Android Project: EAMS

Event Attendance Management System

Final Report

SEG2105: Introduction to Software Engineering

School of Electrical Engineering and Computer Science

Fall 2024

University of Ottawa

Dr. Hussein Al Osman

Group 25

Matias Suxo 300152616
Aditya Baindur 300382718
Miller Ding 300361017
Mark Chen 300354734
Malik Buser 300365340
Jaden Fielding 300344524

Submission Date: December 4th, 2024

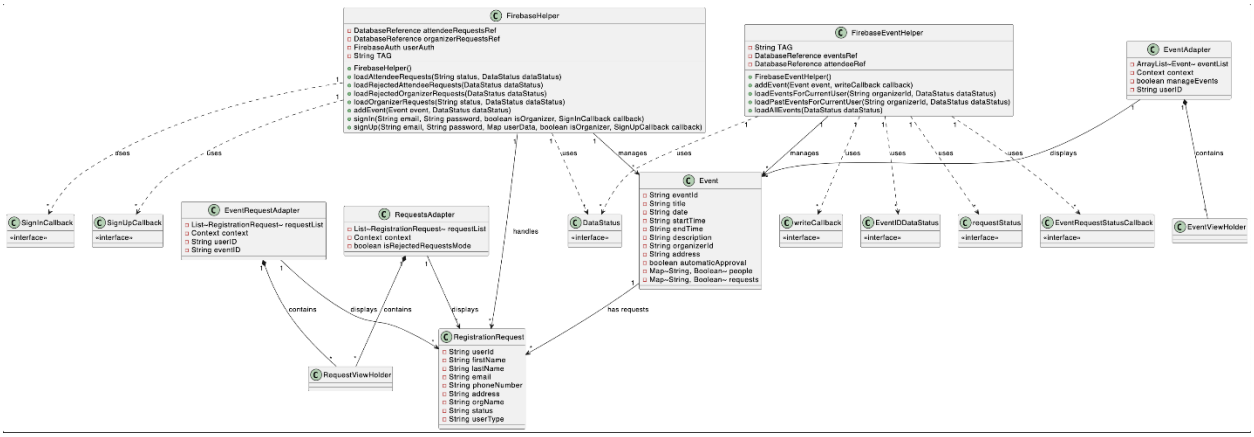
Table of Contents

Introduction

It was sought out to construct an Android app with the purpose of Event Management. This solution is comprised of 3 user types; Attendees; Organizers; Administrator. Of the three, organizers are accepted by the administrator to organize new events and manage existing events, while continuing the user relationship with its attendees. The Attendees are accepted by the organizer to join their events. This application is constructed for the Android mobile platform and uses Firebase managed database host. With the overall design being revolved around the default Android UI Kit with some color palette changes. Additionally, we include our CircleCI config file in our codebase and use an email service to send network requests upon signing up for a user to receive an onboarding message. On production of the app. The mailing script is obfuscated by using a Cloudflare serverless function worker, ensuring a backend is not required and API credentials to email service remain unknown.

Unified Language Model Diagrams

Below are the final UML diagrams representing our finalized project structure:



Estimated Contribution Table

Below is an estimated distribution of work contributed throughout the project. Divided by the delivered tasks:

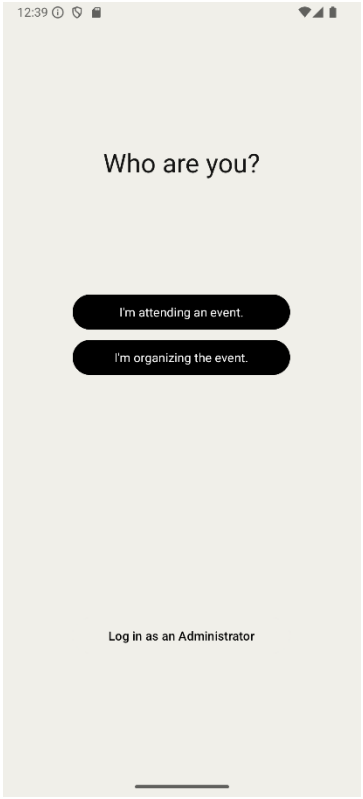
Deliverable	Aditya	Jaden	Malik	Mark	Matias	Miller
1	Made all UI Pages and lined all of them (recorded video submissions for all of the deliverables)	Organizer login page	Made welcome pages	UML	Firebase integration for user's authentication A few bug fixes.	UI Figma designs Login page UI
2	Refactored project to have a firebase helper class and other helper classes	Implemented login through firebase Auth Admin pending request page logic	Input error checking implemented Pending approval pages created	UML	Various UI activities and logic for event creation and signup approval for organizers. Email Service	Update margin and padding Implement calendar picker and sync with current time
3	Made UI page for Organizer Attendee View and worked on bug fixes	Organizer, approve/approve all/reject logic and implementation with firebase Organizer Event deletion	Added the ability for events to be put into a list and for organizer's only to access the list	UML	Bug fixes before production release Added UI and logic for event management on the organizer side CircleCI	Add checkers for invalid date selection, null descriptions, and event queries.

4	Helped write final report, minor bug fixes in the final version and recorded video submission	Attendee view events and search events pages and logic with conflict prevention Helped write final report	Small bug fixes Helped write final report	UML	Helped write final report	JUnit Tests Espresso UI Tests Firebase Search queries impl. for recent events

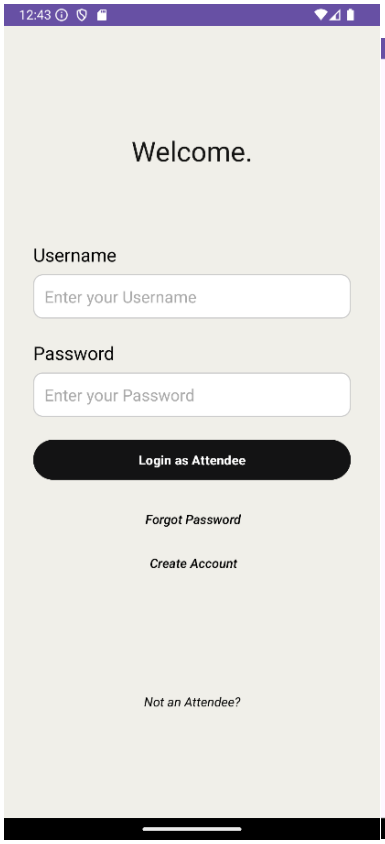
App Screenshot Showcasing

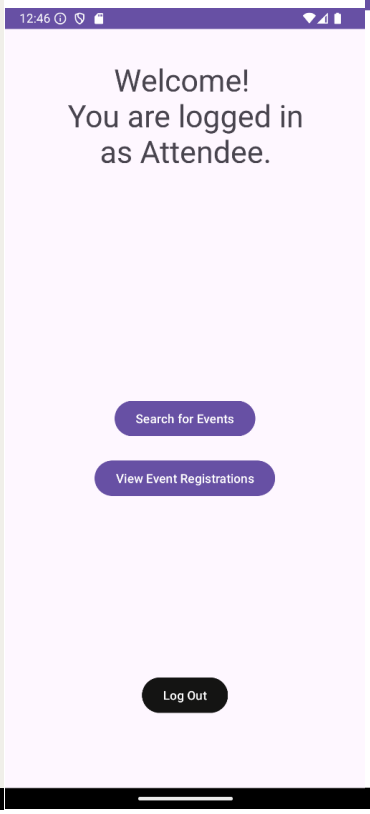
Our app is composed of various views and sections. Divergent to our video mock-up, ubiquitous and relevant apps screens are shown as screenshots below:

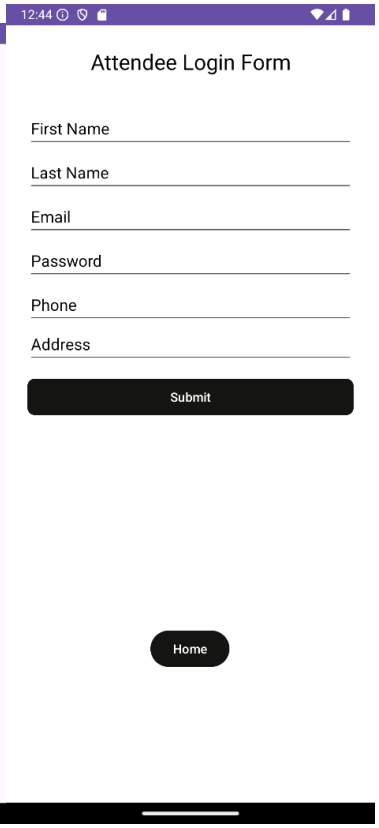
General Pages

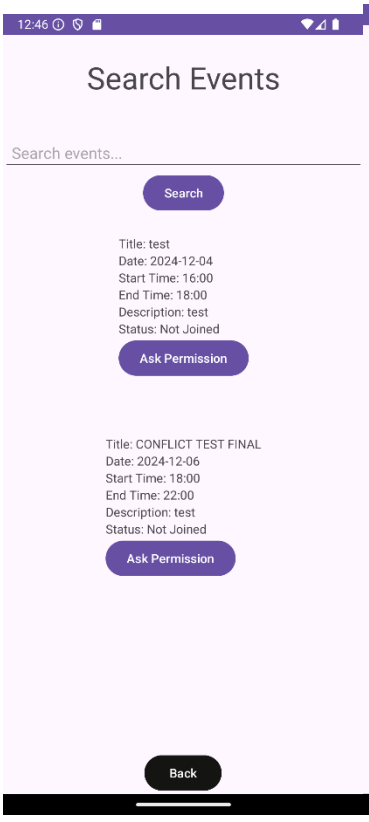


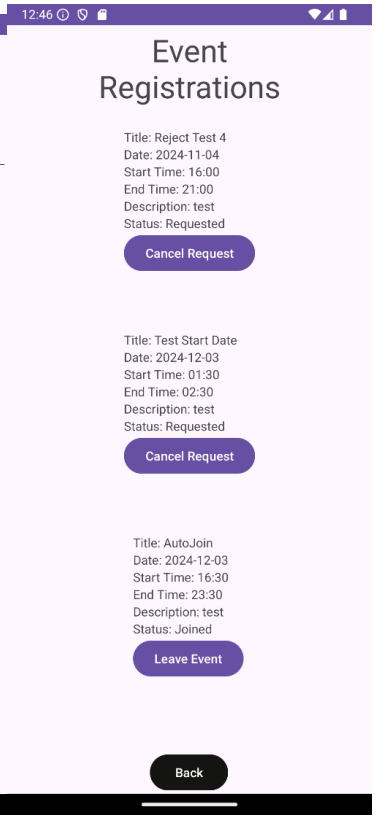
Attendee Pages



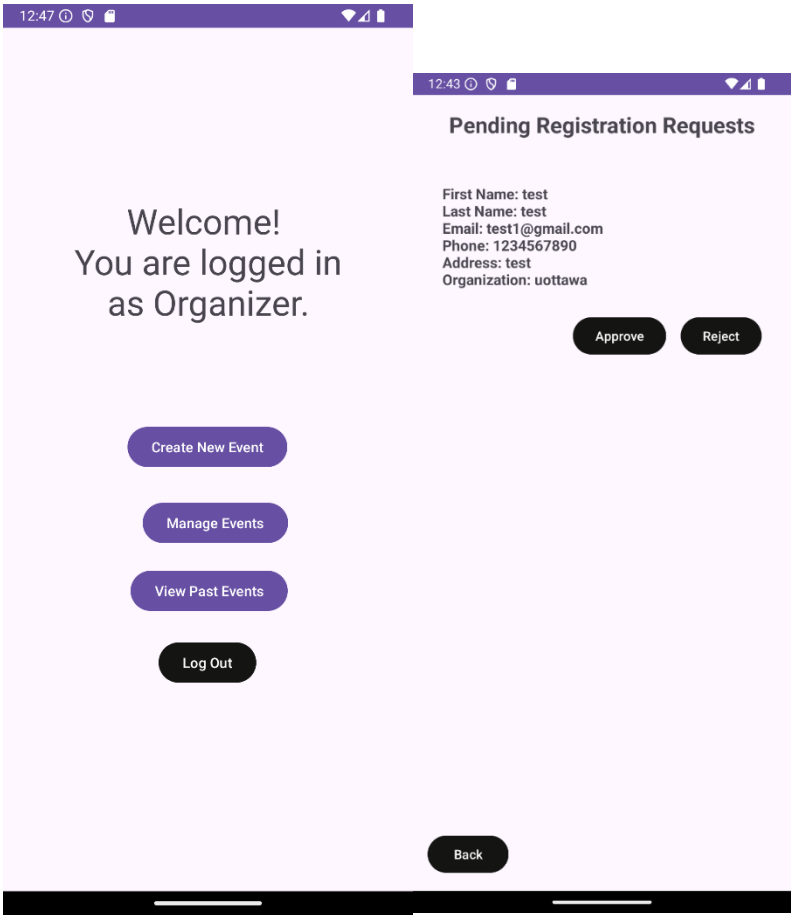




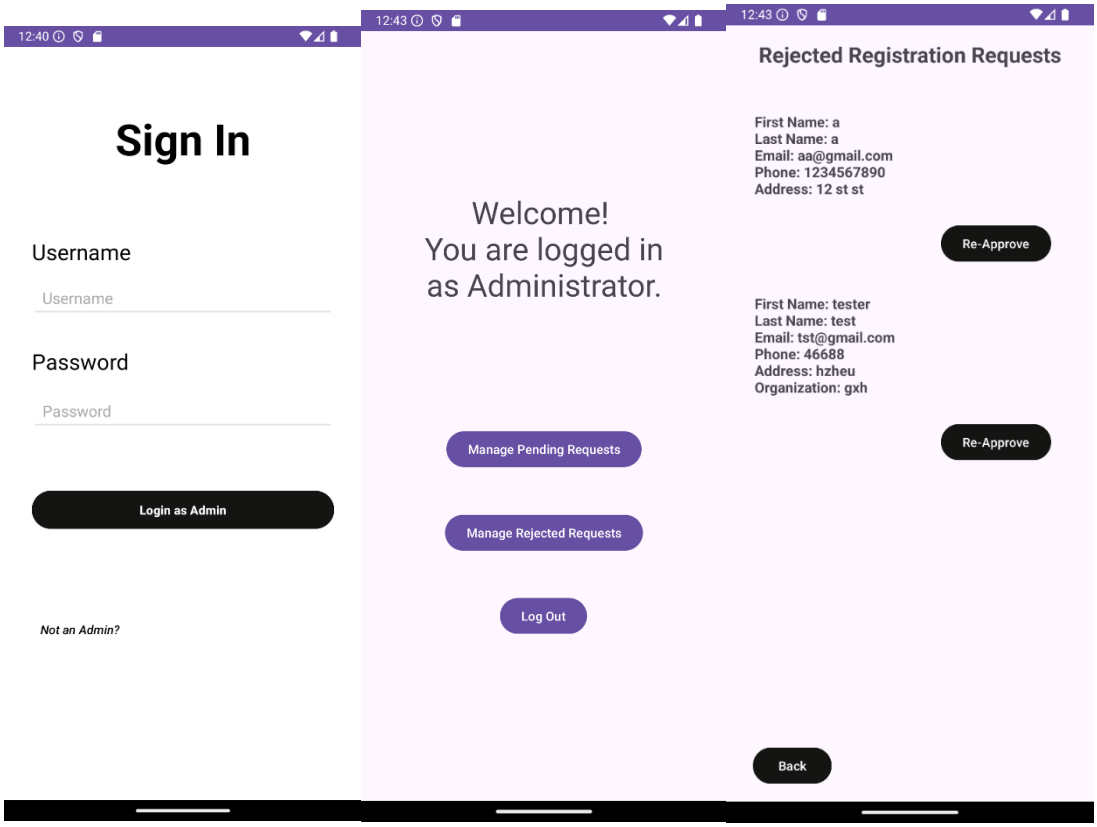




Organizer Pages



Administrator Pages



Final Remarks

Throughout the project, a lot of lessons were carried out and learnt. Some of which included, how to push/pull and perform all basic commands on GitHub and how to perform code reviews. Particularly, in android studio, we familiarized ourselves with common features of Android apps. For example, the first time we had to create a Recycler View and use it to display data using its Adapter plus other associated classes, it was a challenge. However, by the 7^{th} or 8^{th} implementation of a Recycler View, it was a breeze. Other general learnings include how to make our UI better and how to connect it all to the real-time firebase db. Particularly, it became clear as the tasks became increasingly complex, cooperation between commits was crucial as various merge conflicts occurred or two people in the group ended up implementing the same feature in their own way. We quickly came up with a more agile approach to managing the releases and minimized this occurrence towards the end.

The labs and tutorial sessions of this fall session of this SEG 2105 class helped a lot in understanding how to use GitHub, firebase and Circle CI. Overall, a lot was learned from this project.