Get Out Official Project

Brett Bono and Michael Gerber

Published June 2025, Official Project Start May 2024

Contents

1 The Idea

1.1 The Start

During my freshman year of college, I wasn't very involved in extracurricular activities. As a newcomer, I wasn't fully aware of the clubs, events, and opportunities available both on campus and in the surrounding area. I was also interested in exploring the local nightlife, but my main sources of information were a few friends I had made early on, who, like me, were still figuring things out. As a result, I often missed out on events simply because I didn't know they were happening.

One day after a computer science class, an idea struck me. I realized I wasn't alone, many students, both new and returning, struggled to stay informed about what was happening around campus and in the local community. That moment of clarity sparked the idea for a simple, location-based event app designed to be intuitive and accessible for students like me. At the time, I was new to programming and wasn't entirely sure how to bring the idea to life, but the concept stayed with me.

During my sophomore year, I began learning React, a popular JavaScript library for building user interfaces. Not long after, I discovered React Native, a framework that enables developers to build mobile apps for both iOS and Android from a single codebase. That discovery was a turning point. I dove into the documentation and quickly realized that React Native was the ideal tool for bringing my idea to life.

By the end of my junior spring semester, I had a much clearer vision of the app I wanted to build: a social platform centered around events rather than traditional social media content. Initially, my focus was on nightlife, but I soon saw the potential for the app to scale and serve a wide variety of interests and user groups.

The core concept was to build two connected applications sharing a common backend, similar to how DoorDash operates with both a customer app and a business app.

The user-facing app would help individuals discover events nearby. Users could explore events on a map, join groups, add friends, and share experiences. They could also create their own groups and host events within their communities.

The business-facing app would serve local organizations and venues. It would allow them to manage groups, host larger events, and access aggregated event data to guide future planning. While protecting user privacy, the app would provide insights into event performance and offer suggestions to help businesses stay competitive and engaged with the local scene.

In short, I set out to build a dynamic, real-time event marketplace, one that empowers individuals to connect with their communities and helps businesses grow through meaningful, data-informed engagement.

Below is a basic Idea of how the User Side of the application worked, The business side unfortunately was never gotten to. We prioritized getting the user application done before the business side.

1.1.1 User Side

USER ONBOARDING: When a user launches the app, they can log in or create an account. I want this process to be seamless and user-friendly while ensuring strict controls to prevent bot account creation.

Map-Centric Experience: After signing in, the default screen is a map centered around the user's location. This map dynamically updates based on user activity. From the start, it displays real-time public events—anything from fundraisers to karaoke nights at a bar.

Connecting with Others: Users can add friends, send friend requests, and join or create groups.

Friend Interactions: Adding a friend works much like Snapchat: users can direct message friends with photos, texts, videos, share their location, and share events. Group chats are also available (with limited features compared to dedicated groups).

Groups: Users are encouraged to join groups, which can be either public or private. Groups can be anything—corporate teams, social clubs, or hobbyist groups. To join a private group, users must request

access and get approval from someone with the appropriate permissions. Public groups can be joined freely.

Creating Groups: Users can create up to three groups. By default, each group starts as private and requires a minimum number of members to become public. Groups have a permission structure similar to Discord, allowing owners to assign roles like manager, co-owner, or customize specific permissions (e.g. approving membership requests).

Events: Each group can host public or private events, regardless of the group's public/private status. Private events are only visible on the map to group members.

Event Discovery: The more groups a user joins, the more events they'll see on their map, enhancing discovery and engagement.

Overall My goal is to make this app a real-time marketplace for events, encouraging small businesses to advertise and grow their communities through the platform.

1.1.2 Business Side

Left Blank For Future Use

2 Recruitment Process

At the time, this project was highly ambitious given the skills I had. I knew I would need help to make substantial progress within the timeframe I had in mind. Working with a very limited budget, I decided to reach out to one of the computer science professors and asked if they could forward my message to all computer science majors. My goal was to find others who, like me, were eager to gain experience and build something meaningful. It was a win-win: I would get help building the app, and the people who joined would gain real-world experience and something valuable to add to their resumes.

The outreach turned out to be a success. Within a week and a half, I received emails from about 10 to 15 students expressing interest in the project. Most of them were current CS majors, but I also heard from a couple of JMU computer science alumni. The next step was to narrow down the group to those who were most committed and aligned with the project's goals.

After initial interviews, I decided to test each person's responsiveness by assigning them a small coding task with a rough deadline. This helped me identify who was truly motivated to stick with the project. Ultimately, the group narrowed down to one developer I really wanted to work with, a fellow sophomore who showed genuine interest and enthusiasm.

We agreed to keep the development team small to stay agile in the early stages. He focused on front-end development while I handled the backend and guided the overall direction of the app. I made a point to value his input, remain open to his ideas, and treat the collaboration as a partnership built on mutual respect.

3 User Interface

The initial user interface starting with the login screen and homepage was something I began working on about a year before officially starting the project. My goal at the time was to get comfortable with React Native's syntax, explore different database options, and experiment with commonly used React Native libraries. I spent about two weeks on it before stepping away due to school becoming overwhelming.

When Michael joined the project, I walked him through what I had built so far. He was sharp and quick to learn, and I trusted he could get up to speed within a week or two—which he did. Around that same time, a friend recommended I use Jira, a project management platform, to help organize our goals and track progress. I created a basic roadmap outlining the key features we needed to build and broke them down into steps. From there, we followed the roadmap closely, one step at a time.

Since this was my first time structuring an application from the ground up, I approached the process with flexibility, treating it as trial and error in many ways. At the same time, I understood how crucial it was to organize our codebase well to ensure long-term maintainability and team efficiency.

Note: Most of the UI components were not finalized at this stage. We had plans to add smooth animations, improved theming, custom-styled components, and eventually, a fully customized map experience.

3.1 User Flow

Below is the user flow we developed before putting the project on hold due to the start of the school semester.

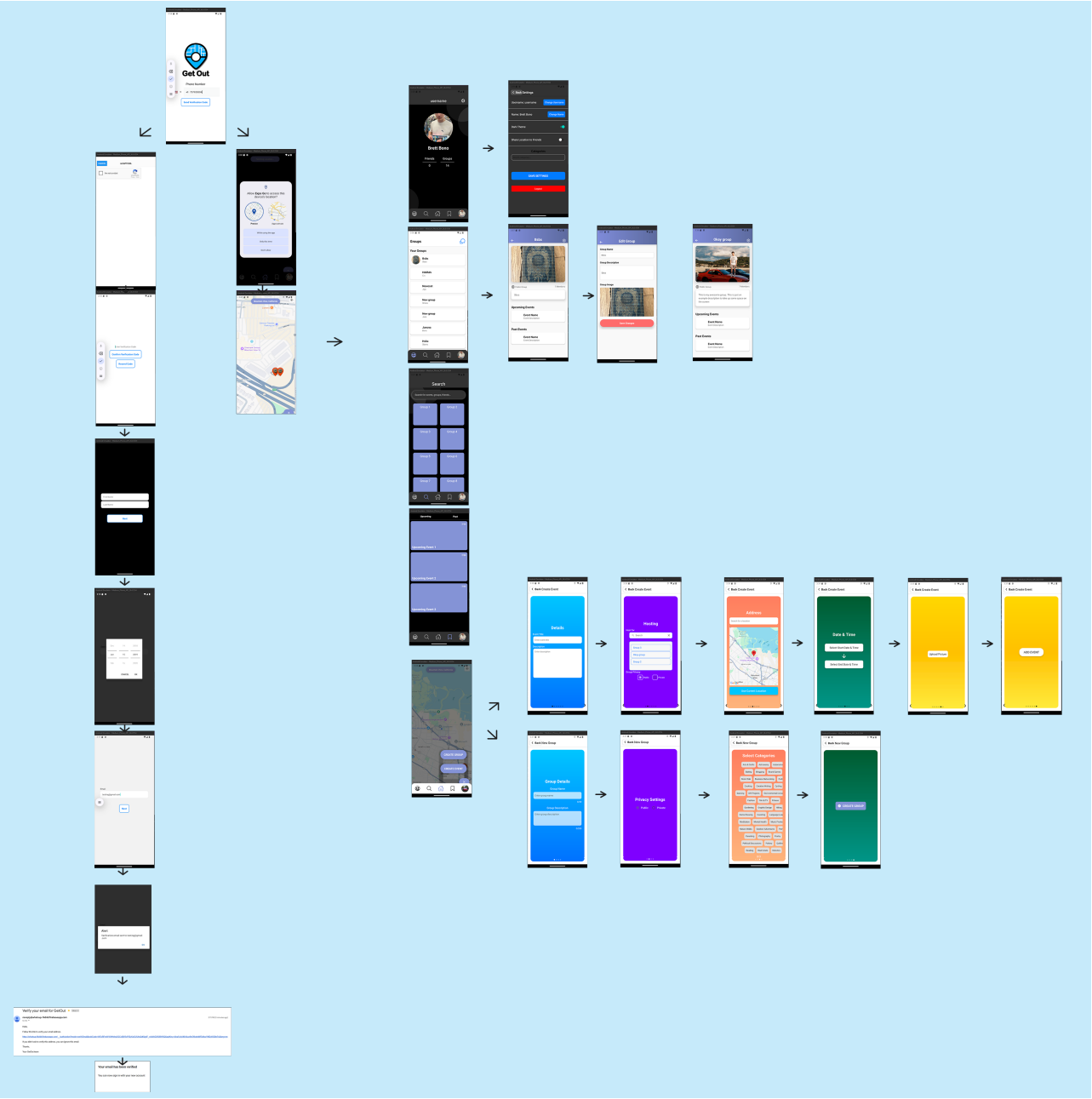


Figure 1: UserFlow Diagram

3.2 Account Creation / Login Screens

To begin building any type of profile-based app, it's essential to set up account creation and login functionality. Each user sets up an account, and each account is typically identified by a unique string (such as a user ID) assigned by the system. This unique string allows the app to link each user's data and actions to their account. In our project, we used Firebase Authentication, which automatically generates unique user IDs for each account upon registration. This user ID is securely stored and mapped to each user's profile data in the database. While Firebase does not explicitly use a "hashing algorithm" for user IDs, it handles all the necessary security and uniqueness behind the scenes.

Our account creation screens were inspired by common user flows found in popular social media platforms. To initialize an account, we collected essential user information: first name, last name, phone number, email address, and date of birth. We implemented phone number verification to ensure authenticity and stored validated numbers in a secure database to prevent spoofed accounts. Additionally, we incorporated Google's CAPTCHA as a second layer of defense against automated bot registrations. Email verification was also required to confirm the user's identity. Finally, collecting date of birth allowed us to filter out certain events or features for minors, ensuring compliance with age restrictions and safety policies. In the backend of things I setup an account creation timestamp and initialized other information in our database, like user-type for scalability, and another timestamp telling up when a user last updated their profile.

There were many revisions done to the UI and the backend of the login pages. Since we were using‘ firebase we needed to store data in two separate places, the authentication information went through Firebase's authentication API, and the other information was sent to Firebase's firestore database. The data however was structured in such a way that access to one database vs the other was fast and not a hurdle at all.

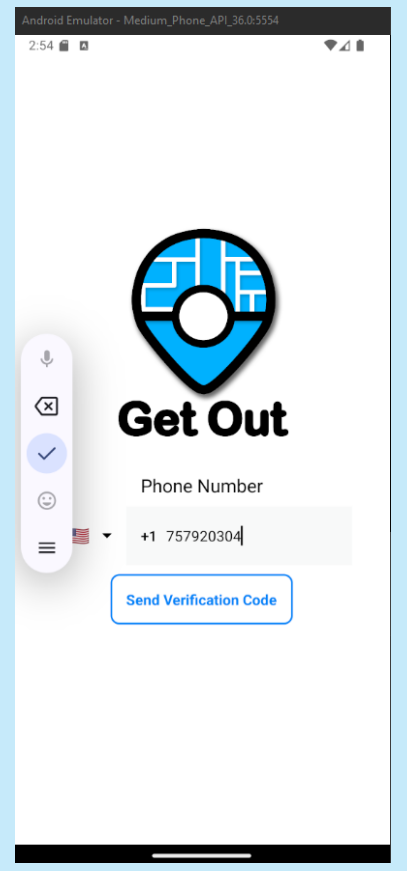


Figure 2: Application Launch Screen 1rst time. (We do have state persistence working)

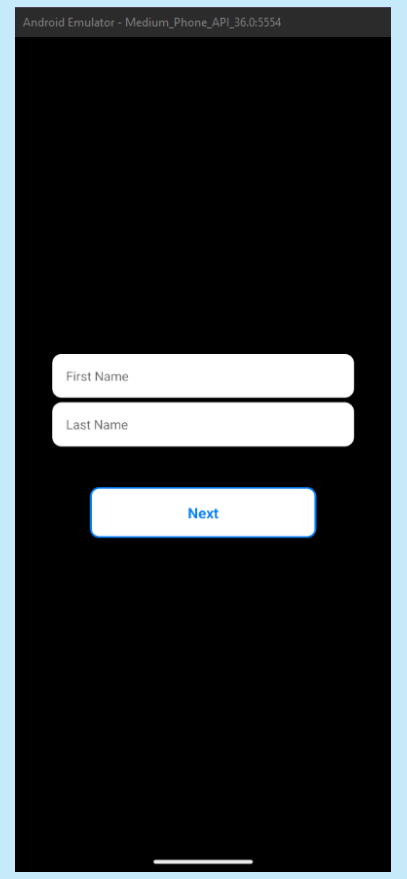


Figure 3: First Name, Last Name

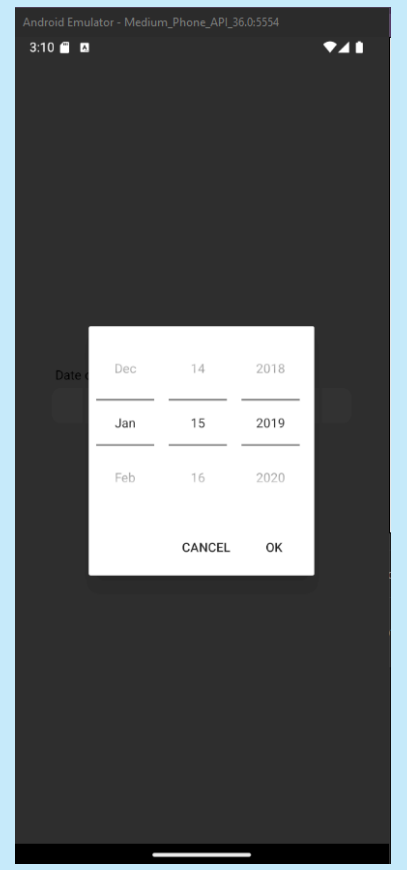


Figure 4: Date Of Birth

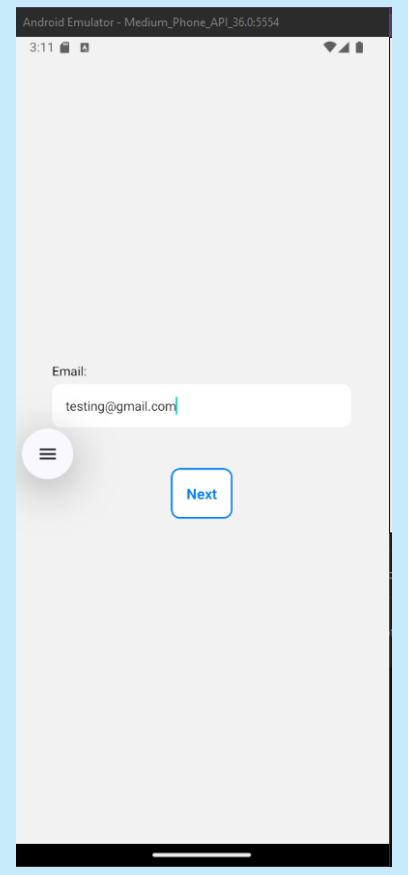


Figure 5: Email

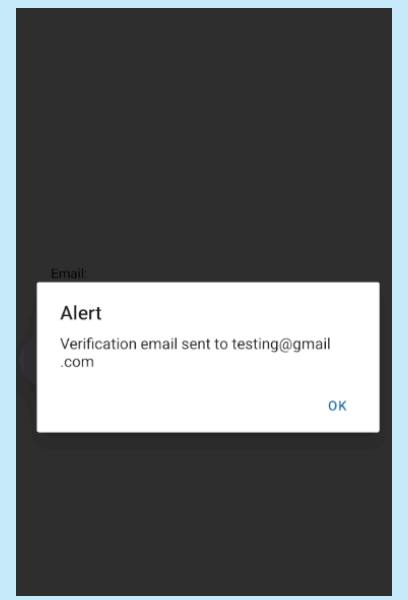


Figure 6: Verification

Figure 7: Verification

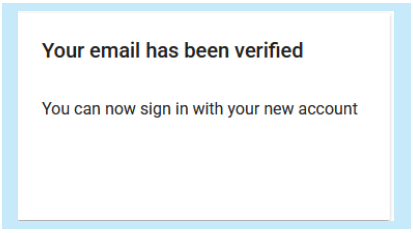


Figure 8: Verification Confirmed

3.3 Home page

After the user successfully creates their account, a listener, flips a switch and redirects the user to a new set of screens independent of the set of login screens. The user is asked for permission to share their location. If the user denies, the app is not very functional. If they accept they are redirected to the default UI, which is the default map application their phone uses. If its an apple device its defaulted to the apple map, if its an android its defaulted to a google map. Images of our homepage UI iterations can be seen below.

There are only 3 interactive buttons on the default map screen all under the "+" icon, which expands to the create group, and create event button.

Also visible from the majority of of logged-in screens, are the navbar buttons, which redirect to the friends-screen, the groups-screen, search-screen, home-screen, events-screen, and profile-screen.

Below is our progression of the home screen

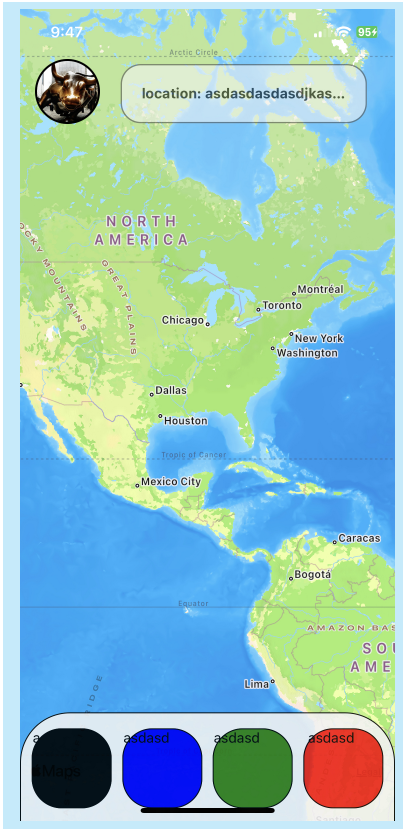


Figure 9: Initial User Interface I put together a year before officially starting the project, everything, from the colors to the "Location: .." are all place holders

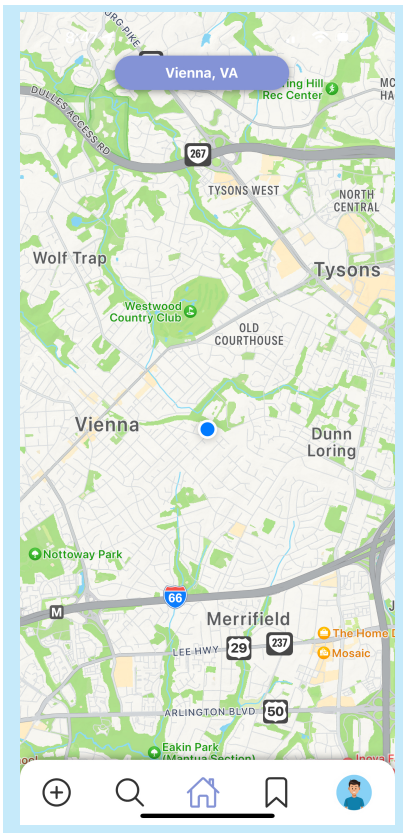


Figure 10: Second Iteration of a User Interface with little backend development

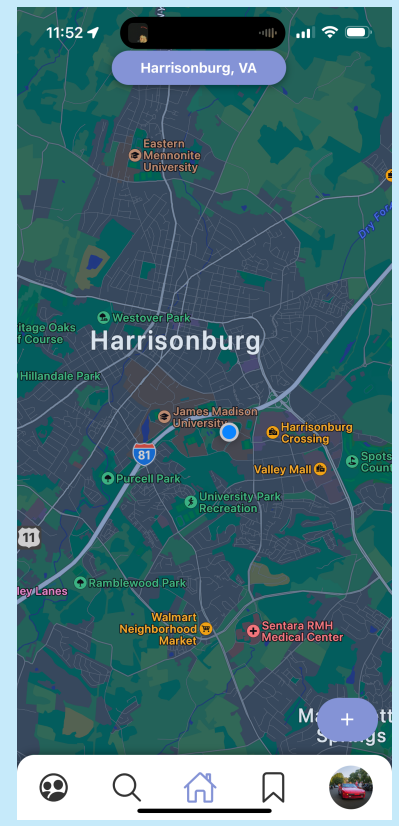


Figure 11: Final Version W/ Out Events registered — Rendered On Expo with IOS

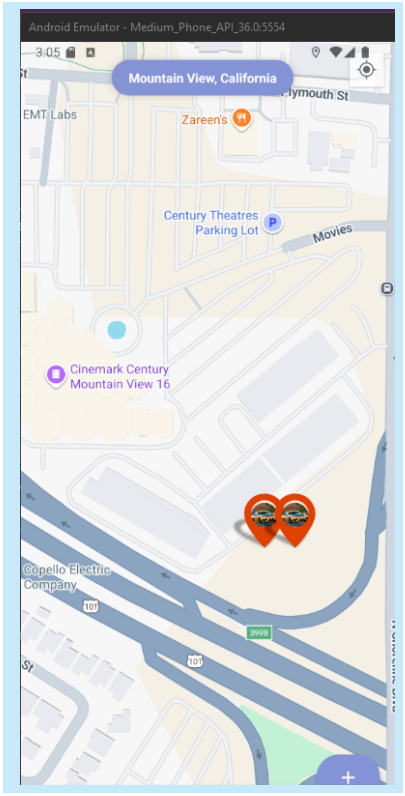


Figure 12: Final Version w Events registered. Rendered On Expo with Android Simulator.(Accidentally have navbar cut off in the screenshot)

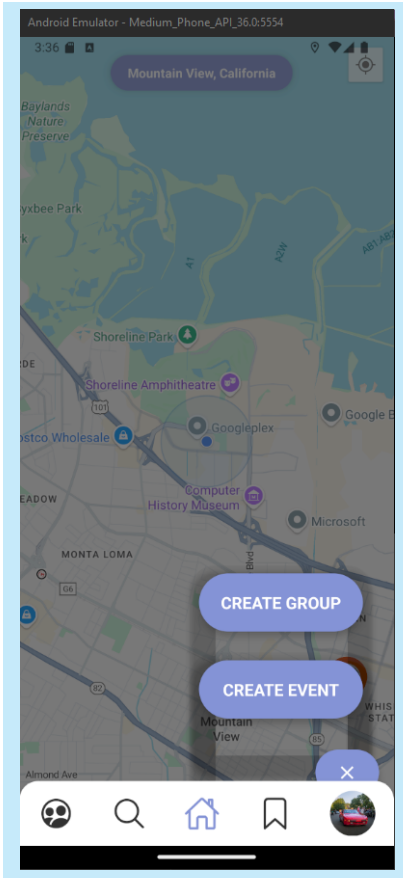


Figure 13: Create Group and Create Event Popup Buttons

3.4 Groups

Off of the Navbar is the groups button, This navigates the user to the list of groups that the user is in, displaying the group Image, name, and the group bio. Also, on the top right of this page is a messenger button which contains friend chats, and groups chats.

Once the user clicks a specific group it pulls up the group page which shows the groups upcoming events and past events.

Also on this page is a group editing button, where if a user has the required permissions, they can edit group information and also give roles or do anything else group related.

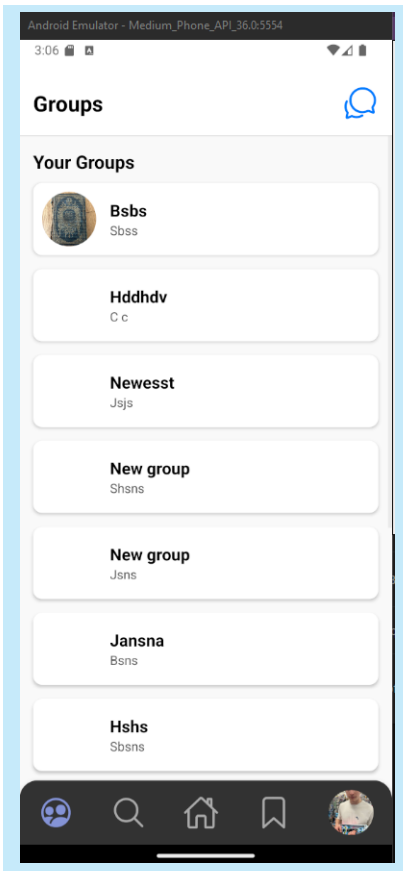


Figure 14: Users Groups

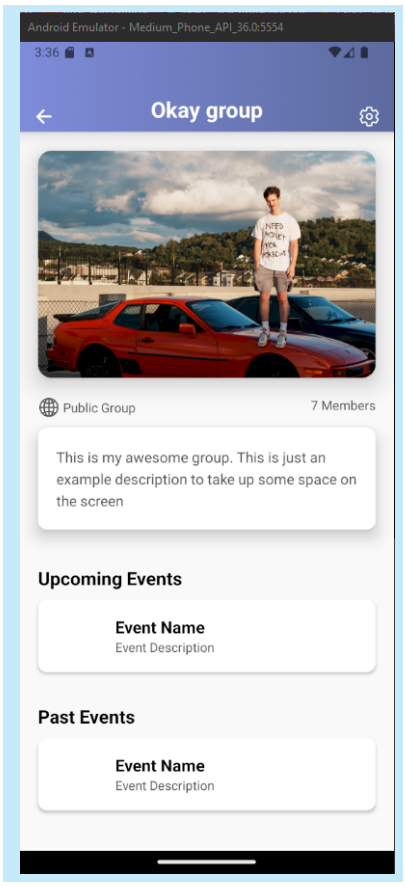


Figure 15: User Selected Specific Group

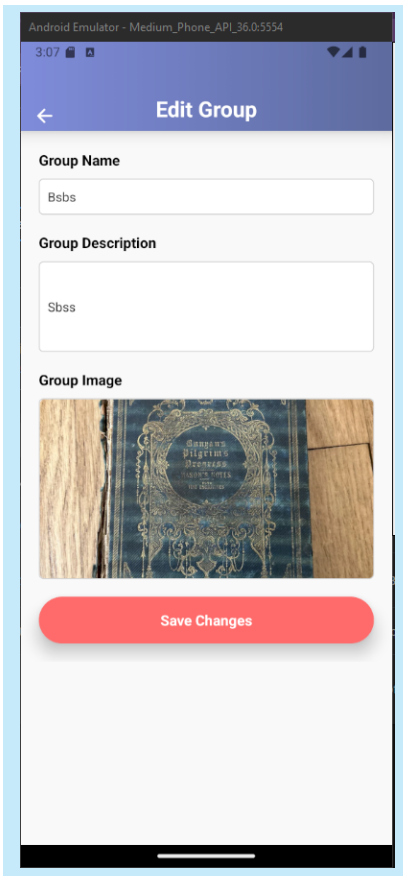


Figure 16: Edit Group Page, Only Users With Permissions Can Edit

3.5 Universal Search

One of the key features we designed was the universal search button, accessible directly from the navbar. This feature would allow users to search for events, groups, or friends from a single interface. We planned to include three basic filters to help users distinguish between different result types more easily.

As the person handling the backend, I found this to be one of the most technically challenging parts of the project. Unfortunately, we had to put development on hold before the universal search was fully implemented. My vision was to build a search experience similar to Instagram's, which is fast, intuitive, and context-aware. I wanted to implement a priority system where mutual friends, groups, and nearby events would be surfaced first, creating a smarter and more personalized result set.

Designing a search algorithm that could handle three distinct data types while respecting a priority hierarchy was quite complex. I experimented with various libraries and even revisited some of my old numerical algorithms books to explore whether concepts like Singular Value Decomposition could be applied here.

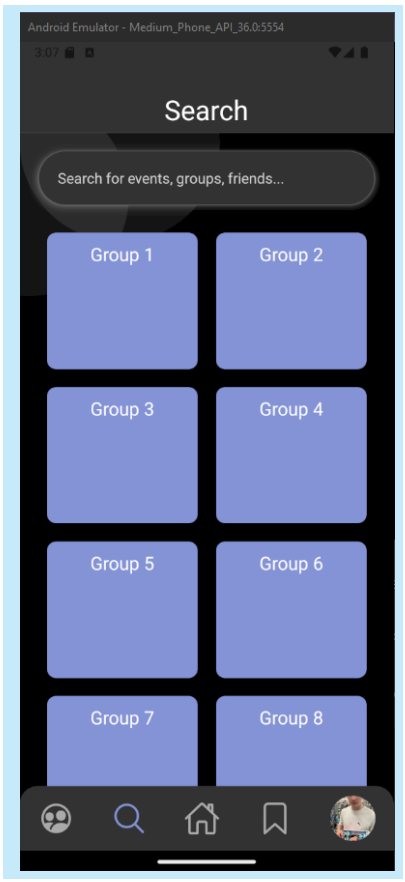


Figure 17: User Settings with DarkTheme Enabled

3.6 Upcoming Events

Off of the Navbar is the Upcoming Events button. This page is a list of events that are bookmarked. It displays the events that are upcoming and events that have already passed.

Figure 18: User Settings with DarkTheme Enabled

3.7 Profile

Accessible from the Navbar, the Profile button takes users to their personal dashboard, where they can view the groups they're part of, check their friends list, and access user settings.

Within the User Settings tab, users can edit their username, update their display name, customize the UI theme, and manage location sharing preferences. Currently, location sharing is a simple on/off toggle. However, we plan to change this feature by allowing users to choose which specific friends they'd like to share their location with.

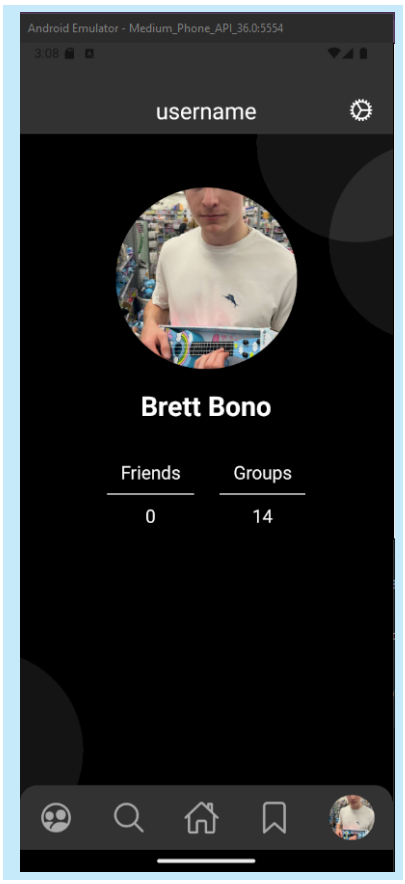


Figure 19: Profile View With Dark-Mode Enabled

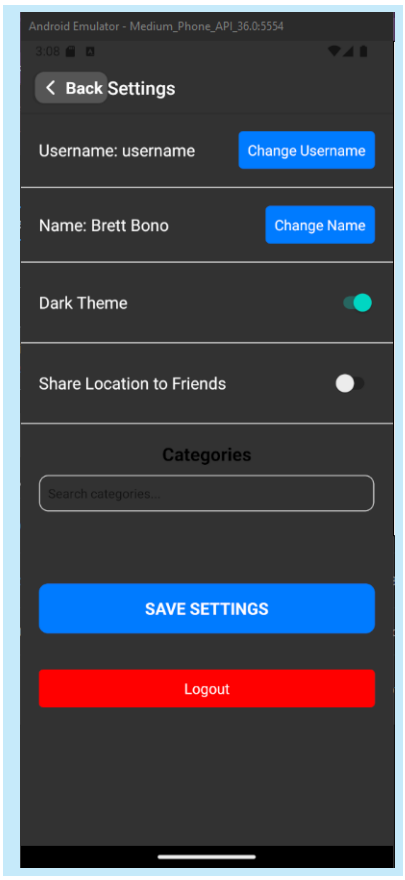


Figure 20: User Settings with DarkTheme Enabled

3.8 Create Event

Off of the default page popup button is the create Event button. This allows for the user to easily create an event on the go. The user must input the event name, description, the host of the event, the privacy of the event, event location, event start and stop, and an image for the event.

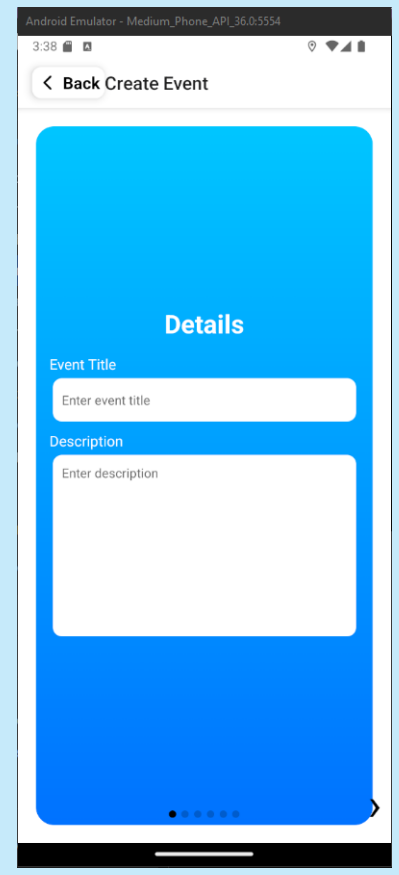


Figure 21: Create Even First Slide, Event Title and Description

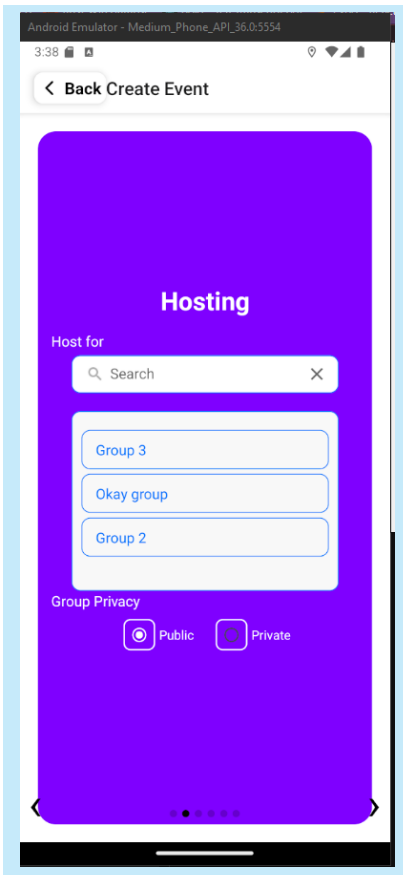


Figure 22: Create Event Second Slide, Hosting Selection for Events and Privacy Options

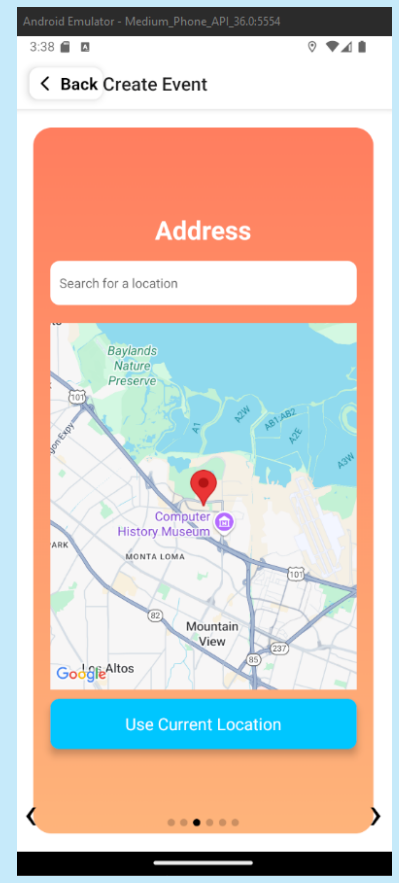


Figure 23: Create Event Third Slide, Set Event Location

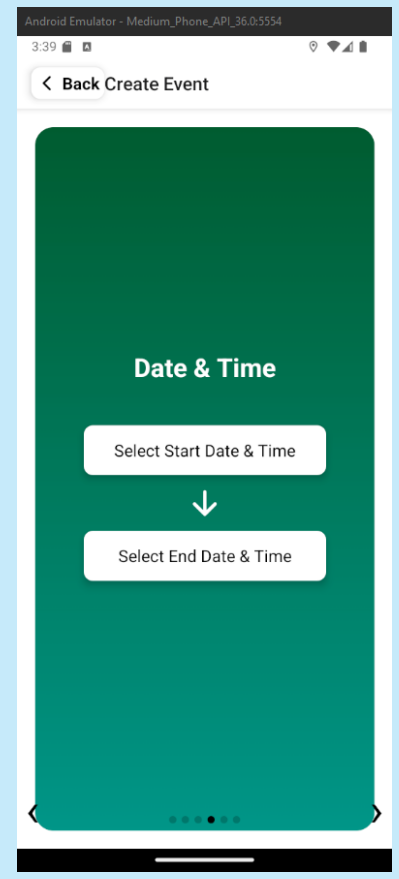


Figure 24: Create Event Fourth Slide, Time and Date of the Event

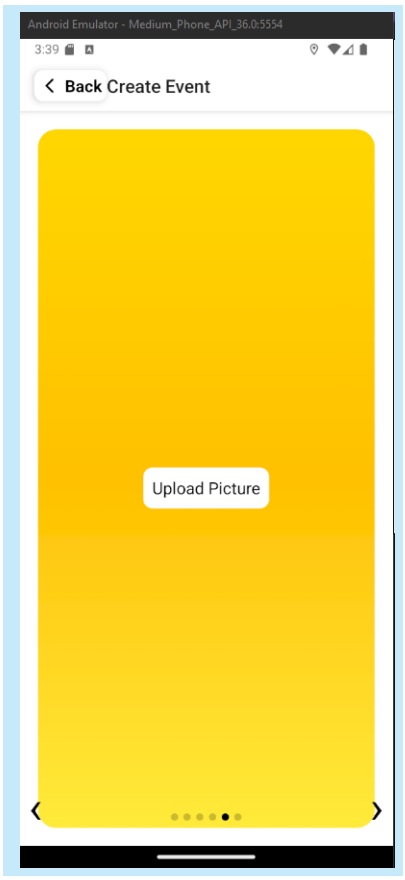


Figure 25: Create Event Fifth Slide, Upload Event Picture

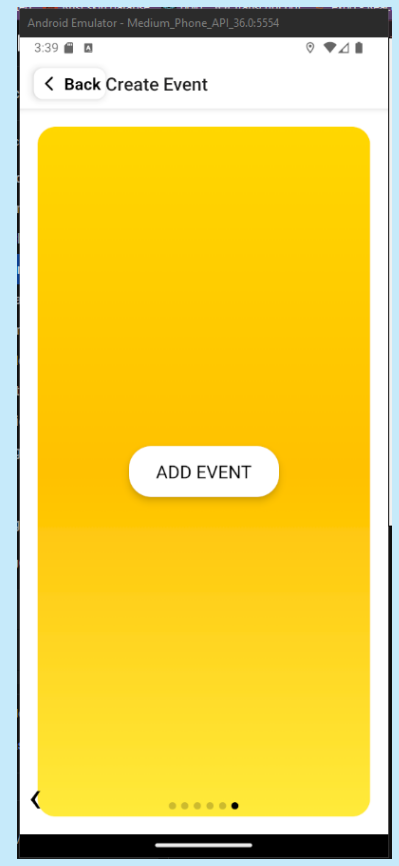


Figure 26: Create Event Sixth Slide, Event Create Confirmation

3.9 Create Group

Off of the default page popup button is the create group button. This allows for the user to easily create a group. The user must input the group name, description, the privacy of the group, and categories for the group.

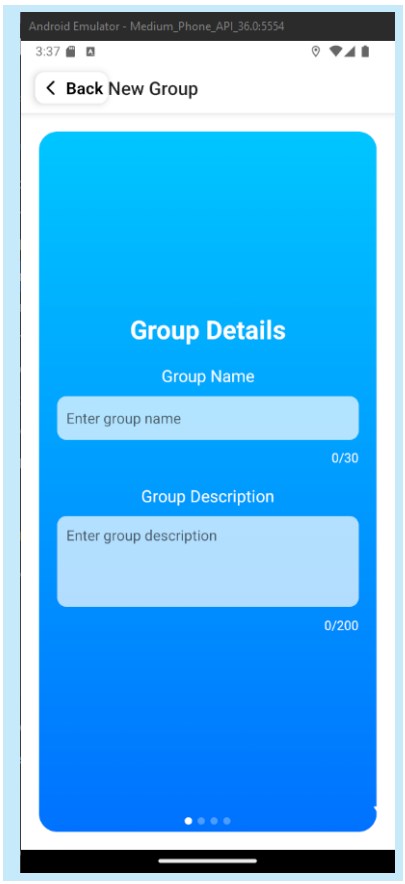


Figure 27: Create Group First Slide, Group Name and Description

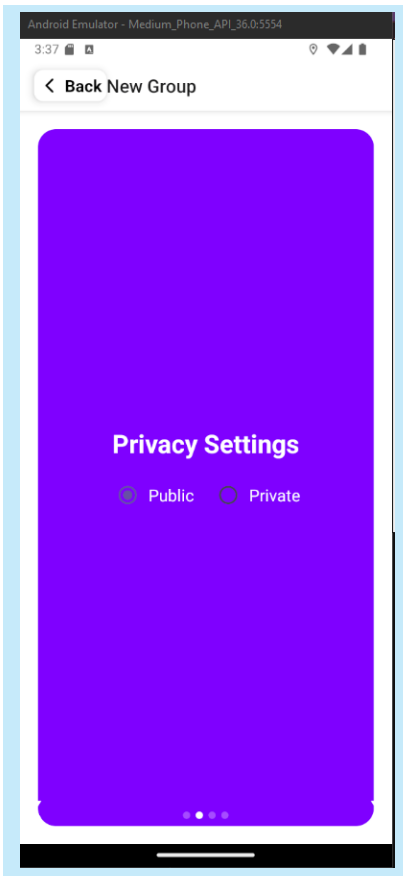


Figure 28: Create Group Second Slide, Group Privacy

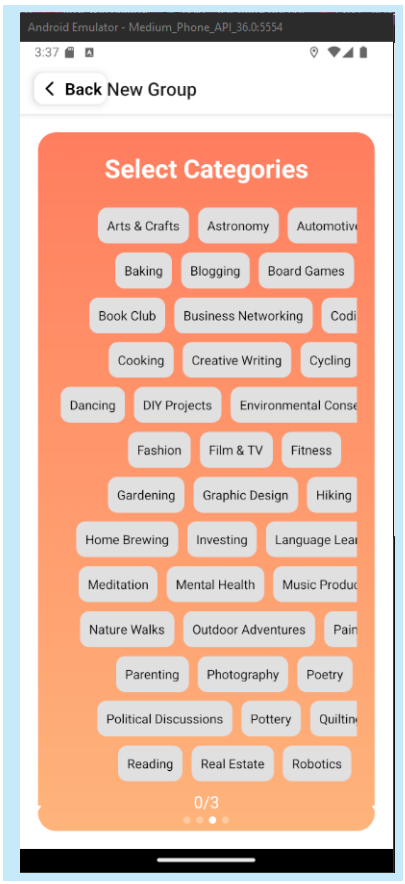


Figure 29: Create Group Third Slide, Group Categories

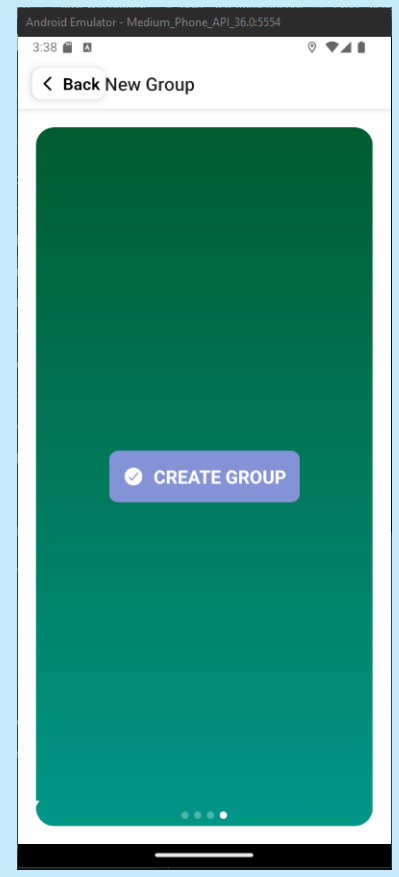


Figure 30: Create Group Fourth Slide, Create Group Confirmation

4 Business Plan

At the same time Michael and I were developing the app, we decided to reach out to James Madison University's Incubator Program, an initiative designed to support student entrepreneurs by offering mentorship, resources, and collaborative opportunities to help transform innovative ideas into viable startups. We had the opportunity to pitch a rough version of our idea and demonstrate our progress. One of the program's lead members gave us valuable advice on the business side and invited us to join her at a University of Virginia incubator event. There, we shared our idea with other startups and received insightful feedback. Overall, the event was fantastic. We learned, among other things, that we needed to write an official business plan.

I took on the task of writing the business plan while Michael focused on refining the UI. To inform the plan, I created a poll and shared it with my colleagues to gather their feedback. Additionally, I used Edgar to research financial data from similar companies, which helped me gain a clearer perspective on the event-management market.

4.1 Analytics

The following are the known "competitors" in the industry with attached information.

Competitors:	App Name	Business Entity	Open Corporates	Number of Branches	Corp Numbers(head, ... branches)	State filed, .. branch states	Private/Public	Edgar
	Poppin	Poppin Technologies inc	https://opencorporate	2	6057377, 5034717	Deleware, California	Private	
	Pull Thru	OctoPlus Technologies LLC	https://opencorporate	1	23009193	Georgia	Private	
	Eventbrite	Eventbrite Inc	https://opencorporate	37	4742147, ...	Delaware	Public	https://www.sec.gov/edgar/browse/?CIK=1475115&owner=exclude
	Posh	PoshGroup Inc			2708409	Delaware	Private	
	LNE	Live Nation Entertainment Inc	https://opencorporate	13	4009151	Delaware	Public	https://www.sec.gov/edgar/browse/?CIK=1335258&owner=exclude

Figure 31

Out of all these companies, only two were public. For these two companies I created a brief Python script to fetch their 10-K filings from Edgar and import the data directly into a Google Sheet for simple analysis.

```python
import pandas as pd
from financetoolkit import Toolkit
import gspread
from gspread_dataframe import set_with_dataframe
from oauth2client.service_account import ServiceAccountCredentials

def main():
    scope = ["https://spreadsheets.google.com/feeds",
             "https://www.googleapis.com/auth/drive"]
    creds = ServiceAccountCredentials.from_json_keyfile_dict({
        "type": "service_account",
        "project_id": "<PROJECT_ID>",
        "private_key_id": "<PRIVATE_KEY_ID>",
        "private_key": "<PRIVATE_KEY>",
        "client_email": "<CLIENT_EMAIL>",
        "client_id": "<CLIENT_ID>",
        "auth_uri": "https://accounts.google.com/o/oauth2/auth",
        "token_uri": "https://oauth2.googleapis.com/token",
        "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
        "client_x509_cert_url": "<CLIENT_CERT_URL>",
        "universe_domain": "googleapis.com"
    }, scope)
    client = gspread.authorize(creds)


    companies = Toolkit(
        tickers=["EB", "LYV"],
        api_key="<YOUR_API_KEY>",
        start_date="2015-01-01"
    )

    spreadsheet = client.open('Financial Analysis')


    income_statement = companies.get_income_statement()
    balance_sheet = companies.get_balance_sheet_statement()
    cash_flow_statement = companies.get_cash_flow_statement()

    # Convert to DataFrames
    df_income = pd.DataFrame(income_statement).reset_index()
    df_balance = pd.DataFrame(balance_sheet).reset_index()
    df_cash_flow = pd.DataFrame(cash_flow_statement).reset_index()

    # Create sheets
    sheet_income = spreadsheet.add_worksheet(title="Income Statements", rows="100", cols="20")
    sheet_balance = spreadsheet.add_worksheet(title="Balance Sheet", rows="100", cols="20")
    sheet_cash_flow = spreadsheet.add_worksheet(title="Cash Flow Statement", rows="100", cols="20")

    # Upload data
    set_with_dataframe(sheet_income, df_income)
    set_with_dataframe(sheet_balance, df_balance)
    set_with_dataframe(sheet_cash_flow, df_cash_flow)

    print("Data uploaded to Google Sheets.")
```

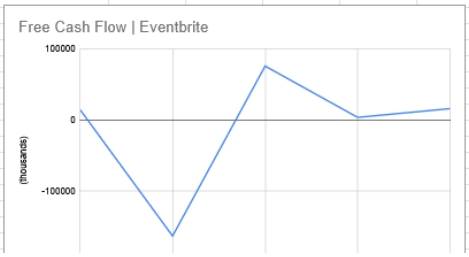
```
55
56 if __name__ == "__main__":
57     main()
```

Listing 1: Script to fetch 10-K data from EDGAR and export to Google Sheets

Here is the excel data.

level_0	level_1	2019	2020	2021	2022	2023							
EB	Net Income	-68760000	-224718000	-139080000	-55384000	-26479000	LYV	Net Income	118212000	-1827790000	-608786000	409193000	734317000
EB	Depreciation and Amortization	24324000	22610000	18716000	14860000	14740000	LYV	Depreciation and Amortization	443991000	485025000	416277000	449976000	516797000
EB	Deferred Income Tax	-380000	-183000	59237000	0	9951000	LYV	Deferred Income Tax	-465000	-37877000	-9639000	7199000	-44018000
EB	Stock Based Compensation	37594000	40215000	47523000	53356000	55056000	LYV	Stock Based Compensation	48785000	116889000	209337000	110049000	115959000
EB	Change in Working Capital	6531000	-123842000	87343000	-22799000	-29023000	LYV	Change in Working Capital	-287907000	79457000	1773993000	637853000	-159985000
EB	Accounts Receivables	-288000	-2505000	-591000	-2221000	-1352000	LYV	Accounts Receivables	-159792000	490588000	-485211000	-463977000	-550670000
EB	Inventory	-9735000	30900000	4544000	0	0	LYV	Inventory	-170486000	141631000	95533000	0	-202834000
EB	Accounts Payables	36840000	-116566000	93246000	31301000	-7777000	LYV	Accounts Payables	-45920000	-1379461000	1315722000	1002158000	460496000
EB	Other Working Capital	-20286000	-35671000	-9856000	-51879000	-19894000	LYV	Other Working Capital	88291000	826699000	847949000	99672000	133023000
EB	Other Non Cash Items	29349000	129026000	5342000	18577000	-535000	LYV	Other Non Cash Items	147167000	101658000	-614000	217793000	207724000
EB	Cash Flow from Operations	28858000	-156892000	79081000	8610000	23710000	LYV	Cash Flow from Operations	469783000	-1082638000	1780568000	1832063000	1370794000
EB	Property, Plant and Equipment	-13598000	-6282000	-2533000	-4451000	-7170000	LYV	Property, Plant and Equipment	-365827000	-222609000	-159834000	-353286000	-475257000
EB	Acquisitions	0	-6375000	0	-1125000	6073000	LYV	Acquisitions	-292351000	-33322000	-404408000	-344514000	-70932000
EB	Purchases of Investments	0	0	0	-83926000	-370160000	LYV	Purchases of Investments	-57280000	-19003000	-6854000	-91186000	-37198000
EB	Sales of Investments	0	0	0	3026000	308000000	LYV	Sales of Investments	57280000	19003000	6854000	5220000	545000
EB	Other Investing Activities	-7710000	-4583000	-1548000	-3026000	-6073000	LYV	Other Investing Activities	-32822000	31889000	-2720000	-925000	-112963000
EB	Cash Flow from Investing	-13598000	-12657000	-2533000	-89502000	-69330000	LYV	Cash Flow from Investing	-691000000	-224062000	-566962000	-784691000	-695805000
EB	Debt Repayment	-73594000	-517000	-143247000	0	0	LYV	Debt Repayment	-437267000	-30987000	-109705000	-45792000	-330383000
EB	Common Stock Issued	44300000	20574000	1429000	3146000	2434000	LYV	Common Stock Issued	14104000	30647000	449630000	35775000	19264000
EB	Common Stock Purchased	-2363000	-5517000	-13705000	-6591000	-7342000	LYV	Common Stock Purchased	-15353000	-47539000	-45845000	-76925000	-9484000
EB	Dividends Paid	0	0	0	0	0	LYV	Dividends Paid	0	0	0	0	0
EB	Other Financing Activities	43140000	255556000	220409000	1366000	-7342000	LYV	Other Financing Activities	768654000	1414103000	923097000	20527000	-342762000
EB	Cash Flow from Financing	-32817000	255039000	51181000	-2079000	-4908000	LYV	Cash Flow from Financing	328889000	1349332000	1171332000	-143340000	6885000
EB	Forex Changes on Cash	0	1065000	-6753000	-13014000	4246000	LYV	Forex Changes on Cash	-11633000	29565000	-43585000	-179450000	38874000
EB	Net Change in Cash	-16460000	85490000	127729000	-95985000	-44851000	LYV	Net Change in Cash	96039000	72197000	2341353000	724582000	626582000
EB	Cash End of Period	422940000	508430000	636159000	540174000	537973000	LYV	Cash End of Period	2474242000	2546439000	4887792000	5612374000	6238956000
EB	Cash Beginning of Period	439400000	422940000	508430000	636159000	582824000	LYV	Cash Beginning of Period	2378203000	2474242000	2546439000	4887792000	5612374000
EB	Operating Cash Flow	28858000	-156892000	79081000	8610000	23710000	LYV	Operating Cash Flow	469783000	-1082638000	1780568000	1832063000	1370794000
EB	Capital Expenditure	-13598000	-6282000	-2533000	-4451000	-7170000	LYV	Capital Expenditure	-365827000	-222609000	-159834000	-353286000	-475257000
EB	Free Cash Flow	15060000	-163174000	76548000	4159000	16540000	LYV	Free Cash Flow	103956000	-1305247000	1620734000	1478777000	895537000

	Eventbrite	thousands	Live Nation	thousands
2019	15060000	15060	103956000	103956
2020	-163174000	-163174	-1305247000	-1305247
2021	76548000	76548	1620734000	1620734
2022	4159000	4159	1478777000	1478777
2023	16540000	16540	895537000	895537



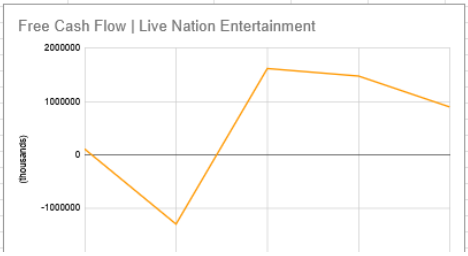


Figure 32: Publicly Visible here https://docs.google.com/...

4.2 Business Plan Rough Draft

Below is the most recent version of the business plan prior to the project being put on hold

INSERT IMAGE HERE

Business Plan

Business Name: GetOut

Brief Summary: GetOut is a mobile ios application that will be a subsidiary of White Star Studios inc. GetOut will provide an array of services that are all centralized in one application. These services allow for the discovery of public and/or personalized events for individuals, and, for businesses, these services promote competition in the realm of event management on a local and broad scale.

All services have their own category and structure to see a list of these services visit ***Exhibit 1***.

Team Members: Email

Brett Bono _____bonobh@dukes.jmu.edu

Michael Gerber _____**********jmu.edu

Executive Summary

WhiteStarStudios Inc.

Contact: Brett Bono

800 S Main St, Harrisonburg, VA 22801

Phone: (757)920-3042

E-mail: bonobh@dukes.jmu.edu

Management Titles: CEO,CFO, CTO

Number of Employees: 2 unpaid

Business Description: GetOut is a mobile ios application dedicated to enhancing social engagement and promoting event discovery through innovative technology. Our mission is to connect users with enriching experiences and foster a vibrant community of event-goers and creators. We aim to make it easy for users to find and enjoy a wide range of social events while helping event organizers promote their gathering and reach a broader audience.

Competitive Advantage: While there are many event platforms out there, GetOut will utilize its very easy to use user interface and will provide an array of services that other platforms do not have. From account creation users, off the bat will be able to find public events in the area. Users will be recommended groups to join based on their interests and preferences. Event creators, whether it be an official business, or not, will be able to promote their events to a broad audience with a click of a button. These creators will also be able to receive statistics on their previous events so that they can adjust and improve future events.

Markets: GetOut targets a broad market that includes but not limited to young adults and business owners. . GetOut is scalable so that events are not just parties, or 21+ events. GetOut can be used for scheduling business meetings, school orientation programs, scheduling of club events, and more.

Competition: Poppin Technologies Inc , Pull Thru by OctoPlus technologies, Eventbrite Inc, Posh Group Inc, Live Nation Entertainment Inc.

Market Analysis

Objectives: The objective of our market analysis is to compare companies within the same industry, big and small, to obtain an estimate of industry growth.

Industry Overview:
The event marketing industry is a relatively small space. Many of the popular companies in this industry follow the same path of venue and ticket marketing. Some major players are Eventbrite, and Live Nation Entertainment.

Recently, there has been an upsurge of small startups within the space. Small companies such as Poppin Technologies Inc, OctoPlus technologies, and Posh Group Inc are attempting to innovate the space by not only allowing ticketing and event creation for large venues and businesses but they are allowing for the scalability of events. On these platforms the user has the freedom to create whatever type of event they want, big or small.

While many of these startups are innovative, WhiteStarStudios Inc believes that many of these startups have flaws that will not help the growth of their companies in this upcoming competitive space.

Below are a list of recent startups:

Poppin Technologies Inc (Poppin)

OctoPlus Technologies Inc (Pull Thru)**:**

Posh Group Inc (Posh):

Financials:

Competition Financials:

GetOut Projected Financials:

Survey Data:

WhiteStarStudios plans to conduct multiple surveys (see **Marketing Plan, Phase 1**).

Current survey data gathered:

Individual Surveys:

General Consensus Survey:

- On 18 July 2024, WhiteStarStudios released the first iteration of the general consensus survey, reaching 43 individuals. All of the survey data can be found in ***Exhibit 6***

  In this survey the following data was obtained.

  1.) Age
  2.) College enrollment
  3.) Enrolled college year
  4.) Major
  5.) Frequency of attending of campus events
  6.) Frequency of attending off campus events
  7.) Frequency of hosting events
  8.) Greek Life Info
  9.) Difficulty of finding on campus events
  10.) Difficulty of finding off campus events
  11.) Description of issues for 9 or 8
  12.) Source of event information
  13.) Platform used to advertise created event
  14.) Would they use GetOut
  15.) Would they want to use GetOut demo upon availability
  16.) Name and email

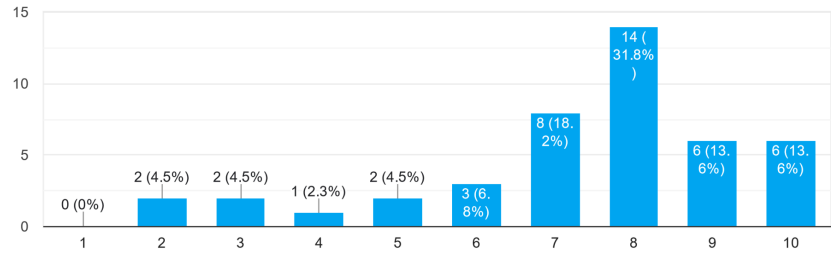
- The primary data of interest are 5,6,7,9,10,11, and 14.

Below is the data

5.)

How often do you go to off campus social events? (Bars, Breweries, Parties, Concerts, etc)

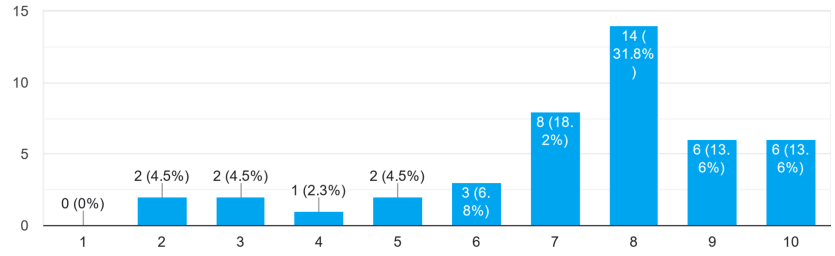
44 responses



6.)

How often do you go to off campus social events? (Bars, Breweries, Parties, Concerts, etc)

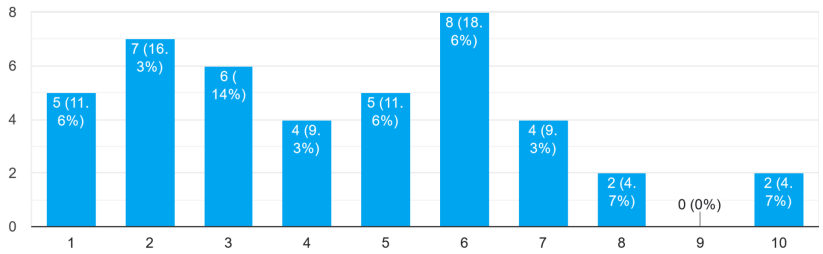
44 responses



7.)

How often do you host events (minimum 5 people)?

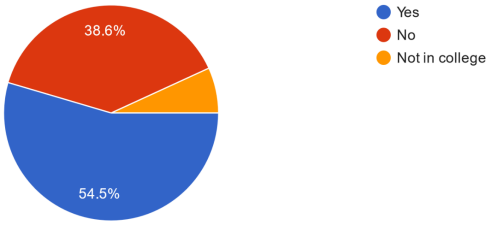
43 responses



9.)

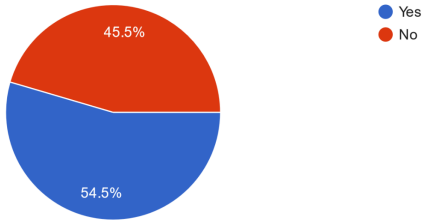
Have you ever had issues finding events on campus?

44 responses



10.)

Have you ever had issues finding events off campus?
44 responses



11.)

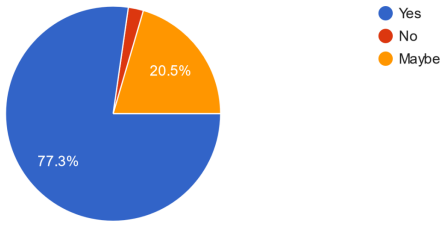
Question: *Could you describe any issues if you said yes to ever having issues finding events on or off campus*. The survey received 21 responses to this.

The majority of the survey respondents mentioned issues with the following:

1.*) Difficulty finding events of particular interest*
2.) *Difficulty finding out what events friends are a"*
3.) *Events are not well advertised*
4) *Transparency of public vs private events*
5.) *Individual commitment and lack of event details*

14.)

If there was an app that was focused on sharing nearby social events and showed you events your friends are at, would you use it?
44 responses



Application Design:

Application Backend Services:

Current Backend Services: Google Firebase Services:

1.) Firestore Database
2.) Authentication
3.) Storage
4.) Cloud Functions

***Depending on user growth, WhiteStar Inc plans to migrate off these services. WhiteStar plans to create its own api service using the gRPC protocol to connect to a more cost efficient database.**

Marketing Campaign:

Introduction GetOut will be heavily user dependent. The users will define the ecosystem within the app. To ensure GetOut has a promising future, the initial marketing campaign will be one of the most important operations. Our main priority is that individuals and businesses will be the suppliers of events and that individuals will always have a constant stream of opportunities at their fingertips.

Marketing Plan:

Phase 1:

Data Collection:

- Individual Surveys*:*

- WhiteStar Studios has plans to conduct a list of surveys with the general public. The list of surveys are:

  1.) General Consensus Survey:

  This survey gauges whether or not the general public would find interest in our application.

  <u>*This survey has already been conducted. WhiteStarStudios reached out to 43 individuals and plans to reach out to more (Market Analysis, Survey Data)</u>

  2.) Individual Feature Survey:

  This survey asks the general public what features they would use and not use as well as features that they would like.

  *Visit ***Exhibit 4*** for survey specifics.

- Business Surveys:

  - WhiteStar Studios has plans to conduct a list of surveys with businesses. The list of surveys are:

    1.) General Consensus Survey:

    This survey gauges whether or not the businesses would find interest in our application.

    2.) Business Feature Survey:

    This survey asks businesses what features they would use and not use as well as features that they would like.

Phase 2:

Application Promotion:

- Business Promotion:

  WhiteStar Studios plans to pay local businesses for a period of time to promote their business events on GetOut. We plan for these businesses to use Getout to advertise their events before we onboard standard users so that when the first batch of users do register there is already a pool of events that users can view.

  Having events established before a first batch of users join the app is essential to user growth. Eventually, once there are enough users to keep the app sustainable, businesses will want to keep promoting their events on the app so that their events are advertised to a vast number of users.

- Individual Event Promotion:
  WhiteStarStudios plans to pay small organizations who have a relatively large audience and are deemable as a marketing agent, such as fraternities, sororities, and clubs, to promote their events using GetOut.

- Experiential Marketing
  WhiteStarStudios plans to organize local and interactive events in communities that we deem is marketable. Please visit ***Exhibit 5*** for a list of interactive events.

Expectations:

Events:

- Title
- Description
- Address (Street Address, City, State, Country, Zip)
- Start Date
- Start Time

- End Date
- End Time
- Group
- CoGroups
- Publicity (Public or Private)
- Image (custom or group photo)

Overview Of Exhibits:

Definitions:

User: An individual
Group: A collection of users. A group can be an official business or a gathering.
Event: Point of interest defined by a user or group.
Relationship Status: Attributes defined between users, groups, and events.

Exhibit 1:

Individual Services:

A.)User Defined Map:

- A centralized map that exists on the homepage upon registration. This is a conditional based, user focused map, that is overlapped by points of interest. By default, each user is able to view certain points of interest, such as public events that businesses and individuals create, and the location of user added friends that are at events, within a variable radius.
  Depending on the relationship status between the user and their groups, users have the ability to also view private events that are associated with the groups the user is in. As an example: User 1 is in Group A and Group B. User 2 is in Group A and Group C. Group A, B, and C are hosting private events. User 1 and 2 will both have Group A's event rendered on the map. However, Group B's event will be only visible by User 1 and Group C's event only visible by User 2.

B.) Event Filtering:

- Users are able filter out events based on certain parameters. Users will be able filter events that are upcoming, live, part of certain groups, and that are public / private.

C.) Relationship/Junction Services:

- Individuals:
    - Each user has the functionality to create a relationship status with another user. This relationship status can be seen as adding or blocking another user. Depending on the relationship status between the users determines the relationship functionality. The most common relationship status for two users is the friend status. The friend status between users allows for direct messaging, sharing of locations, and more.

- Groups:
    - Each user has the ability to create a relationship status with a group. This ability varies depending on relationship permissions. A user relationship status between themselves and a group can be viewed as joining and blocking a group. When a user creates a group the relationship between the user and the group created will default to the user joining the group with themselves having owner set permissions. Each relationship status has its own functionality. If a user joins a group, they by default, will have member permissions. The member permissions will be the most common. These permissions allow the user access to functionalities such as reading and writing of certain group attributes. An example of this would be a member communicating to the group via messaging, a member viewing group events, whether public or private, and so forth.

- Events:
    - Each user has the ability to create events depending on the user type. The user can declare an event to be written under a group that the user has permissions for, or as an individual event. All individual events are set to be public, but if the event is written under a group, the user can declare the event public or private. Declaring event attributes such as titling the event, setting an event description, and setting an event location are subject to certain rules (***Exhibit 2***). Upon event creation, a relationship status between the user who created the event (if individual), or groups associated with hosting the event will be created. The existence of a relationship status allows certain functionality, such as editing event attributes. An example of these attributes are event title, location, whitelist, blacklist, etc.

Group Services:

Relationship/Junction Services:

- Groups(Co Hosting)
    - Each group has the ability to create a relationship status with other groups, similar to how users can add other users. All relationship statuses are temporary. Depending on the relationship status, functionalities allow for joint group chats and sharing of events.

- Events
    - Each group has the ability to create a relationship status with events. Once an event is created by a user, the user can define the event to be a group event with the correct permissions. Once a group event is defined a relationship status is created between that event and the group or groups (if co-hosted). The functionalities associated with the relationship status are event chats and so forth.

Business Services:

- Data Gathering:
    - When events are created, certain event data will be used as input for statistical models to improve a future event. Data will be anonymous. Data will include demographics, a population density measurement, user movement within an event radius, and more. These MicroServices are based on the data collected and will be sold to businesses for improvement of event management solely to improve competition within the market. See ***Exhibit 3*** for a list of microservices.

- Promotional Events:
    - Certain events will have the ability to be promoted within a certain radius. Businesses or Individuals can use the promotional system to advertise their events so that they can be seen by more users.

***Exhibit 2*:**

Rules

***Exhibit 3*:**

MicroServices:

Customizable Map:

Ticketing:

Exhibit 4:

Survey Specifics:

Exhibit 5:

Interactive Events:

- Official GetOut events
- Sale of Physical Stock

Exhibit 6:

Survey Data:

A Project Struggles and Conclusion:

Unfortunately, the project was never fully completed. Michael and I officially began development in June 2024 while he was juggling a summer internship, and I was balancing part-time work with summer classes. Once the fall semester at JMU began, both of us became overwhelmed with academic responsibilities. Michael had to shift focus to his computer science coursework, and I had to do the same with my mathematics classes.

Despite having no prior experience with React Native or many of the libraries we used, we gained a tremendous amount of knowledge throughout the process. We encountered several challenges, but managed to work through most of them. While the project is currently on pause, there's potential for it to be revived and possibly even marketed in the future.

Below are personal notes outlining what I would do differently if the project were to be restarted.

A.1 Obstacles

Database Originally, I explored using a different database with a gRPC API layer to handle communication between services. While I enjoyed learning GoLand and gained valuable experience with it, I quickly realized that setting up this architecture would significantly delay development. Given our time constraints, we ultimately opted for Firebase due to its ease of use and quick integration, which allowed us to maintain momentum on the project.

In retrospect, a different database may have served us better. Firebase's NoSQL structure posed challenges when modeling relational data. To handle many-to-many relationships, I implemented junction tables with encrypted field names. While this worked, it made the codebase more difficult to read, debug, and maintain over time.

Unisearch One of the most technically demanding aspects was designing the universal search feature. I envisioned a system similar to Instagram's—capable of surfacing mutual friends, groups, and events with relevance-based priority and fast response times. I even considered implementing a variation of Google's PageRank algorithm, but quickly realized the complexity involved and found it difficult to determine a starting point. As a result, the search algorithm remained incomplete when the project was paused.

Next time, if I tackle a similar challenge, I would take a more research-driven approach. I would start by implementing a basic, working search using indexed queries and simple filters, then gradually layer on ranking based on relevance. I would spend more time upfront researching information retrieval techniques, such as vector similarity.

Business-Side With barely any prior knowledge of how the business side of mobile applications works, I had to rely heavily on information from people and online resources. Balancing this alongside the software development proved to be quite overwhelming. Next time, I believe it would be more effective to first prioritize building a fully functional prototype of the app. Once that foundation is established, I could then shift focus to researching the business aspects and developing a solid business plan. Alternatively, recruiting a team member with business expertise to handle those responsibilities would allow me to concentrate on the technical development while ensuring the business side is properly managed.