Grimpeurs Cycling Club

SEG 2105 - Intro to Software Engineering
Fall 2023
School of Electrical Engineering and Computer Science
University of Ottawa

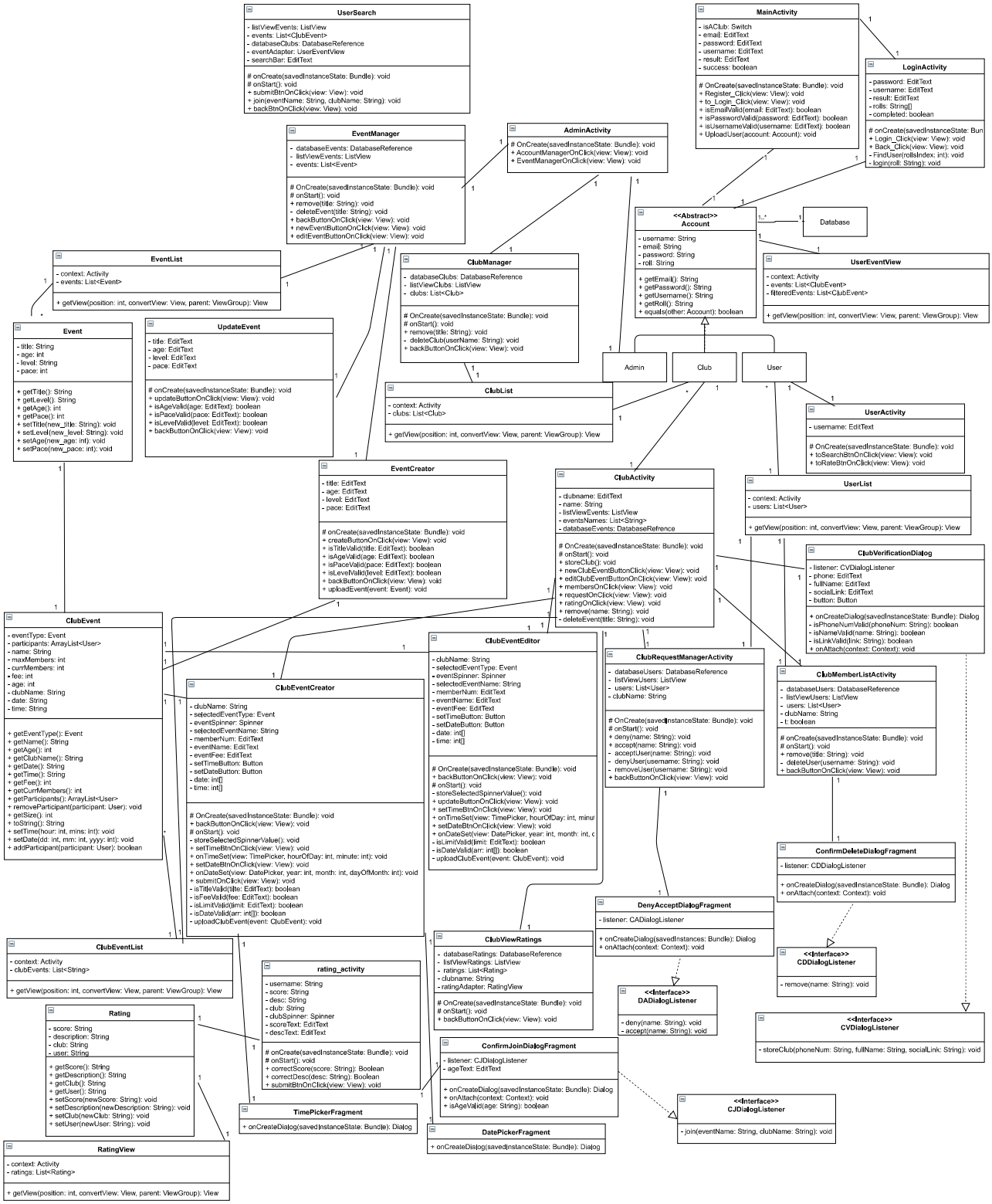
Course Coordinator: Dr. Garzón, Miguel
Teaching Assistants: Bhandari, Shivani
Rajendran, Sai Swarna Dowley
Kadiwar, Prince Balvantrai
Dara, Supriya
Zielinska, Emilia
Kakavand, Zahra
Chauhan, Aditya
Nanda, Mehul
Vundela, Maheedhar

Group 101
Rachel Olugbemiro 300302892
Winona Chung 300298148
Jonathan Cojita 300283917
Jarrett Lowinski 300301837
Submission Date: Dec 6, 2023

Introduction

This report documents the entire progress of the cycling app project from class SEG2105, section B and C. The cycling app project began on September 25th 2023, and ended on December 6th 2023. The project was divided into four deliverables and the app was developed through Android Studio with APK -. For a database, Firebase was employed. The main objectives of this project are;

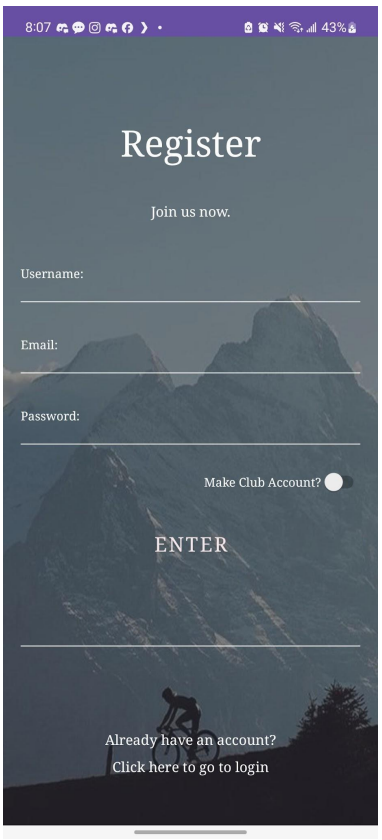
- Implement login and register features linked to Firebase
- Create multiple types of accounts with unique features (administrator, club and user)
- Implementing the event creator and allowing users to join events individually
- Implement of user managing features for administrative and club accounts

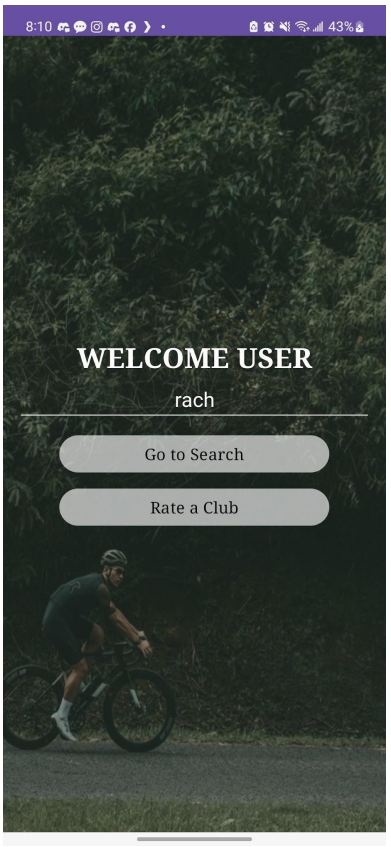

UserSearch
- listViewEvents: ListView
- events: List<ClubEvent>
- databaseClubs: DatabaseReference
- eventAdapter: UserEventView
- searchBar: EditText
onCreate(savedInstanceState: Bundle): void
onStart(): void
+ submitBtnOnClick(view: View): void
+ join(eventName: String, clubName: String): void
+ backBtnOnClick(view: View): void
MainActivity
- isAClub: Switch
- email: EditText
- password: EditText
- username: EditText
- result: EditText
- success: boolean
OnCreate(savedInstanceState: Bundle): void
+ Register_Click(view: View): void
+ to_Login_Click(view: View): void
+ isEmailValid(email: EditText): boolean
+ isPasswordValid(password: EditText): boolean
+ isUsernameValid(username: EditText): boolean
+ UploadUser(account: Account): void
LoginActivity
- password: EditText
- username: EditText
- result: EditText
- rolls: String[]
- completed: boolean
onCreate(savedInstanceState: Bun
+ Login_Click(view: View): void
+ Back_Click(view: View): void
- FindUser(rollsIndex: int): void
- login(roll: String): void
EventManager
- databaseEvents: DatabaseReference
- listViewEvents: ListView
- events: List<Event>
OnCreate(savedInstanceState: Bundle): void
onStart(): void
- remove(title: String): void
- deleteEvent(title: String): void
+ backButtonOnClick(view: View): void
+ newEventButtonOnClick(view: View): void
+ editEventButtonOnClick(view: View): void
AdminActivity
OnCreate(savedInstanceState: Bundle): void
+ AccountManagerOnClick(view: View): void
+ EventManagerOnClick(view: View): void
<<Abstract>>
Account
- username: String
- email: String
- password: String
- roll: String
+ getEmail(): String
+ getPassword(): String
+ getUsername(): String
+ getRoll(): String
+ equals(other: Account): boolean
Database
EventList
- context: Activity
- events: List<Event>
+ getView(position: int, convertView: View, parent: ViewGroup): View
ClubManager
- databaseClubs: DatabaseReference
- listViewClubs: ListView
- clubs: List<Club>
OnCreate(savedInstanceState: Bundle): void
onStart(): void
+ remove(title: String): void
- deleteClub(userName: String): void
+ backButtonOnClick(view: View): void
UserEventView
- context: Activity
- events: List<ClubEvent>
- filteredEvents: List<ClubEvent>
+ getView(position: int, convertView: View, parent: ViewGroup): View
Event
- title: String
- age: int
- level: String
- pace: int
+ getTitle(): String
+ getLevel(): String
+ getAge(): int
+ getPace(): int
+ setTitle(new_title: String): void
+ setLevel(new_level: String): void
+ setAge(new_age: int): void
+ setPace(new_pace: int): void
UpdateEvent
- title: EditText
- age: EditText
- level: EditText
- pace: EditText
onCreate(savedInstanceState: Bundle): void
+ updateButtonOnClick(view: View): void
+ isAgeValid(age: EditText): boolean
+ isPaceValid(pace: EditText): boolean
+ isLevelValid(level: EditText): boolean
+ backButtonOnClick(view: View): void
Admin
Club
User
ClubList
- context: Activity
- clubs: List<Club>
+ getView(position: int, convertView: View, parent: ViewGroup): View
UserActivity
- username: EditText
OnCreate(savedInstanceState: Bundle): void
+ toSearchBtnOnClick(view: View): void
+ toRateBtnOnClick(view: View): void
EventCreator
- title: EditText
- age: EditText
- level: EditText
- pace: EditText
onCreate(savedInstanceState: Bundle): void
+ createButtonOnClick(view: View): void
+ isTitleValid(title: EditText): boolean
+ isAgeValid(age: EditText): boolean
+ isPaceValid(pace: EditText): boolean
+ isLevelValid(level: EditText): boolean
+ backButtonOnClick(view: View): void
+ uploadEvent(event: Event): void
ClubActivity
- clubname: EditText
- name: String
- listViewEvents: ListView
- eventsNames: List<String>
- databaseEvents: DatabaseRefrence
OnCreate(savedInstanceState: Bundle): void
onStart(): void
+ storeClub(): void
+ newClubEventButtonOnClick(view: View): void
+ editClubEventButtonOnClick(view: View): void
+ membersOnClick(view: View): void
+ requestOnClick(view: View): void
+ ratingOnClick(view: View): void
+ remove(name: String): void
- deleteEvent(title: String): void
UserList
- context: Activity
- users: List<User>
+ getView(position: int, convertView: View, parent: ViewGroup): View
ClubVerificationDialog
- listener: CVDialogListener
- phone: EditText
- fullName: EditText
- socialLink: EditText
- button: Button
+ onCreateDialog(savedInstanceState: Bundle): Dialog
- isPhoneNumValid(phoneNum: String): boolean
- isNameValid(name: String): boolean
- isLinkValid(link: String): boolean
+ onAttach(context: Context): void
ClubEvent
- eventType: Event
- participants: ArrayList<User>
- name: String
- maxMembers: int
- currMembers: int
- fee: int
- age: int
- clubName: String
- date: String
- time: String
+ getEventType(): Event
+ getName(): String
+ getAge(): int
+ getClubName(): String
+ getDate(): String
+ getTime(): String
+ getFee(): int
+ getCurrMembers(): int
+ getParticipants(): ArrayList<User>
+ removeParticipant(participant: User): void
+ getSize(): int
+ toString(): String
+ setTime(hour: int, mins: int): void
+ setDate(dd: int, mm: int, yyyy: int): void
+ addParticipant(participant: User): boolean
ClubEventEditor
- clubName: String
- selectedEventType: Event
- eventSpinner: Spinner
- selectedEventName: String
- memberNum: EditText
- eventName: EditText
- eventFee: EditText
- setTimeButton: Button
- setDateButton: Button
- date: int[]
- time: int[]
OnCreate(savedInstanceState: Bundle): void
+ backButtonOnClick(view: View): void
# onStart(): void
- storeSelectedSpinnerValue(): void
+ updateButtonOnClick(view: View): void
+ setTimeBtnOnClick(view: View): void
+ onTimeSet(view: TimePicker, hourOfDay: int, minu
+ setDateBtnOnClick(view: View): void
+ onDateSet(view: DatePicker, year: int, month: int, c
- isLimitValid(limit: EditText): boolean
- isDateValid(arr: int[]): boolean
- uploadClubEvent(event: ClubEvent): void
ClubRequestManagerActivity
- databaseUsers: DatabaseReference
- listViewUsers: ListView
- users: List<User>
- clubName: String
OnCreate(savedInstanceState: Bundle): void
onStart(): void
+ deny(name: String): void
+ accept(name: String): void
- acceptUser(name: String): void
- denyUser(username: String): void
- removeUser(username: String): void
+ backButtonOnClick(view: View): void
ClubMemberListActivity
- databaseUsers: DatabaseReference
- listViewUsers: ListView
- users: List<User>
- clubName: String
- t: boolean
onCreate(savedInstanceState: Bundle): void
onStart(): void
+ remove(title: String): void
- deleteUser(username: String): void
+ backButtonOnClick(view: View): void
ClubEventCreator
- clubName: String
- selectedEventType: Event
- eventSpinner: Spinner
- selectedEventName: String
- memberNum: EditText
- eventName: EditText
- eventFee: EditText
- setTimeButton: Button
- setDateButton: Button
- date: int[]
- time: int[]
OnCreate(savedInstanceState: Bundle): void
+ backButtonOnClick(view: View): void
# onStart(): void
- storeSelectedSpinnerValue(): void
+ setTimeBtnOnClick(view: View): void
+ onTimeSet(view: TimePicker, hourOfDay: int, minute: int): void
+ setDateBtnOnClick(view: View): void
+ onDateSet(view: DatePicker, year: int, month: int, dayOfMonth: int): void
+ submitOnClick(view: View): void
- isTitleValid(title: EditText): boolean
- isFeeValid(fee: EditText): boolean
- isLimitValid(limit: EditText): boolean
- isDateValid(arr: int[]): boolean
- uploadClubEvent(event: ClubEvent): void
ConfirmDeleteDialogFragment
- listener: CDDialogListener
+ onCreateDialog(savedInstanceState: Bundle): Dialog
+ onAttach(context: Context): void
DenyAcceptDialogFragment
- listener: CADialogListener
+ onCreateDialog(savedInstances: Bundle): Dialog
+ onAttach(context: Context): void
<<Interface>>
CDDialogListener
- remove(name: String): void
ClubEventList
- context: Activity
- clubEvents: List<String>
+ getView(position: int, convertView: View, parent: ViewGroup): View
rating_activity
- username: String
- score: String
- desc: String
- club: String
- clubSpinner: Spinner
- scoreText: EditText
- descText: EditText
onCreate(savedInstanceState: Bundle): void
onStart(): void
+ correctScore(score: String): Boolean
+ correctDesc(desc: String): Boolean
+ submitBtnOnClick(view: View): void
ClubViewRatings
- databaseRatings: DatabaseReference
- listViewRatings: ListView
- ratings: List<Rating>
- clubname: String
- ratingAdapter: RatingView
OnCreate(savedInstanceState: Bundle): void
onStart(): void
+ backButtonOnClick(view: View): void
<<Interface>>
DADialogListener
- deny(name: String): void
- accept(name: String): void
<<Interface>>
CVDialogListener
- storeClub(phoneNum: String, fullName: String, socialLink: String): void
Rating
- score: String
- description: String
- club: String
- user: String
+ getScore(): String
+ getDescription(): String
+ getClub(): String
+ getUser(): String
+ setScore(newScore: String): void
+ setDescription(newDescription: String): void
+ setClub(newClub: String): void
+ setUser(newUser: String): void
ConfirmJoinDialogFragment
- listener: CJDialogListener
- ageText: EditText
+ onCreateDialog(savedInstanceState: Bundle): Dialog
+ onAttach(context: Context): void
+ isAgeValid(age: String): boolean
<<Interface>>
CJDialogListener
- join(eventName: String, clubName: String): void
TimePickerFragment
+ onCreateDialog(savedInstanceState: Bundle): Dialog
DatePickerFragment
+ onCreateDialog(savedInstanceState: Bundle): Dialog
RatingView
- context: Activity
- ratings: List<Rating>
+ getView(position: int, convertView: View, parent: ViewGroup): View

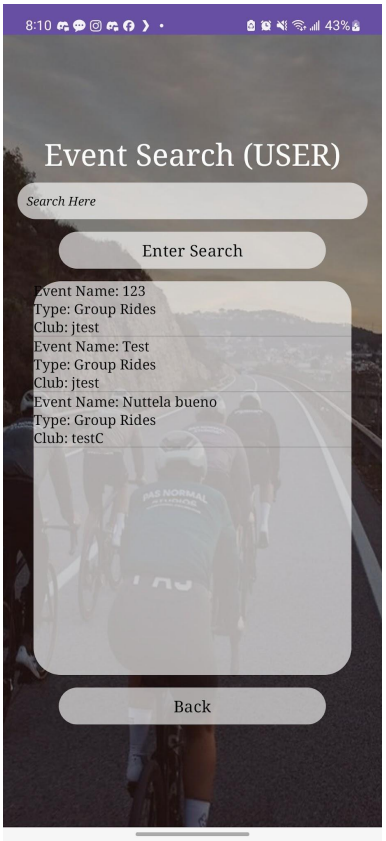
Project Roles

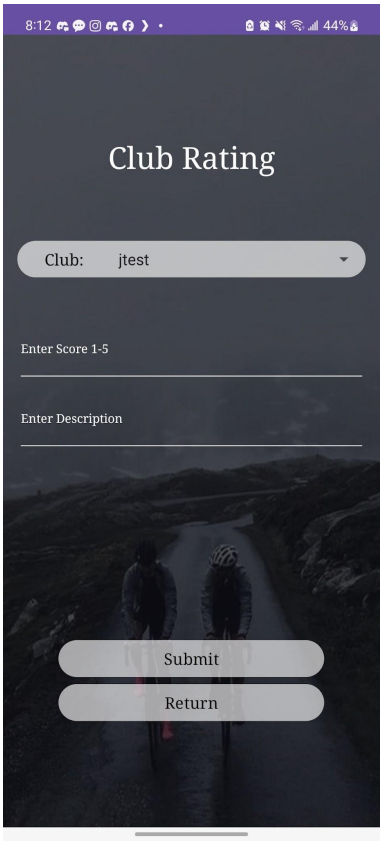
Name	Winona Chung	Jarrett Lowinski	Jonathan Cojita	Rachel Olugbemiro
Role	**UI Design**	**Club/Event Editor**	**Firebase Integrator**	**UML Diagrams**
Deliverable 1	- Developed UI designs	- Added Admin, Club, and User account activities	- Created base for all account types - Implemented register feature for all account types - Stored login info in Firebase	- Created UML Diagram
Deliverable 2	- Developed UI designs	- Implemented Admin event creation and deletion	- Implemented Login and Admin features	- Updated UML Diagram
Deliverable 3	- Developed UI designs	- Added club additional information prompt - Added lists showing a club's current members and requested users	- Added Club event creation	- Updated UML Diagram
Deliverable 4	- Developed UI designs - Worked on Final Report	- Implemented users joining ClubEvents	- Added User's event search feature	- Updated UML Diagram - Worked on Final Report

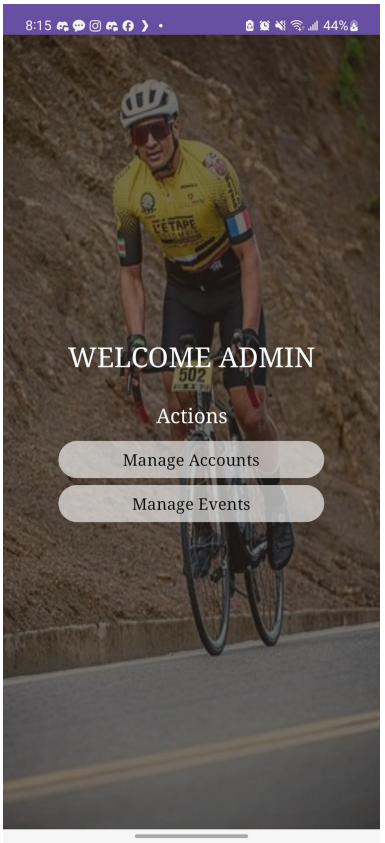
Screenshots

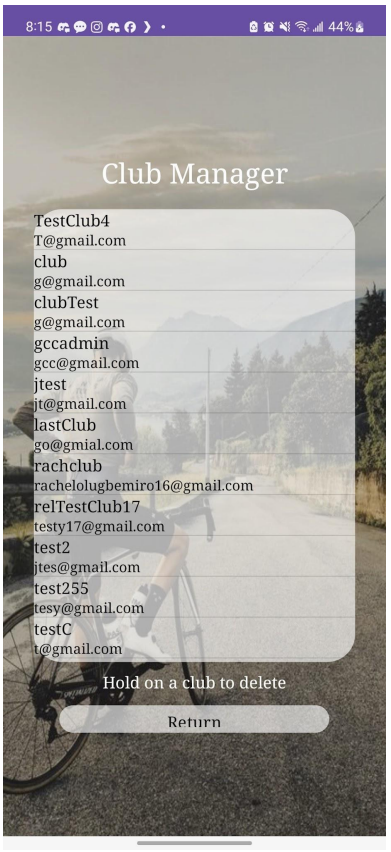


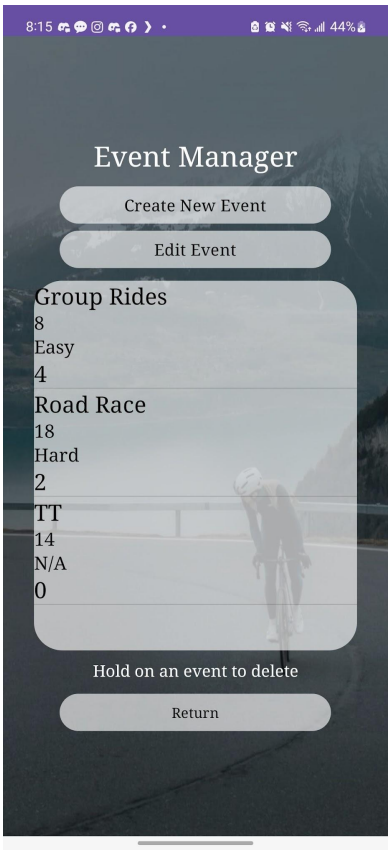


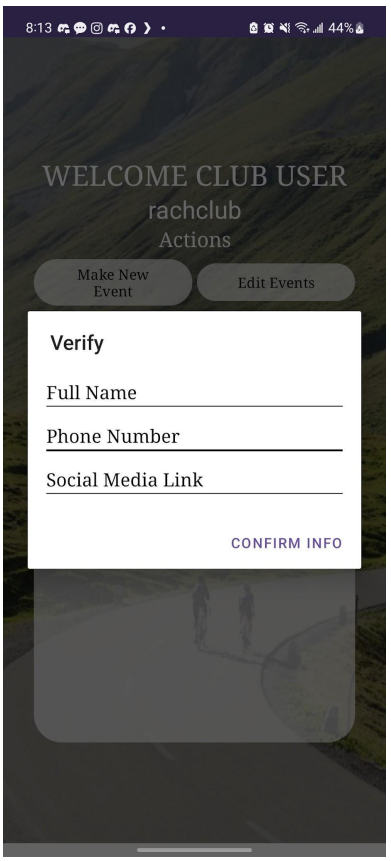


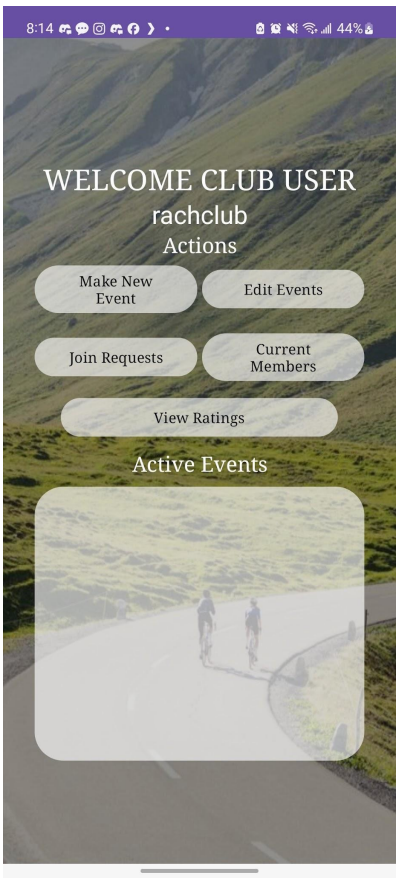


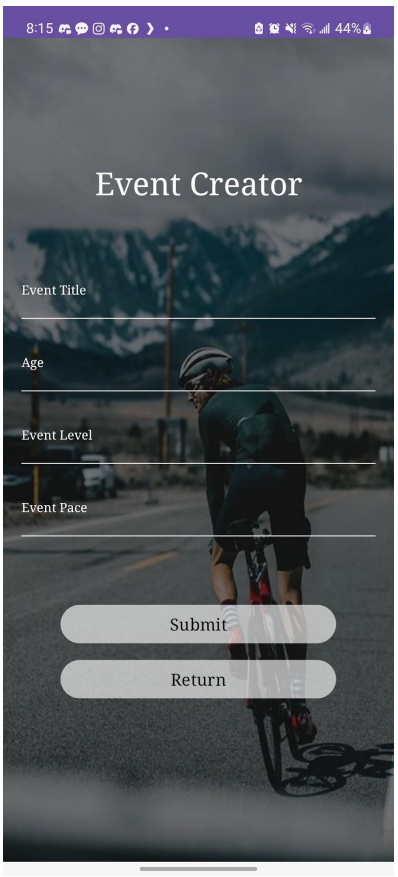


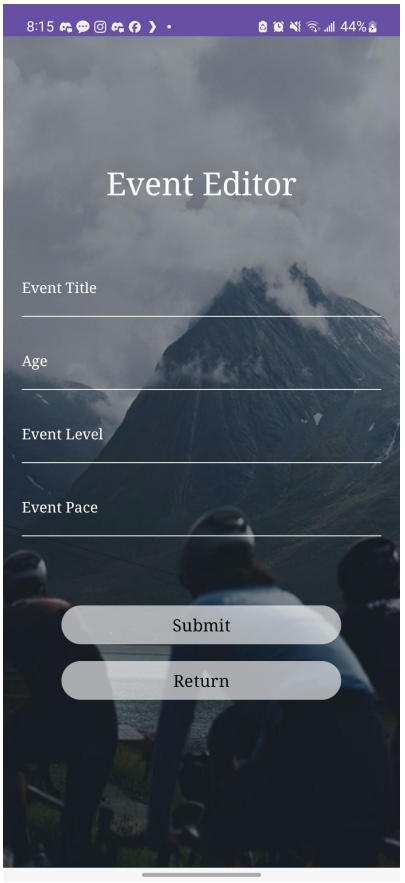

Club Manager
TestClub4
T@gmail.com
club
g@gmail.com
clubTest
g@gmail.com
gccadmin
gcc@gmail.com
jtest
jt@gmail.com
lastClub
go@gmial.com
rachclub
rachelolugbemiro16@gmail.com
relTestClub17
testy17@gmail.com
test2
jtes@gmail.com
test255
tesy@gmail.com
testC
t@gmail.com
Hold on a club to delete
Return

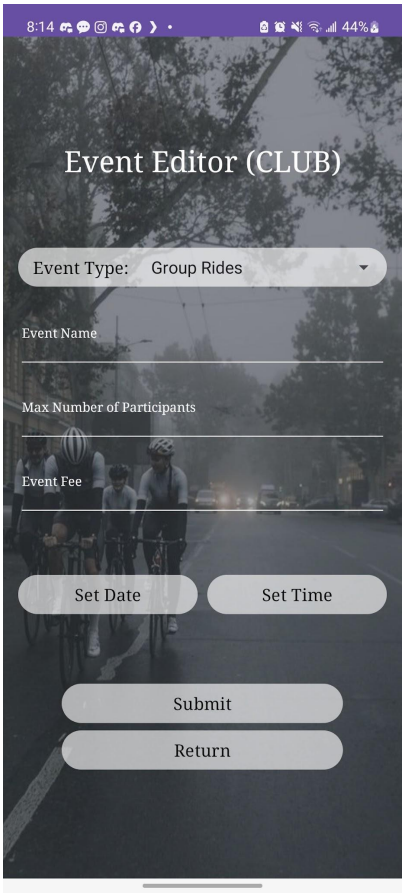

Event Manager
Create New Event
Edit Event
Group Rides
8
Easy
4
Road Race
18
Hard
2
TT
14
N/A
0
Hold on an event to delete
Return

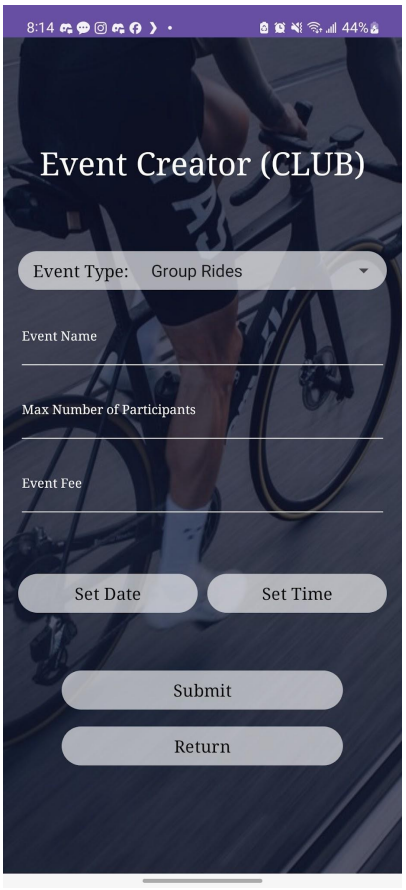

WELCOME CLUB USER
rachclub
Actions
Make New Event
Edit Events
Verify
Full Name
Phone Number
Social Media Link
CONFIRM INFO


WELCOME CLUB USER
rachclub
Actions
Make New Event
Edit Events
Join Requests
Current Members
View Ratings
Active Events


Event Creator
Event Title
Age
Event Level
Event Pace
Submit
Return


Event Editor
Event Title
Age
Event Level
Event Pace
Submit
Return


8:14
44%
Event Editor (CLUB)
Event Type:
Group Rides
Event Name
Max Number of Participants
Event Fee
Set Date
Set Time
Submit
Return


8:14
44%
Event Creator (CLUB)
Event Type:
Group Rides
Event Name
Max Number of Participants
Event Fee
Set Date
Set Time
Submit
Return

Lessons Learned

During our time collaborating on this app, we encountered some issues that affected the trajectory of our progress. These problems were usually found through Firebase, merge conflicts with GitHub, and our project management skills. Using Firebase was a new experience for all of us and brought a lot of challenges with its default permissions and inputting data into the platform. One of the issues we came across happened to be the default Firebase permissions as they were not set to read and write data so during our first experience using the platform, we had trouble going into Firebase directly and changing information there. Another issue we faced with Firebase was that it took in null values when we needed it to reject them. All of our problems relating to Firebase were resolved after submitting Deliverable 1 as we go more accustomed to using the platform. Additionally, we had trouble with using GitHub when it came to collaborating on the same project. An example of a constant issue we came across was pushing code which resulted in merging conflicts. These problems occurred as there were sometimes 2 group members working on the same piece of code and when their tasks were finished and they pushed their code, it created issues related to different value names being replaced and basic app functionalities being overwritten. Similarly, we had the same issues with GitHub but they pertained to group members debugging in the same section. A specific example would have to be when one of us deleted a club on Firebase while another was debugging using that club to test their code. Another example would also be us at some point having 2 instances of admin in our program when there only needed to be 1. A key takeaway from using a website like GitHub would be to coordinate more on the code that is being pushed to the repository and to make sure that others are not working on the same tasks to limit these merge conflicts from happening. Moreover, our project management skills also needed to be improved. Something that we all could have improved on was making sure we had a set time every week to debrief about our progress and delegate new tasks so everyone would be on the same page. In conclusion, working on this project was not easy. We faced a lot of problems that made the project hard, but it taught us a lot about what it's like to work as a group in the tech field.