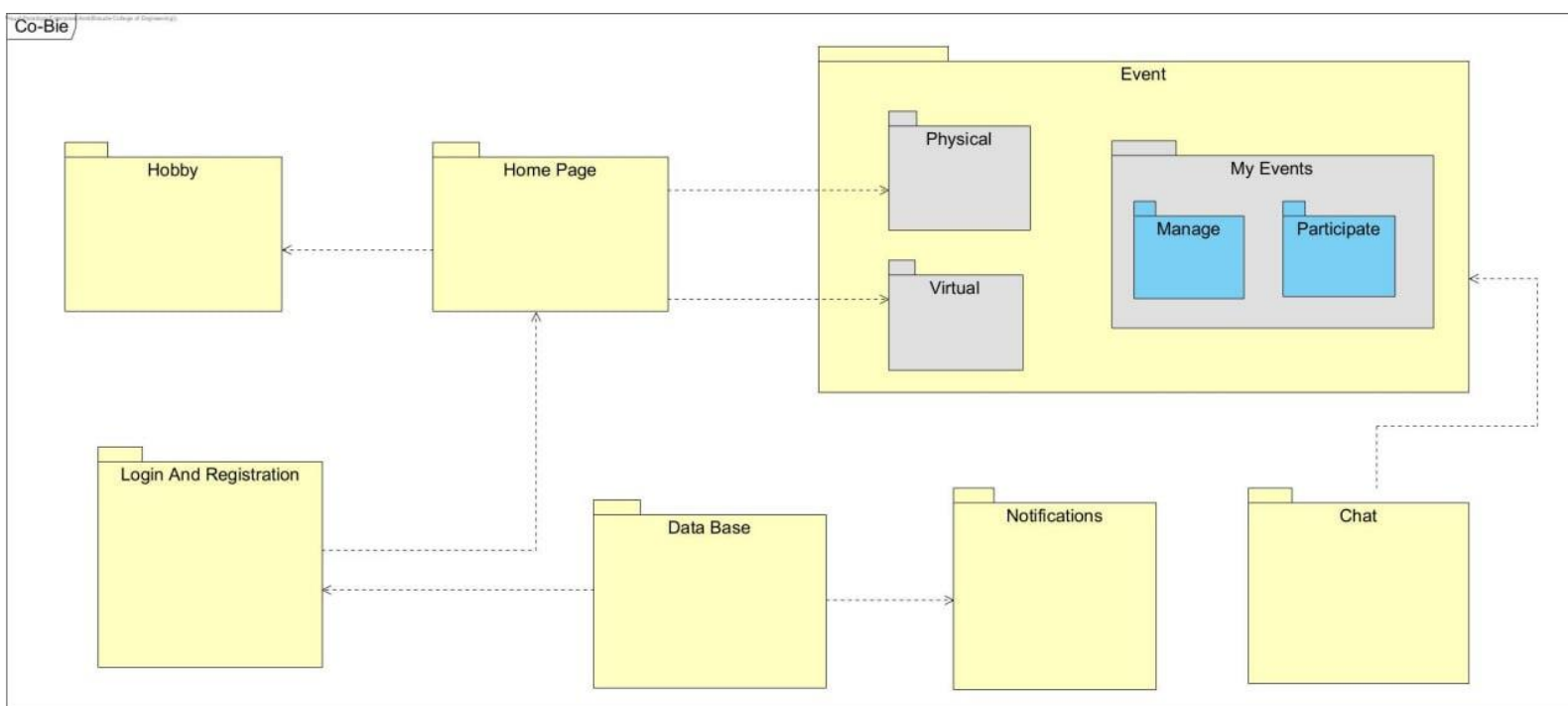


Fig 12: Package Diagram

Activity Diagram:

Create event

1. The user login to the system.
2. The system displays the home screen.
3. The user selects the menu option.
4. The system displays the menu options.
5. The user chooses "Create Event" option.
6. The system displays the event details configuration screen.
7. The user chooses the event category.
8. The system displays all the hobbies that are under the selected category.
9. The user chooses the event hobby.
10. The user chooses the event time, description, public/private, physical/virtual.
11. If the user has selected a public event, pass to step 14.
12. If the user has selected a private event, the system will pop up a window to select invited users.
13. The user chooses the invited users.
14. If the user has selected a virtual event, pass to step 17.
15. If the user has selected a physical event, the system will pop up a window to select a location.
16. The user chooses the event location, pass to step 19.
17. The system will pop up a window to insert a link.
18. The user will enter a link to the meeting.
19. The user selects "Submit".
20. The system creates and display the event.

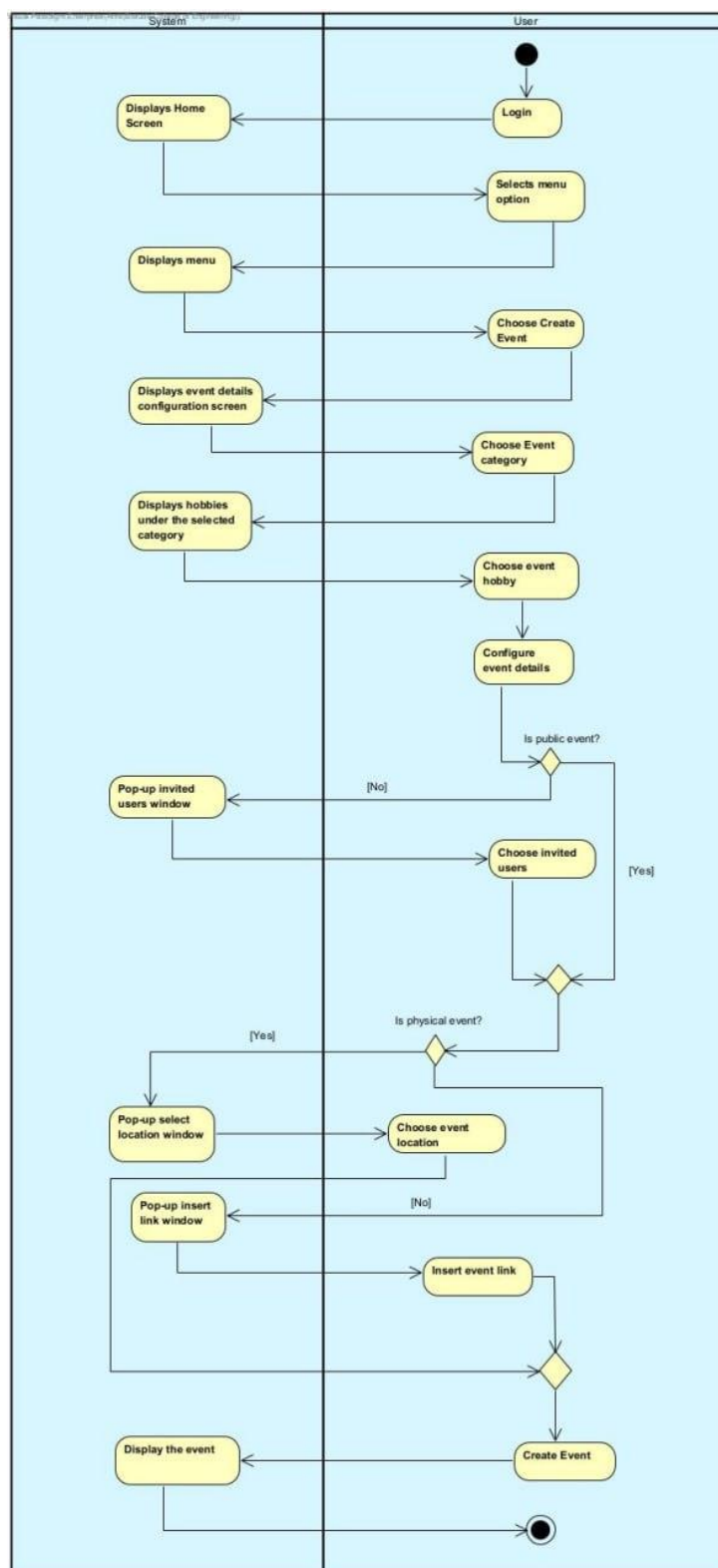


Fig 13: Activity Diagram – Create Event

Join event

1. The user login to the system.
2. The system displays the home screen.
3. The user selects “Virtual” / “Physical” tab.
4. If the customer selects “Virtual”, pass to step 6.
5. If the user selects “Physical”, the system displays a map with all existing physical public events.
6. If the user selects “Virtual”, the system displays a list with all existing virtual public events.
7. The user selects the event he wants to join.
8. The system will display the event details.
9. The user selects “Submit”.
10. The system pops up a message "You have successfully joined".

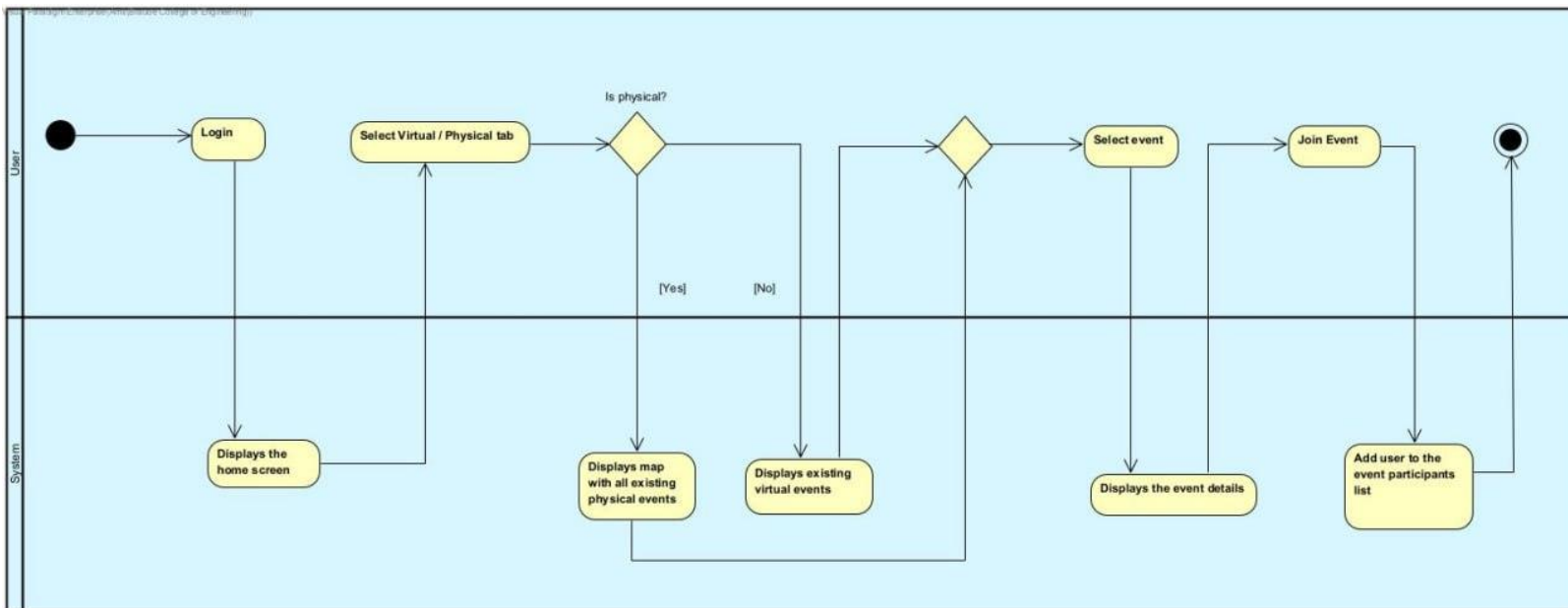


Fig 14: Activity Diagram – Join Event

Adding responses and intellectual achievements to event

1. The user login to the system.
2. The system displays the home screen.
3. The user selects the menu option.
4. The system displays the menu options.
5. The user chooses "My events" options.
6. The system displays all events that the user has joined.
7. The user selects the requested event.
8. If the event has ended, the system will display the event's comments page.
9. The user selects "Add comment".
10. The system pops up a text box.
11. The user enters his response in the text box provided.
12. The system adds the response to the response list of the event.

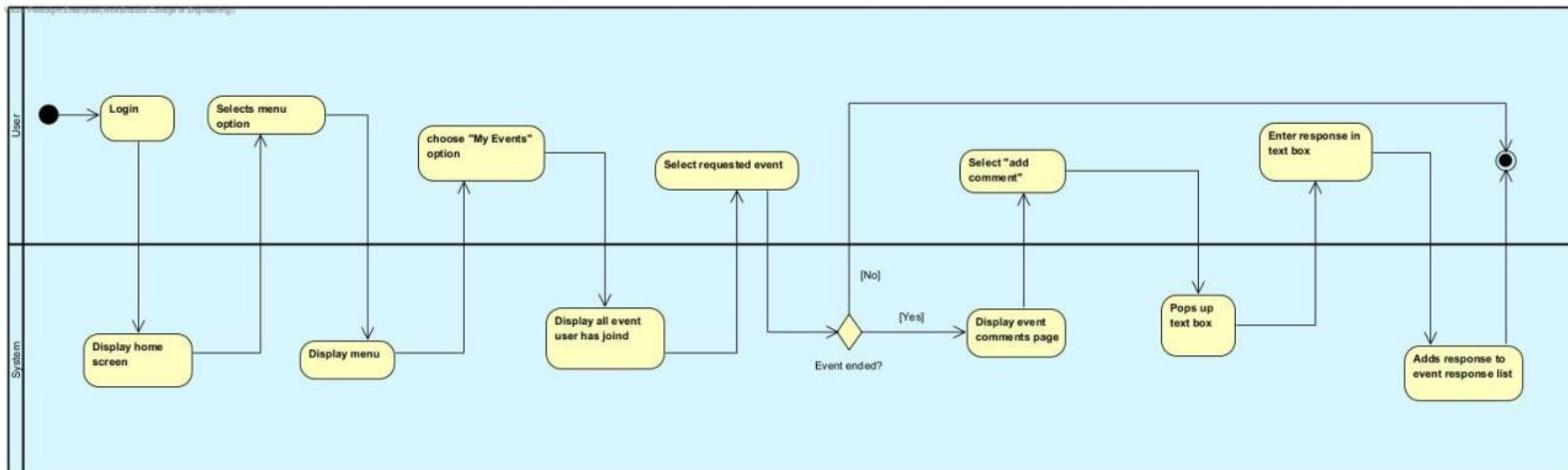


Fig 15: Activity Diagram – Add Review

Program Structure Description

Firestore Real-time Database:

We used real-time Firestore Database as the backend for our project, enabling seamless data synchronization and collaboration among our users. By integrating Firestore's real-time capabilities, multiple readers could access and interact with the book simultaneously, seeing updates in real-time as they occurred. This allowed us to create an engaging and interactive reading experience, where readers could leave comments, discuss chapters, and share their thoughts, all in real-time.

DB Scheme:

Table Name	Value	Type
Users	User_Uid	Firestore Authentication ID
	username	String
	password	String
	email	String
	full_name	String
	birth_date	String
	gender	String
	status	String
	profile_img	String
	hobbies_List	Hobby []

Table Name	Value	Type
Events	Physical_Events	Table
	Virtual_Events	Table

Table Name	Value	Type
Physical_Events	event_date	Date
	event_description	String
	reviews	Review []
	event_duration	Duration
	event_name	String
	event_time	Time
	event_type	String
	event_img	String
	event_Hobby	Hobby
	event_id	String
	location	Location
	manager_uid	String
	participants	User []

Table Name	Value	Type
Virtual_Events	event_date	Date
	reviews	Review []
	event_description	String
	event_duration	Duration
	event_name	String
	event_time	Time
	event_type	String
	event_img	String
	event_Hobby	Hobby
	event_id	String
	event_platform	String
	manager_uid	String
	participants	User []
	meeting_link	String

Table Name	Value	Type
Chats	Chat_Id	String
	Sender	String
	Receiver	String
	Message	String

System Requirements:

To run the Co-Bie application, the following requirements should be met:

- **Android Device:** The application is designed for Android devices, so you will need a compatible smartphone or tablet running the Android operating system.
- **Compatible Android Version:** Ensure that your Android device is running a compatible version of the Android operating system as specified by the Co-Bie application. This information can be found in the application's description on the Google Play Store.
- **Sufficient Storage Space:** Make sure that your Android device has enough storage space available to accommodate the Co-Bie application. The exact storage requirements will depend on the size of the application, which can be found in the application's description on the Google Play Store.
- **Google Services:** The Co-Bie application integrates with various Google services, such as Firebase and Google Maps. Therefore, it is necessary to have Google Play Services installed and up to date on your Android device. This is typically installed by default on most Android devices.

Environment Setup:

To set up Co-Bie, the following components are required:

- **Android Studio IDE:** Android Studio is an integrated development environment (IDE) used for developing Android applications. It provides a comprehensive set of tools, including code editing, debugging, and testing capabilities, making it an essential software for building the Co-Bie app.
- **Android Emulator:** An Android emulator is a virtual device that mimics the functionality of a physical Android device. It allows developers to test their applications on various virtual device configurations without the need for a physical device. The emulator is necessary to run and evaluate the Co-Bie app during development.
- **Personal Google Maps API Key:** A Google Maps API key is a unique identifier that allows access to Google Maps services. Co-Bie utilizes Google Maps functionality for features like displaying maps, geolocation, and route planning. By obtaining a personal API key from the Google Cloud Platform, developers can ensure proper integration of these services into the Co-Bie app.

Installation Process:

- Clone the code from GitHub: Download a copy of the code repository from GitHub to your local machine using the Git version control system.
- Open the code in Android Studio: Launch the Android Studio development environment and load the downloaded code project into it for further editing and building.
- Add your personal Google API key to Android Studio: Insert your unique Google API key into the appropriate configuration file of the project to enable access to Google services and APIs.
- Run the project with the emulator or with an Android device: Execute the project within Android Studio using either a software emulator or a physical Android device for testing and running the application.

References

Related Work:

- Pinterest – <https://en.wikipedia.org/wiki/Pinterest>
- Meetup – <https://en.wikipedia.org/wiki/Meetup>
- Facebook – <https://en.wikipedia.org/wiki/Facebook>

Survey:

- Google Forms - <https://www.google.com/forms/about/>

Agile Development:

- Definition – <https://www.javatpoint.com/software-engineering-agile-model>

Android:

- Definition – [https://en.wikipedia.org/wiki/Android_\(operating_system\)](https://en.wikipedia.org/wiki/Android_(operating_system))

Client-Server Model:

- Definition – <https://www.javatpoint.com/computer-network-client-and-server-model>

Mobile Cloud Computing:

- Definition – <https://azure.microsoft.com/en-in/resources/cloud-computing-dictionary/what-is-cloud-computing/#benefits>

Espresso Testing Framework:

- Definition – <https://developer.android.com/training/testing/espresso>

JUnit Testing Framework:

- Definition – <https://en.wikipedia.org/wiki/JUnit>

Git Link

<https://github.com/RazAvr3/Co-Bie>