## **Android Project: Event Attendance Management System (EAMS)** SEG 2105 [A] – Intro to Software Engineering Fall 2024 University of Ottawa

Professor: Dr. Hussein Al Osman

**Group 19**  
Alex Ajersch 300292166

Brooklyn McClelland 300311745

Moïse Kenge Ngoyi 300308366

Naomi Braun 300337199

Rachel Qi 300292435

Steven Wu 300370421

Submission Date: 2024/12/04

# Introduction

The Event Attendance Management System (EAMS) is an Android app developed with Java and Android Studio, allowing users to post or register to events and track attendance. The app uses Firebase, a real-time database system, to store information about user logins and event registrations.

There are three types of users: Administrators, Attendees, and Organizers. Attendees and Organizers must register for the app and can only begin using it once their registration has been approved. An Administrator is a pre-registered user that can view, approve, and reject these app registration requests from Attendees and Organizers. Once approved, an Attendee can search for events, register to or cancel registration to an event, and view the list of events they have registered to, with an indicator reflecting their event registration approval status. Once approved, an Organizer can create and view upcoming and past events, accept or reject Attendee registration requests, and view the list of registered Attendees to an event.

The general Firebase structure is as follows:

{

"events": {

"eventKey": {

"city": "",

"databaseKey": "eventKey",

"date": "",

"description": "",

"endTime": "",

"postalCode": "",

"province": "",

"registeredAttendees": {

"attendeeKey1": "approved",

"attendeeKey2": "pending",

"attendeeKey3": "rejected"

},

"startTime": "",

"street": "",

"title": ""

}

},

"users": {

"administrators": {

"adminKey": {

"email": "",

"password": ""

}

},

"attendees": {

"approved": {

"attendeeKey": {

"city": "",

"databaseKey": "attendeeKey",

"email": "",

"eventsRegisteredTo": {

"eventKey1": "pending",

"eventKey2": "approved",

"eventKey3": "rejected"

},

"firstName": "",

"lastName": "",

"password": "",

"phoneNumber": "",

"postalCode": "",

"province": "",

"street": ""

}

},

"pending": {

},

"rejected": {

}

},

"organizers": {

"approved": {

"organizerKey": {

"city": "",

"databaseKey": "organizerKey",

"email": "",

"firstName": "",

"lastName": "",

"organization": "",

"password": "",

"phoneNumber": "",

"postalCode": "",

"province": "",

"street": ""

}

},

"pending": {

},

"rejected": {

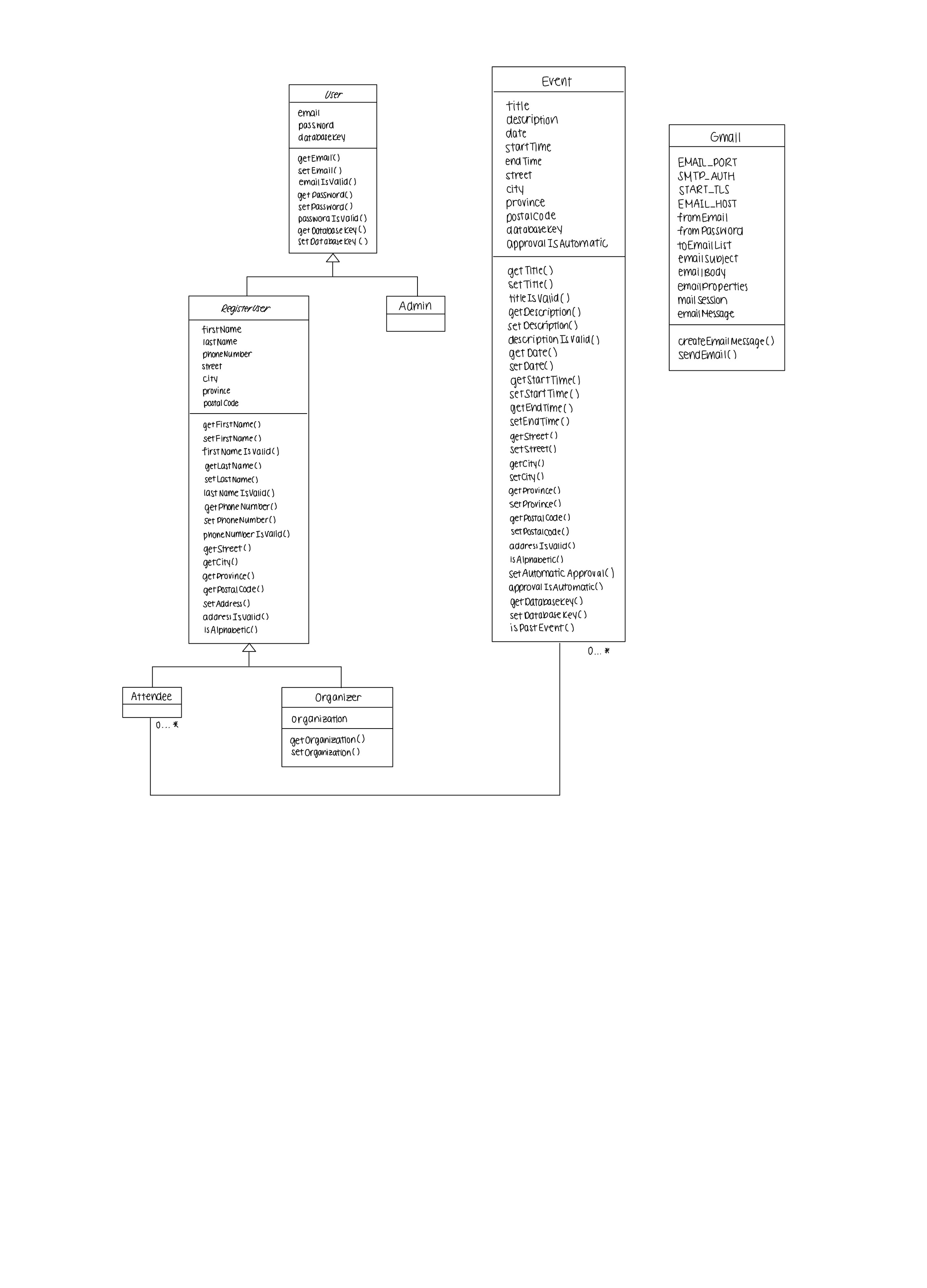
}

}

}

}

***Figure 1: Updated UML Diagram***



***Table 1: Table Specifying the Contributions of Team Members for each Deliverable***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Deliverable 1** | **Deliverable 2** | **Deliverable 3** | **Deliverable 4** |
| Alex Ajersch | * GitHub Classroom Setup * Created User Classes and Class Hierarchy * Field Validation | * Administrator Features (view list of requests & user info) * Registration Request Management (functionality for accept/reject) * Rejected Registration Requests (approve previously rejected) * Database Storage for Registration Requests * Comments Addition | * View Past & Upcoming Events * Field Validation and Error Messages * Attendee Registration Management (view list of requests & user info) * Firebase Structure Update | * Search Events * Request Event Registration |
| Brooklyn McClelland | * Account Creation (Attendee) * Field Validation * APK Submission * Demo Video Submission | * Created Admin Inbox and List UI * Retrieval of User Info from Database | * Created Event Classes * Event Creation * View Past & Upcoming Events * Rachel’s Emotional Support <3 | * Conflict Prevention * Event Deletion * Cancel Registration |
| Moïse Kenge Ngoyi | * Account Creation (Organizer) | * Bonus: Notifications * Demo Video Submission (showcasing bonus) | * Date Validation | * Bonus: Notifications |
| Naomi Braun | * Welcome Screens * Log Off Functionality | * Rejected Registration Requests (view list of rejected requests) * Demo Video Submission (showcasing app) | * Manual and Automatic Registration Approval * Attendee Registration Management (view list of requests & user info; approve (all) /reject) * Event Deletion * Demo Video Submission | * View Registered Events * Registration Status Indicator * Cancel Registration * Demo Video Submission |
| Rachel Qi | * Welcome Screens * Log Off Functionality * UML Class Diagram | * Rejected Registration Requests (view list of rejected requests) * UML Class Diagram * APK Submission * JavaDoc Addition | * Attendee Registration Management (view list of requests & user info; approve (all) /reject) * Manual and Automatic Registration Approval * Event Deletion * UI Overhaul * Firebase Structure Update * UML Class Diagram * APK Submission | * View Registered Events * Registration Status Indicator * Cancel Registration * UML Class Diagram * Final Report * APK Submission |
| Steven Wu | * Bonus: Database Usage | * Login Functionality Based on Registration Status (redirection to welcome screen or message) | * Attendee Registration Management (view list of requests & user info) * Firebase Structure Update * Refactoring/Debugging | * Circle CI Integration * Unit Test Cases * Instrumentation Tests * Refactoring/Debugging |

Note: The majority of Naomi Braun and Rachel Qi’s contributions are identical because they coded together. Thank you, Brooklyn! <3

# App Screenshots

|  |  |
| --- | --- |
| ***Figure 2: Welcome (Sign In) Screen*** | ***Figure 3: Organizer Registration*** |
| ***Figure 4: Attendee Registration*** | ***Figure 5: Attendee’s Welcome Page***  ***A screen shot of a cell phone  Description automatically generated*** |
| ***Figure 6: Attendee’s View Events*** | ***Figure 7: Attendee’s Search Events***  ***A screenshot of a cell phone  Description automatically generated*** |
| ***Figure 8: Attendee’s Event Details*** | ***Figure 9: Organizer’s Welcome Page*** |
| ***Figure 10: Organizer’s Create New Event*** | ***Figure 11: Organizer’s View Events*** |
| ***Figure 12: Organizer’s View Event Registrations*** | ***Figure 13: Organizer’s View Registration Info*** |
| ***Figure 14: Admin Welcome Page*** | ***Figure 15: Admin Inbox (Pending)*** |
| ***Figure 16: Admin Inbox (Rejected)*** |  |

# Lessons Learned

This project taught us how to code in collaboration with others. From splitting tasks, to solving merge conflicts, we faced many challenges that stemmed from having to work as a group. At first, it was difficult to split the workload when the functionality of the app was so complex and often intertwined with other behaviour. For example, testing Deliverable 3 was difficult without the completion of Deliverable 4 to register Attendees. However, we learned to split tasks by a specific function of the app, rather than between front-end and back-end. This greatly improved our efficiency, as team members were less reliant on others to begin their parts of the project.

Deliverable 3 also taught us to approach coding with a plan and be adaptable to change. The association between Attendees and Events was initially designated to a third class, EventRegistration. However, we quickly realized that this was overly complicated to implement with Firebase and Android Studio, but failed to develop a concrete plan before re-approaching the code. This resulted in multiple Firebase structure changes between Event and Attendee, with each change requiring that we update all the associated code. After many failed attempts, we conceptualized the plan to have Attendees and Events have a double-sided association in Firebase. Once we had this plan in place, we were able to effortlessly overhaul the code and implement the required features. This effort could also have been avoided by separating the Firebase methods into another class, limiting the changes to one file each time the Firebase structure changed.