COLORADOSCHOOLOFMINES
EARTH • ENERGY • ENVIRONMENT

CSCI 370 Final Report

Faith Connect

Elijah Sage
Jordan Nguyen
Sonia Vidrio

December 6, 2024

CSCI 370 Fall 2024

Ms. Kristen Peglow

Table 1: Revision History

Revision	Date	Comments
New	August 26	Created document and added Introduction, and Requirements sections
Rev – 2	September 19	Added System Architecture sections
Rev – 3	October 20	Added Software Test and Quality as well as Ethical Considerations
Rev - 4	November 10	Added Results
Rev - 5	December 6	Added Feedback

Table of Contents

I. Introduction

Faith Connect was designed to help Muslim students build a strong sense of community and to connect with others for prayer and other activities. The development of this app was deeply informed by the voices of the Muslim community at Mines, whose thoughts and concerns were carefully considered through a series of interviews. This feedback has shaped the app's mission to create a platform that facilitates prayer gatherings and strengthens the bonds of faith among Muslim students.

II. Functional Requirements

Faith Connect is designed for users to create accounts, organize events, and connect with others in their faith community.

Account Creation:
Users can set up personal profiles, share information about their religious affiliation, and choose a profile picture.

Creating Events:
Event creation options allow clubs to organize public events or private gatherings. Each event includes a description, location, and time, ensuring attendees have all the relevant details.

III. Non-Functional Requirements

For a positive user experience, Faith Connect has a user-friendly interface that is easy to navigate and will keep information secure.

User-friendly Interface:
A clean, simple design makes navigation easy for users. Designed with clear labels and straightforward menus to ensure accessibility for all users.

Security:
The app features a basic user authentication system, including secure password storage and encrypted user data. A significant priority is the protection of user data to maintain trust and safety.

Basic Error Handling:
Basic error handling is implemented to ensure the app remains responsive. For instance, if login attempts fail, clear messages such as "Incorrect password, please try again" will guide users to resolve the issue.

IV. Risks

Given the app's focus on multiple religious communities, security and privacy are critical, especially in the current social climate. Lack of prior experience with app security and data privacy poses risks that must be mitigated carefully. Faith Connect will only share user's data such as events they plan to attend and offers visibility controls to events and posts. Not to mention the added privacy by making sure only friends can message each other.

Since Faith Connect is a social media app, there are inherent risks associated with the attendance of events. Due to the nature of making an event, there is always the risk of someone using that given information with ill intentions in mind. This is a safety concern especially since our app will have minorities gathering in one location. To mitigate this, only friends of a user can see the events they will be attending. Granted this would be after the friendship is formed through the app.

Moreover, unfiltered communication in posts or direct messages poses another risk we have to consider. Users may unintentionally or intentionally post or send messages that are offensive, disrespectful, or otherwise inappropriate. This could harm the community's inclusive and respectful atmosphere.

Another challenge is our limited experience in building social applications from the ground up. This is due to how our project differs from the rest of the projects since we will not be implementing a project for a company. To address this, extra effort is needed for skill-building, particularly with tools like Swift, which will help us overcome initial learning curves and ultimately improve project outcomes.

V. Definition of Done

The definition of done for this project will be a functioning social media platform for people of various faiths to connect through features such as posting events, messaging friends, and being able to scroll through the feed and RSVP for upcoming events. The social media app should be functional but also have a proper data warehouse to store the data in. Given time constraints, a stretch goal is to implement an algorithm to analyze profiles and suggest friend connections.

VI. System Architecture

The app is built using a client-server architecture with Firebase as the backend service provider. This allows the app to provide features such as user authentication, real-time database updates, and cloud storage for posts. The database design consists of key tables: users, events, and friendships. The Users table contains user-specific information such as religion and email. The Events table stores details about community events, including the title, date, location, and such. The relationships between these tables are depicted in Figure 1.

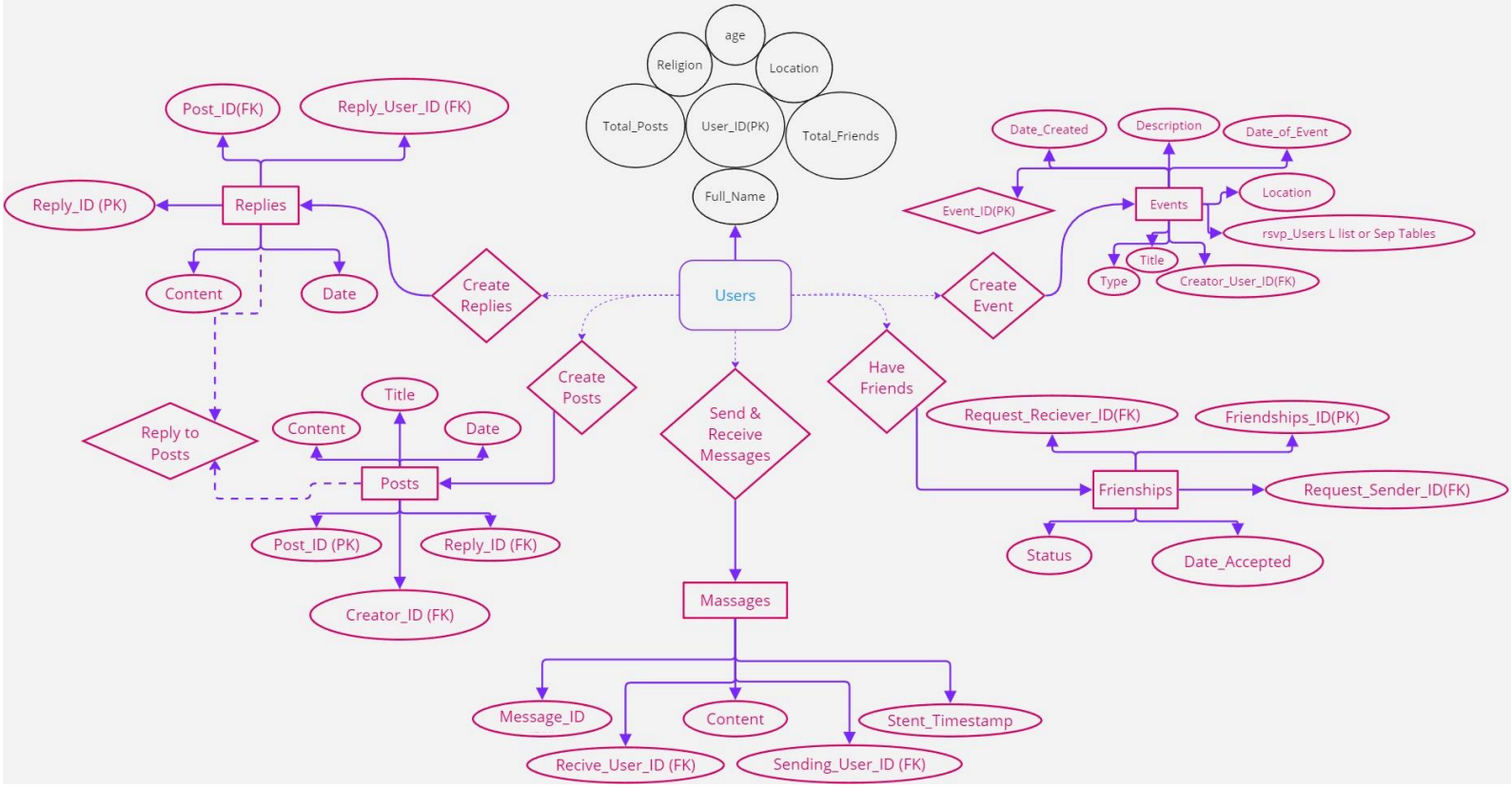


Figure 1: ERD Diagram

As for the data flow, the app begins with registration. This is where users provide their religion and personal details like email addresses and passwords. Upon logging in, users can manage their profile, view or accept friend requests, send messages, etc. With the different tabs, users can then choose a certain tab to be able to view their events and their friends. Navigation between screens can be seen in Figure 2.

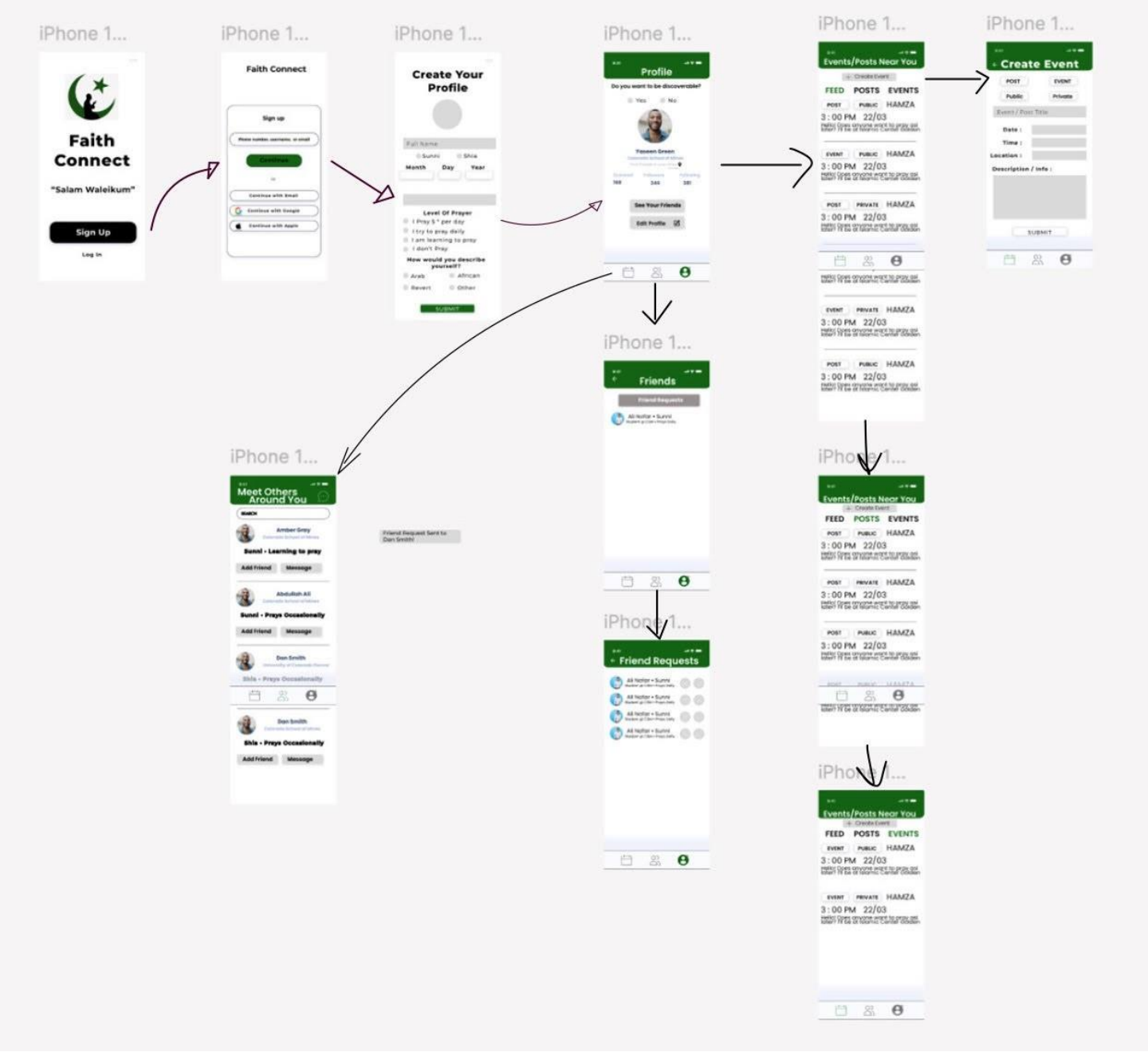


Figure 2: Figma (Data Flow)

ISSUES

Elijah and Sonia both have school computers that need admin permission to be able to install the Xcode application. This took away from valuable time that could have been spent coding. Luckily the IT department was able to grant them access within 2 days of the issue. Another issue has been choosing a database. Originally, planned on using Snowflake to store data since Elijah is familiar with it through his work. Due to this plan, no time was invested in researching other alternatives. Upon trying to make a snowflake account that wasn't linked to Elijah's work, the high cost of an account led to a change of plans which led to a switch to Firebase. Although discouraged at first because of the lack of familiarity, Firebase has worked perfectly so far.

VII. Software Tests and Quality

To ensure good quality software in our app, several testing strategies were implemented as well as code reviews.

For unit testing, which will test the functionality of the code, unit test functions in Xcode were created for each core function of the app. Including account creation, sign-in, message exchange, event creation, RSVP-ing, and adding friends. Previews were also included to test isolated views. The example previews can be found below:

Sign-up preview

Log-in preview

For user interface testing, interface test functions were implemented in Xcode. Some examples of this would be a specific test to see if the app works when users tap on the screen, as well as another test to see if users can type in text boxes and text fields. When giving demos through the XCode simulator to both our client and advisor, they gave feedback. Our client, who is a professor at Mines and has experience with user interfaces, gave suggestions on the overall design in the sign-up view. Our advisor suggested additional tests, such as when a user clicks a button multiple times.

Integration testing will focus on verifying seamless real-time data exchange with Firebase. This includes comparing real-time updates and ensuring smooth user authentication flows. We will confirm that users can sign up/log in and log out properly, with persistent sessions across app relaunches. CRUD (create, read, update, delete) database operations will be heavily tested to ensure reliable data management within the app.

For code reviews, pair programming was done, and SwiftLint was used to maintain code quality. SwiftLint enforces formatting standards, such as correct indentation, line breaks, and spacing. It also ensures naming conventions are followed, like camelCase for variables and PascalCase for classes. SwiftLint can also detect potential issues such as force unwrapping, which can lead to runtime crashes, and suggest safer alternatives.

VIII. Project Ethical Considerations

One of the most significant concerns for our app is the protection of user privacy (ACM principle 1.6). This is due to how our app will consist of users of religious communities that are susceptible to hate crimes and discrimination from other users. To reduce these risks, we prioritize safeguarding personal information and ensuring that users have control over their data. We will allow users to have the ability to choose whether their profiles and events are public or private, which goes along with ACM principle 1.7 (Honor Confidentiality). This provides users with flexibility to manage their visibility and helps protect them from potential harm. We also added the add friends feature so users can choose who can message them and invite them to events.

Despite these precautions, there is always a possibility that users could misuse the platform or that harmful interactions could occur. While we will implement a reporting feature to allow users to flag inappropriate content or accounts, we cannot guarantee that all users will be shielded from hate speech or harmful behavior. Nevertheless, we are aware that users are putting a great amount of trust in us, and we want to be sure of being able to hold up to their expectations as we actively try to protect their information.

IX. Results

As mentioned earlier, most testing was done with the Xcode simulator and views of the app. Demos were then done with both our client and our advisor. All unit testing and interface testing have been completed, and all features are now fully operational, and all features work seamlessly together.

As of the end of the semester, the app now supports user account creation, with the necessary information like an email address, password, and religious affiliation. They can upload profile photos which are stored using Cloud Storage for Firebase, which is built on fast and secure Google Cloud infrastructure. Clubs can make an account specifically for their club, to make events or announcements through posts. Clubs need verification to be able to make a club account. While users can make posts, they won't be able to make events due to security reasons. Users are also able to send friend requests once the requestee confirms they are now friends on the app and can see each other's posts. Users can view the profiles of their friends which includes their friend's posts, or events if they are viewing a club profile. Users can also direct message their friends or unfriend them if they desire.

We've also implemented an RSVP feature, where users can RSVP to events. And then those events will show up in the built-in calendar that has been integrated. In the calendar view, dates that users have RSVP'd for are highlighted in red. When the users click on that date the information regarding that event will appear. The Friends tab has also been completed where a user will see who they are friends with and be able to message them separately as well as search for friends.

The final feature is the Discover page. Which is a feed of all clubs that are affiliated with the same religion as the user will show up. When a user clicks on a certain club, it will lead them to a list of the club's members as well as a button to request to join the club. One of our non-formal clients, Camila who pitched the project idea, found this new addition ties in with her original vision of the app and is satisfied with how it turned out.

Overall, we have created an application that allows religious individuals to message one another, RSVP, attend, and create events, which ultimately allows people of Faith, to Connect.

X. Future Work

Some features that couldn't be implemented due to time constraints are as follows:

- **QR codes**: Enable faster friendship requests or form friendships using QR codes.
- **Report accounts**: Implement a feature for users to report accounts.
- **Admin management**: Develop a system for admins to manage user reports.
- **Club suggestions**: Create an algorithm to suggest clubs to join based on factors like religion, friendships, and college campuses.

The app could also potentially be made to be available through the App Store, but with this comes some additions and changes. As the app stands now, it would need a more dynamic database to be able to handle increasing numbers of users. Upgrading to better hosting would significantly enhance performance, scalability, as well as reliability. This would provide support for future growth. The app will also need to implement more features such as content moderation, as well as modify the app a bit to be able to be compatible with Android.

It would also be good to talk with a legal team if the app were to get on the App Store since the target audience is people of different faiths and could be discriminated against. Not to mention that there are going to be events that are planned through the app. It would also be beneficiary due to app store policies, and any state policies as well as being able to draft Terms of Service and User Agreements.

Implementing advertising would be essential for driving user growth and increasing engagement. Advertising allows the app to reach target audiences, who may benefit from Faith Connect. With that also comes the engagement of other college campuses, especially with incoming students to be able to form a better sense of community at their campus since adjusting to a new environment can often feel isolating. Especially if they seek to maintain or deepen their spiritual connections.

XI. Lessons Learned

Developing Faith Connect from scratch provided the development of technical skills, where we learned about database selection, feature prioritization, and the importance of thorough initial planning. First, selecting an appropriate database is critical, especially with considerations like storage for information like accounts but also for photos such as profile photos.

We also learned that with a project as dynamic as this one, focusing too heavily on specific features early on can be counterproductive. Initially, we invested significant time building separate views for friends and messages, only to find that combining them into a single view was more efficient. This experience taught us the value of prioritizing core functionality first, ensuring features are in place before diving into enhancements or intricate design details.

Another key takeaway is the importance of a well-thought-out database structure. Our team had to adjust table columns multiple times. Such as adding an "end time" column for events in addition to a "start time" and frequently modifying fields in the user profile table. This process emphasized that investing more time upfront in designing tables and columns would have saved us from time-consuming revisions later. In hindsight, spending at least a week brainstorming and planning the database schema would have provided a solid foundation to accommodate future changes more smoothly. These lessons have underscored the value of early-stage planning and flexibility in building a scalable app.

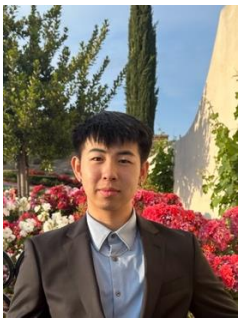
XII. Acknowledgments

We would like to express our gratitude to our client, Rob Thompson, and our advisor, Kristen Ingram, for their invaluable support and guidance throughout the development of our app. Their insights were instrumental in refining both the structure and testing of our app. Allowing us to create a more user-friendly and efficient product. They consistently offered constructive feedback, ensuring that our app meets high standards of functionality and usability. We are immensely appreciative of the thoughtful input they provided at every stage of the process. Thank you for being such an integral part of our journey.

XIII. Team Profile

Team Profile

Name: Elijah Sage
Major: Computer Science
Work Experience: TEAM Services Group, Data Engineering intern (current), BNSF Railway, Transportation Intern (Summer 2023)
I am excited to create a working app from scratch. I look forward to learning about mobile app development and working with my team to achieve our goals!



Name: Jordan Nguyen
Major: Computer Science
Work Experience: Raytheon Software Engineer intern (current), CU Denver D Incubator Backend intern, IgniteXL Data Analytics intern
Can't wait to be able to create a mobile app from scratch. This will be a very interesting and fun project, and I am really looking forward to it! I hope to learn a lot from mobile app development and be able to meet the requirements and create a product for a client.

Name: Sonia Vidrio
Major: Computer Science
Work Experience: Code Denver Advisor
I'm thrilled to be working on this app, not only because it allows me to create something meaningful, but also because it's a fantastic opportunity to gain valuable experience and grow my skills.

Appendix A – Key Terms

Include descriptions of technical terms, abbreviations, and acronyms.

Term	Definition
Xcode	*An integrated development environment (IDE) developed by Apple for creating software for MacOS, iOS, etc. It provides tools for coding, testing, and debugging applications as well as interface design.*
Firebase	*A platform for mobile and web applications, providing real-time data exchange and authentication solutions*
Swift	*A modern programming language developed by Apple*
SwiftLint	*A tool for Swift code quality ensuring formatting consistency and detecting potential coding issues*
ACM Principles	*Ethical standards by the Association for Computing Machinery, guiding developers on topics like user privacy and confidentiality*