Deliverable 4: EAMS Project Report

SEG2105[A] - Introduction to Software Engineering

Fall 2024
School of Electrical Engineering and Computer Science
University of Ottawa

Course Coordinator: Professor Hussein Al Osman

Justin El-Chibani 300337191
James Malek 300352042
Melanie Malek 300351596
Ralph Nabhan 300338908
Lourdes Najjar 300350308
Chafic Nehme 300337201

Submission Date: December 4th 2024

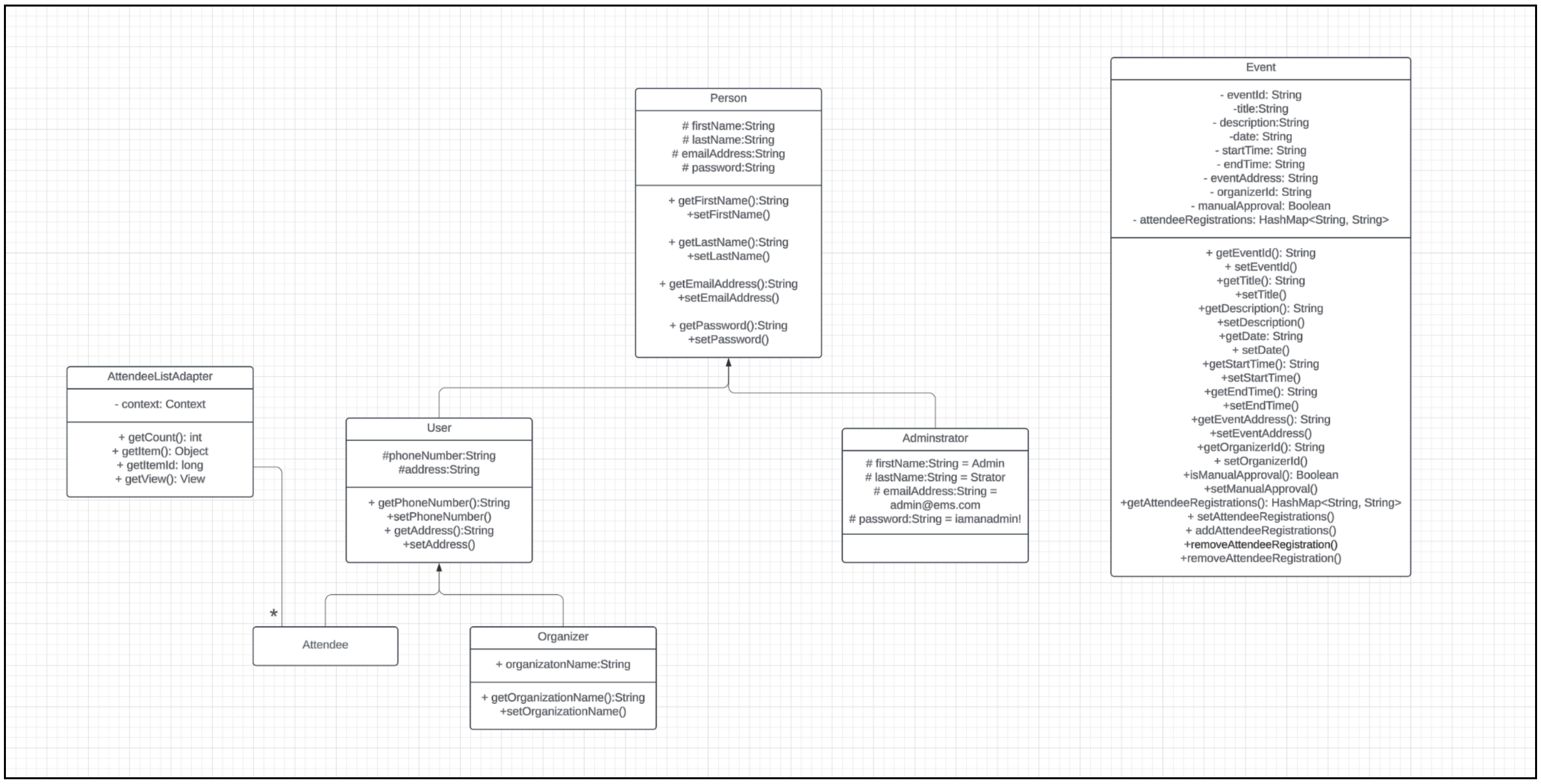
Introduction

The Event Attendance Management System (EAMS) project was developed as part of the SEG2105 (Introduction to Software Engineering) course to provide a practical learning experience in designing and implementing real-world software systems. The main objective of this project was to streamline the process of event registration and attendance tracking within a university setting. The application was built to support three major user roles: Attendee, Organizer, and Administrator, each with distinct functionalities.

Throughout the project, our team followed a progressive development process, delivering features in various phases to ensure seamless integration and adherence to specifications. Leveraging Android Studio and Firebase for development, the EAMS project provided practical experience in mobile applications and emphasized collaborative teamwork, user-centric design principles, and robust system architecture.

This report displays an overview of our development process, including a UML class diagram, contributions from each team member, screenshots of our application, and lessons learned.

UML Class Diagram



Contributions

Team Members	Deliverable 1	Deliverable 2	Deliverable 3	Deliverable 4
Justin El-Chibani	-Implemented logoff functionality -Helped with sign-up page UI	-Login Functionality Based on Registration Status	-Field Validation -Help with demo video submission	-Field Validation and Error messages -Reimplemented the logoff functionality to make it up-to-date
James Malek	-Classes Creation -Comments: includes Javadoc documentation -Welcome Screen specification -Logging off proper functionality -Demo video	-Administrator features	-Javadoc Documentation -Comments	-Cancel Registration -Event deletion -Screenshots
Melanie Malek	- Field Validation - Creation of entity classes - Javadoc Documentation + comments - APK submission - Demo video submission	- Field Validation - APK submission - Demo video submission	- Javadoc Documentation + comments - Demo video submission	- Conflict prevention - Field Validation - Project report: introduction, lessons learned - APK Submission -Demo video submission
Ralph Nabhan	Set up the repository for version control and project collaboration. Implemented a screen displayed after successful authentication, providing users with immediate feedback.			Added a system to notify attendees of upcoming events. Ensured notifications are triggered based on scheduled event times. Described Lessons learned in project report specifically; Time management

	Depending on the user's credentials, the screen dynamically specifies the user's role			and project scope, User-centric design, Database integration and data management
Lourdes Najjar	- Account creation - Updated some entity classes to contain all necessary information - UML diagram	- Collaborated on the UI - UML diagram	- UML diagram	- Application screenshots in the report - Search events function - Attempted to complete Unit tests - UML diagram - Assisted with conflict prevention
Chafic Nehme	- GitHub Classroom Setup - Demo Video - Account Creation - Welcome Screen - Logout Functionality - Field Validation - Database Usage	- All Administrator Features - Login Functionality Based on Registration Status - Database Storage for Registration Requests	- Event Creation - Date Validation - Manual and Automatic Registration Approval - View Events - Attendee Registration Management - Event Deletion - Field Validation and Error Messages	- View Registered Events - Registration Status Indicator - Cancel Registration - Search Events - Request Event Registration - Conflict Prevention - Event Deletion - Field Validation and Error Messages

Application Screenshots

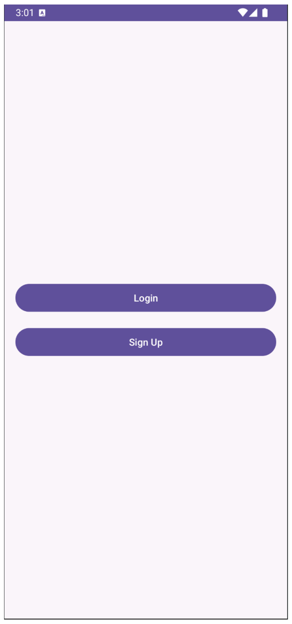


Figure 1: Welcome screen

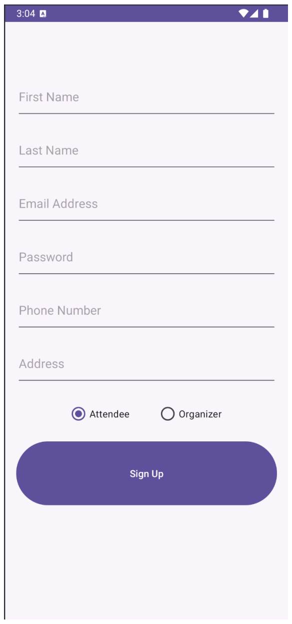


Figure 2: Sign up page for attendee

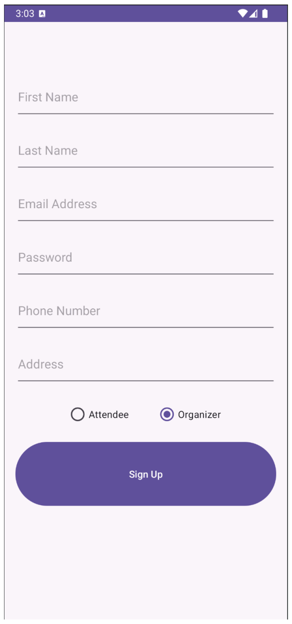
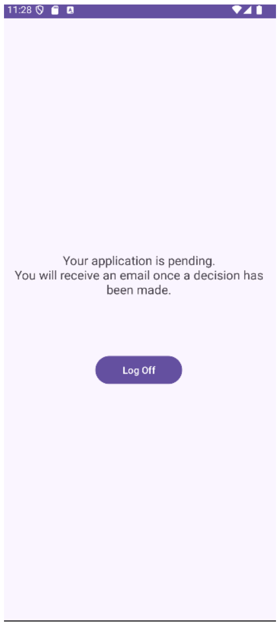


Figure 3: Sign up page for organizer



Your application is pending.
You will receive an email once a decision has been made.

Figure 4: Pending registration acceptance

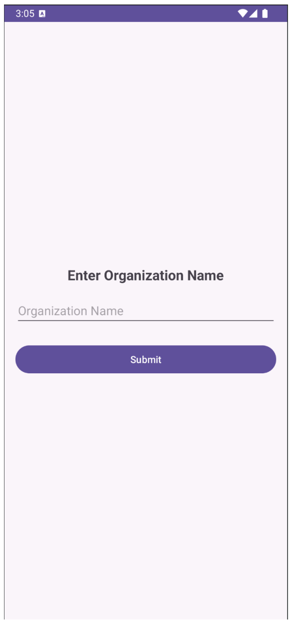


Figure 5: continuation of sign up page for organizer, Organization name

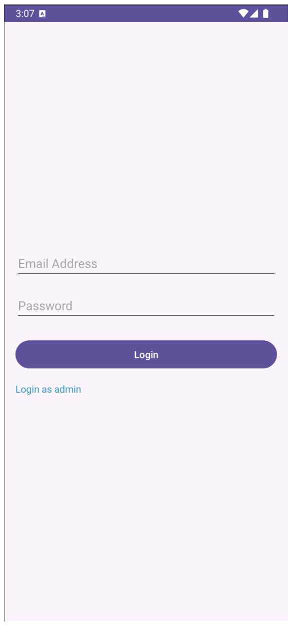


Figure 6: Login page

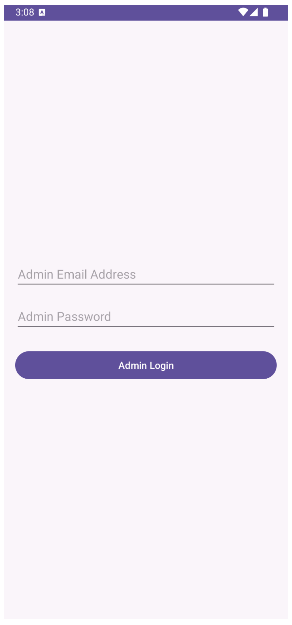


Figure 7: Admin login page

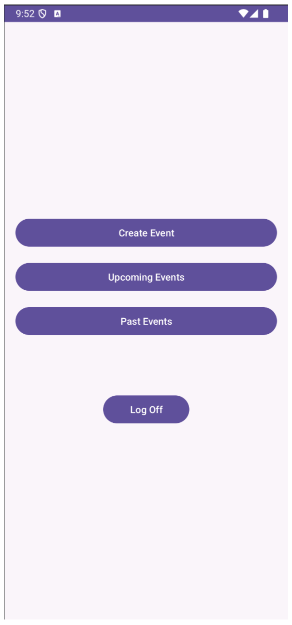
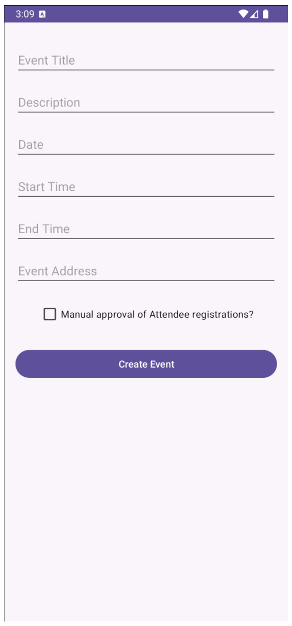


Figure 8: Main page for organizer


3:09

Event Title

Description

Date

Start Time

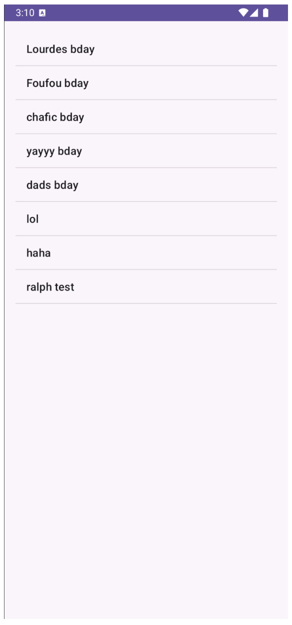
End Time

Event Address

Manual approval of Attendee registrations?

Create Event

Figure 9: Create event page for organizer


3:10

Lourdes bday

Foufou bday

chafic bday

yayyy bday

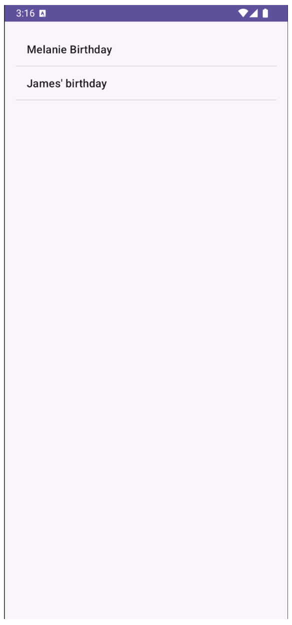
dads bday

lol

haha

ralph test

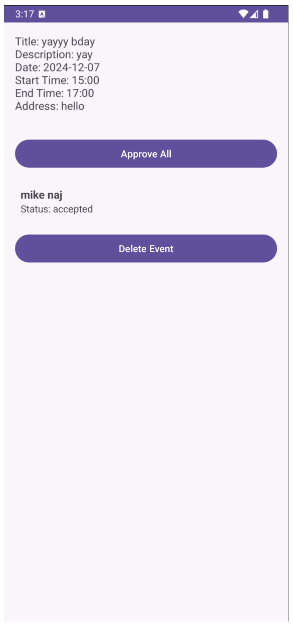
Figure 10: Upcoming events page for organizer


3:16

Melanie Birthday

James' birthday

Figure 11: Past events page for organizer


3:17

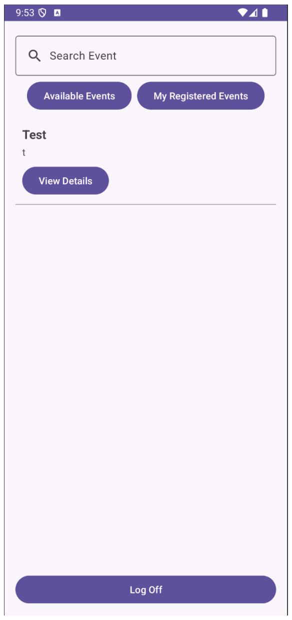
Title: yayyy bday
Description: yay
Date: 2024-12-07
Start Time: 15:00
End Time: 17:00
Address: hello

Approve All

mike naj
Status: accepted

Delete Event

Figure 12: Event detail page from organizer's POV


9:53

Search Event

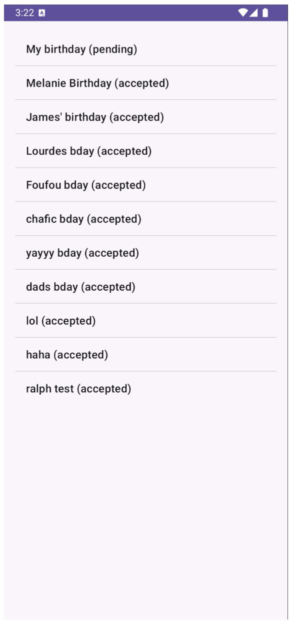
Available Events | My Registered Events

Test
t

View Details

Log Off

Figure 13: Main page for attendee


3:22

My birthday (pending)

Melanie Birthday (accepted)

James' birthday (accepted)

Lourdes bday (accepted)

Foufou bday (accepted)

chafic bday (accepted)

yayyy bday (accepted)

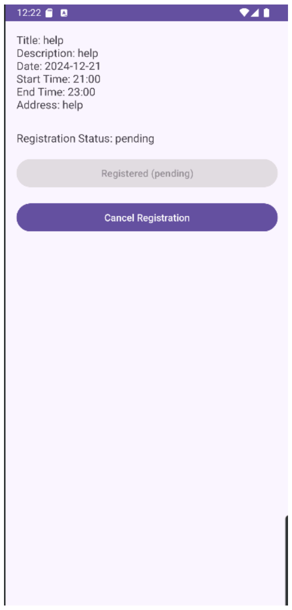
dads bday (accepted)

lol (accepted)

haha (accepted)

ralph test (accepted)

Figure 14: Registered events page for the attendee


12:22

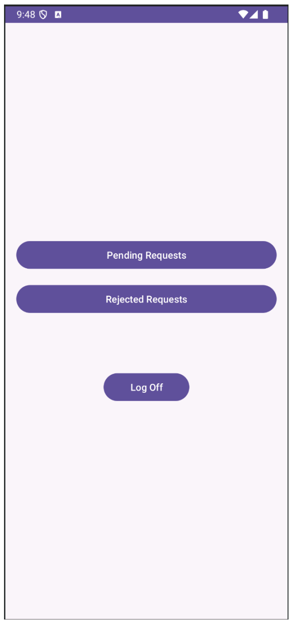
Title: help
Description: help
Date: 2024-12-21
Start Time: 21:00
End Time: 23:00
Address: help

Registration Status: pending

Registered (pending)

Cancel Registration

Figure 15: Event detail page from attendee POV

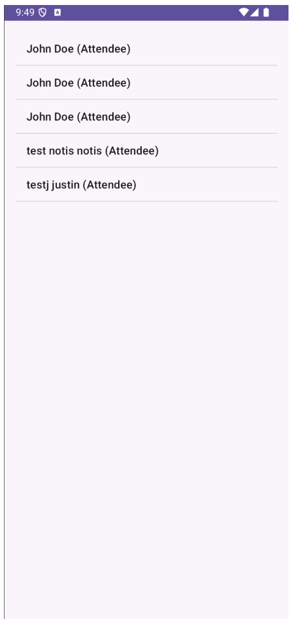

9:48

Pending Requests

Rejected Requests

Log Off

Figure 16: Main page for admin


9:49

John Doe (Attendee)

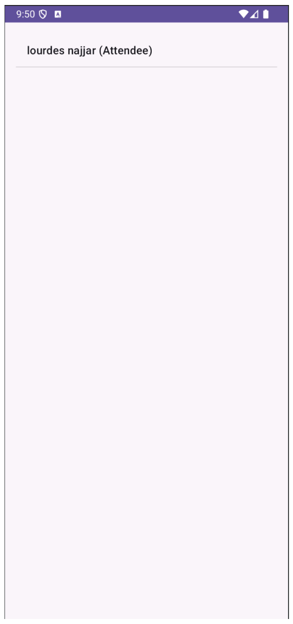
John Doe (Attendee)

John Doe (Attendee)

test notis notis (Attendee)

testj justin (Attendee)

Figure 17: Pending requests from admin pov


9:50

lourdes najjar (Attendee)

Figure 18: Rejected requests from admin pov

Lessons Learned

The development of the EAMS provided us with an invaluable opportunity to apply software engineering principles seen in class in a practical context. Throughout the project, our team encountered challenges and successes that shaped our understanding of technical and collaborative aspects of software development. Here are a few key lessons learned:

1. **Importance of incremental/progressive development**
   Breaking down the project into smaller, more manageable tasks allowed us to realize the importance of focusing on specific features in each phase. Although the deliverable requirements for this project were already set up for us to guide us in deciding which tasks to tackle first, we were able to grasp this concept to use it in future projects, where we will create our own specific deadlines. In other words, this approach enabled us to prioritize tasks, test individual functionalities, and slowly integrate them into our cohesive system.

2. **Error handling and validation**
   Field validation was a crucial step that kept appearing in each deliverable. This recurrence emphasized the importance of defensive programming and anticipating user behavior. Throughout the project, we implemented rigorous field validation to check for common issues, such as missing information, invalid email formats, or passwords that did not meet security requirements. For example: input fields, such as email addresses, were validated against regular expressions to ensure proper formatting. Also, phone numbers and addresses were checked for logical constraints. Passwords were required to meet complexity criteria, such as length and character variety. This approach not only prevented the system from receiving invalid data but also enhanced the user experience by providing actionable feedback.

3. **Database integration and data management**
   Integrating data seamlessly ensures consistency and reliability. Proper schemes and UML design, along with efficient query optimization, helped improve performance. Fata security measures like encryption and access control are crucial for safeguarding user information. Regular backups and error handling ensure the system remains robust and recoverable.

4. **User-centric design**
   We concluded that a user-centric design requires a deep understanding of the user's needs. Creating clear personas and gathering user feedback from each other helped prioritize features and improve usability. Testing revealed areas for improvement, emphasizing the

importance of continuous iteration. Designing for accessibility from a user's point-of-view also proved essential, ensuring the app is usable by all.

5. **Time management and project scope**
   Managing time and our project scope was a valuable learning experience. In terms of management, we realized the importance of prioritizing tasks and setting clear milestones to stay on track. It was also crucial to build in some buffer time for unexpected delays and to keep communications open within the team to address issues quickly. Regarding the project scope, defining it clearly from the start helped prevent feature overload and ensured that the team stayed focused on key objectives. Flexibility was important as it allowed us to adapt when needed without straying too far from the initial plan. Regularly revising the scope and staying in touch with each other helped keep everything aligned with the project's goals.