GNG2101

Introduction to Product Development
and Management for Engineers
and Computer Scientists
Mana Azarm

PM: Ayesha Khan
TA: Zaineb Al-Faesly

Project Deliverable C: Conceptual Design and Project Plan

Group B-14: Personal Safety

Submitted by:

Sabeeha Ahamed 300167378
Sabina Solomon 300174303
Kian Ashrafganjouei 300173780
Iman Jama 300172218
Darby Martin 300198642

Sep 30, 2021

University of Ottawa

Abstract

The purpose of this report is to analyze the different concepts and designs that the team members create in order to find the best possible solution to our problem statement (stated in the previous deliverable). The goal is to develop a working prototype that meets all the target specifications and satisfies the client.

Table of Contents

List of Figures

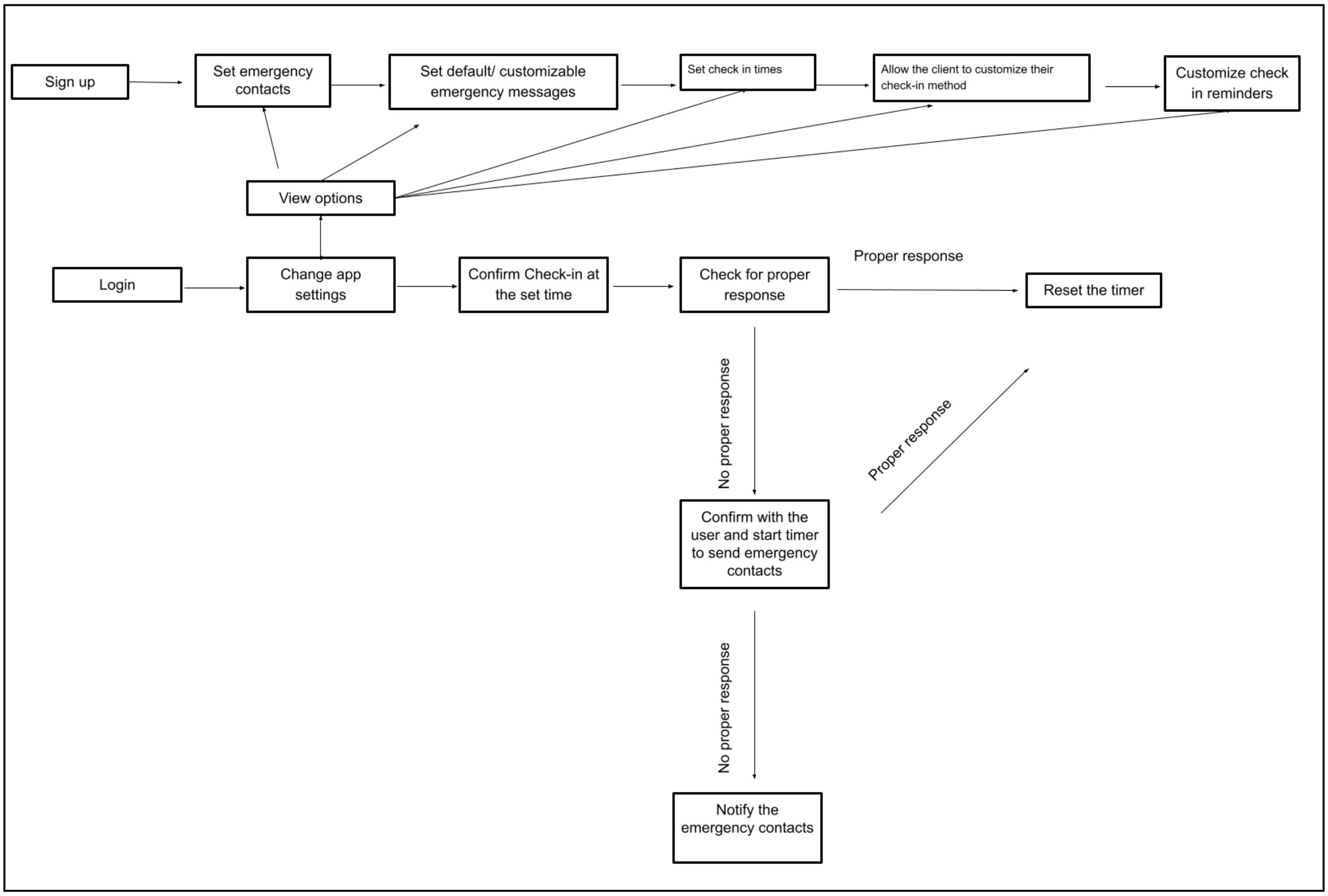
List of Tables

Introduction

The most important part about building a product is planning. We want to be able to understand their functionality and usage. This requires generating several ideas that will be used to create one generalized concept for the prototype. Each member used the functional decomposition of our project to create 3 concepts that they would like to implement. However, most of the concepts created were similar. Analyzing each component helped us outline the process we need to reach the target specification set in the previous deliverable.

Functional Decomposition

Figure 0: Functional Decomposition


Sign up
Set emergency contacts
Set default/ customizable emergency messages
Set check in times
Allow the client to customize their check-in method
Customize check in reminders
View options
Login
Change app settings
Confirm Check-in at the set time
Check for proper response
Proper response
Reset the timer
No proper response
Proper response
Confirm with the user and start timer to send emergency contacts
No proper response
Notify the emergency contacts

This is an overview of the functionality of the application we would like to implement.

Product Concepts

Each member of the group created ideas of what they would like to develop in the personal safety app. After looking through the generated concepts, we created a pros and cons list to help us come to a conclusion on the details of the app’s functionality.

Figure 1: Case: User Attempts to Check In - Sabina

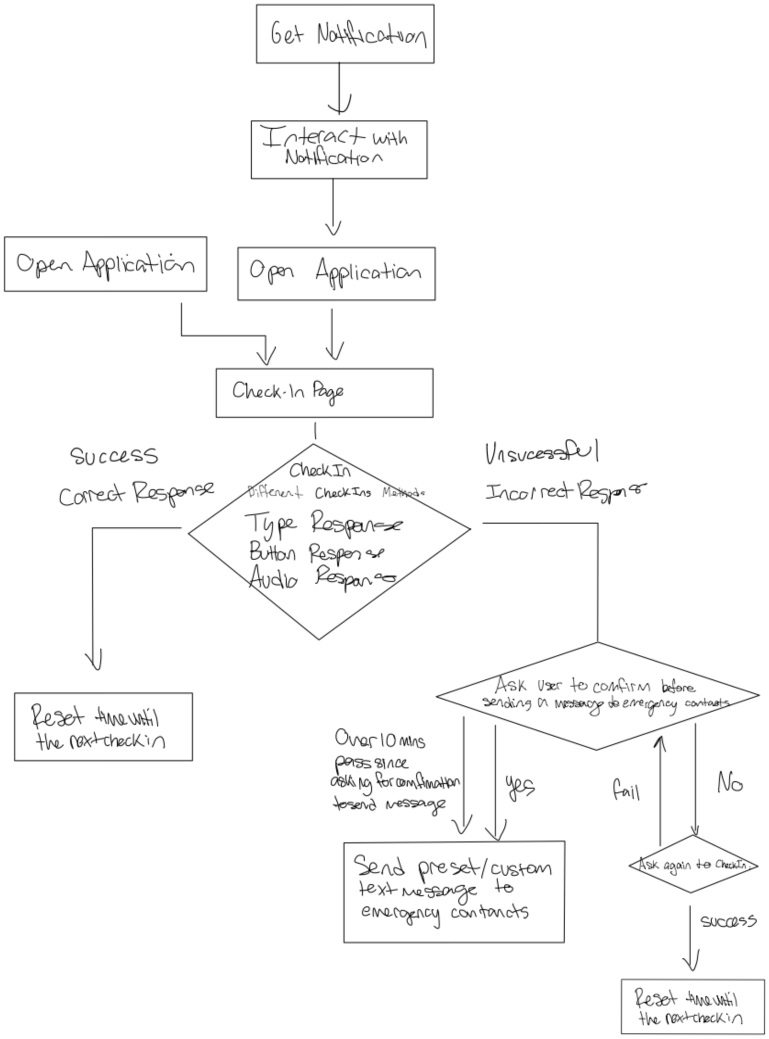

Get Notification
Interact with Notification
Open Application
Open Application
Check-In Page
Success
Correct Response
Unsucessful
Incorrect Response
Check In
Different CheckIns Methods
Type Response
Button Response
Audio Response
Reset time until the next checkin
Ask user to confirm before sending a message to emergency contacts
Over 10 mins pass since asking for confirmation to send message
yes
fail
No
Send preset/custom text message to emergency contancts
Ask again to CheckIn
success
Reset time until the next checkin

Figure 2: Case: Missed Check-In - Sabina

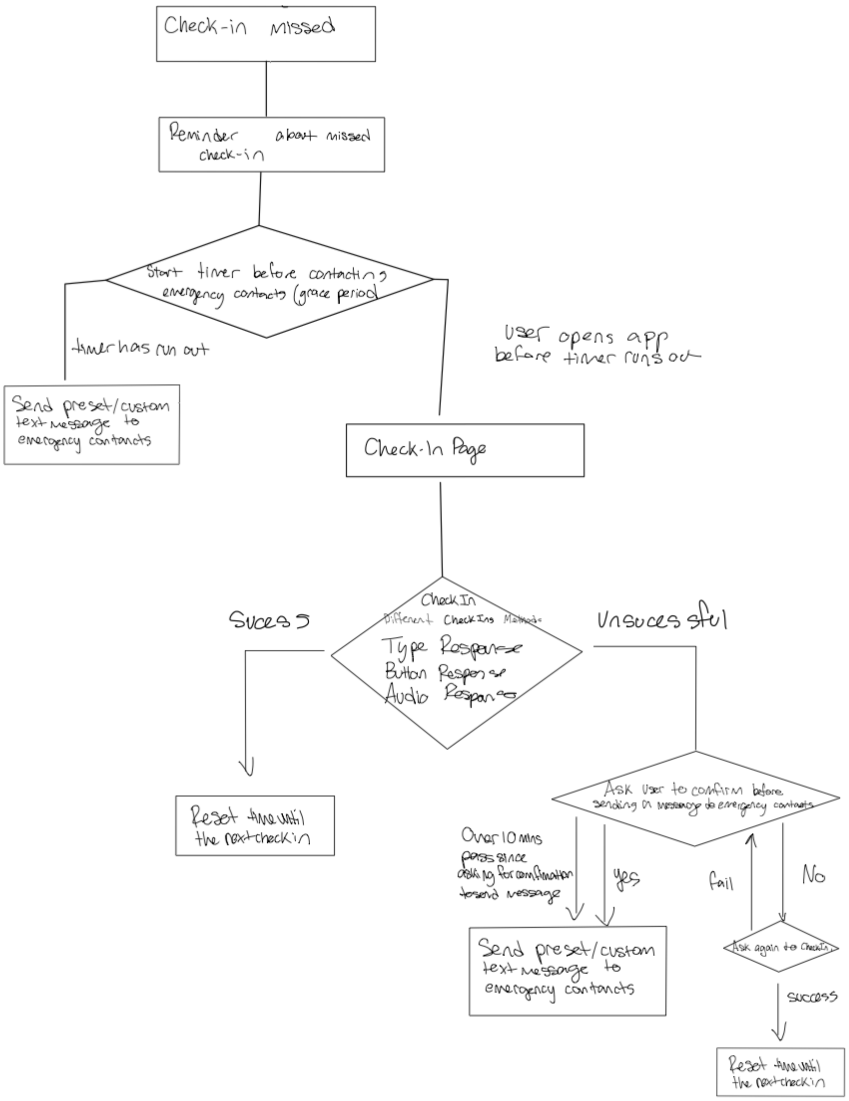

Check-in missed
Reminder about missed check-in
Start timer before contacting emergency contacts (grace period)
timer has run out
Send preset/custom text message to emergency contancts
User opens app before timer runs out
Check-In Page
Sucess
Check In
Different Check Ins Methods
Type Response
Button Response
Audio Response
Unsucessful
Reset time until the next check in
Ask user to confirm before sending a message to emergency contacts
Over 10 mins pass since asking for confimation to send message
yes
fail
No
Send preset/custom text message to emergency contancts
Ask again to CheckIn
success
Reset time until the next checkin

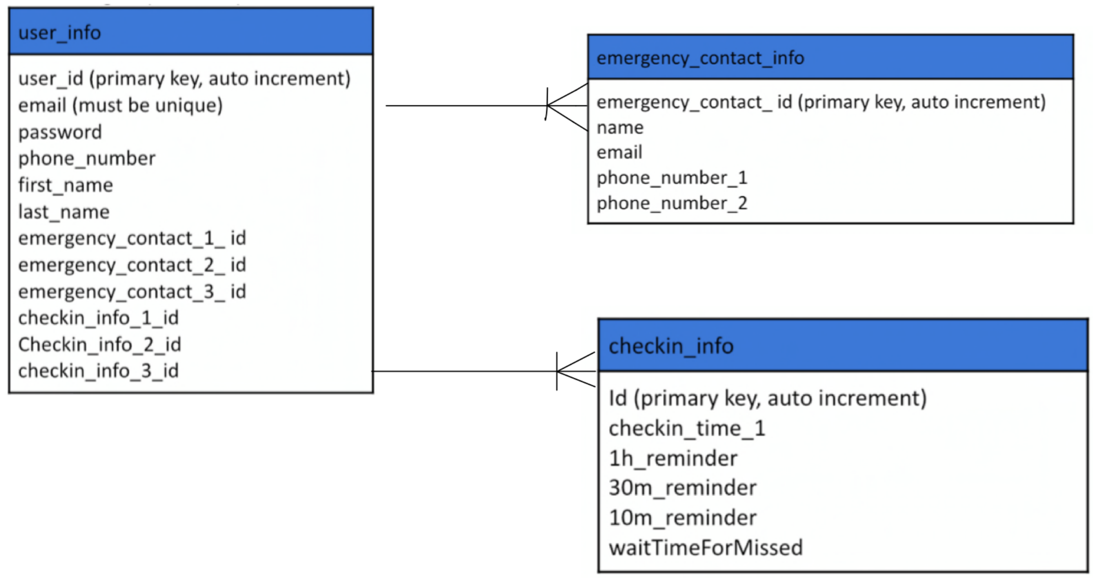
Figure 3: Database Design 1 (non relational)- Sabina

user_info
Id (primary key, auto increment) email (must be unique) password phone_number first_name last_name

emergency_contact_1_ name
emergency_contact_1_ phone
emergency_contact_1_ email
emergency_contact_2_ name
emergency_contact_2_ phone
emergency_contact_2_ email
emergency_contact_3_ name
emergency_contact_3_ phone
emergency_contact_3_ email
checkin_time_1
checkin_time_2
checkin_time_3

Add any additional settings we can think of

Figure 4: Database Design 2 (relational) - Sabina



user_info

user_id (primary key, auto increment)
email (must be unique)
password
phone_number
first_name
last_name
emergency_contact_1_ id
emergency_contact_2_ id
emergency_contact_3_ id
checkin_info_1_id
Checkin_info_2_id
checkin_info_3_id

emergency_contact_info

emergency_contact_ id (primary key, auto increment)
name
email
phone_number_1
phone_number_2

checkin_info

Id (primary key, auto increment)
checkin_time_1
1h_reminder
30m_reminder
10m_reminder
waitTimeForMissed

Figure 5: Sabeeha - How the app checks for appropriate message

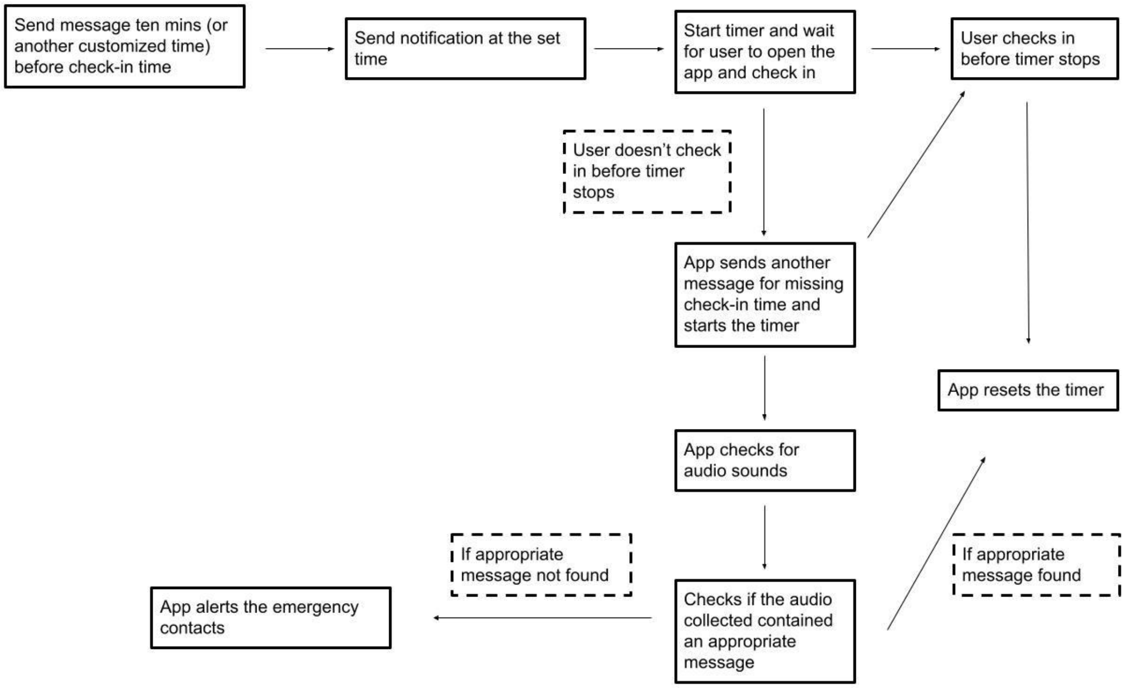

Send message ten mins (or another customized time) before check-in time
Send notification at the set time
Start timer and wait for user to open the app and check in
User checks in before timer stops
User doesn't check in before timer stops
App sends another message for missing check-in time and starts the timer
App resets the timer
App checks for audio sounds
If appropriate message not found
If appropriate message found
App alerts the emergency contacts
Checks if the audio collected contained an appropriate message

Figure 6: Sabeeha - Customizable check in options

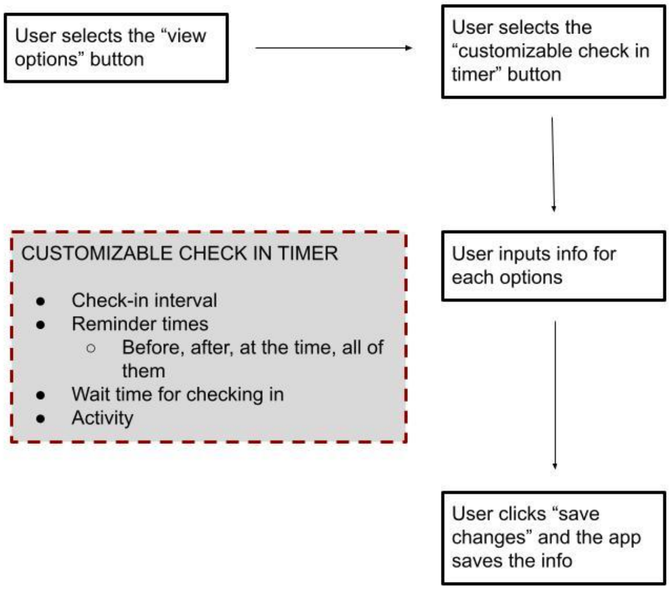

User selects the "view options" button
User selects the "customizable check in timer" button
CUSTOMIZABLE CHECK IN TIMER
Check-in interval
Reminder times
Before, after, at the time, all of them
Wait time for checking in
Activity
User inputs info for each options
User clicks "save changes" and the app saves the info

Figure 7: Sabeeha - Set up emergency contacts

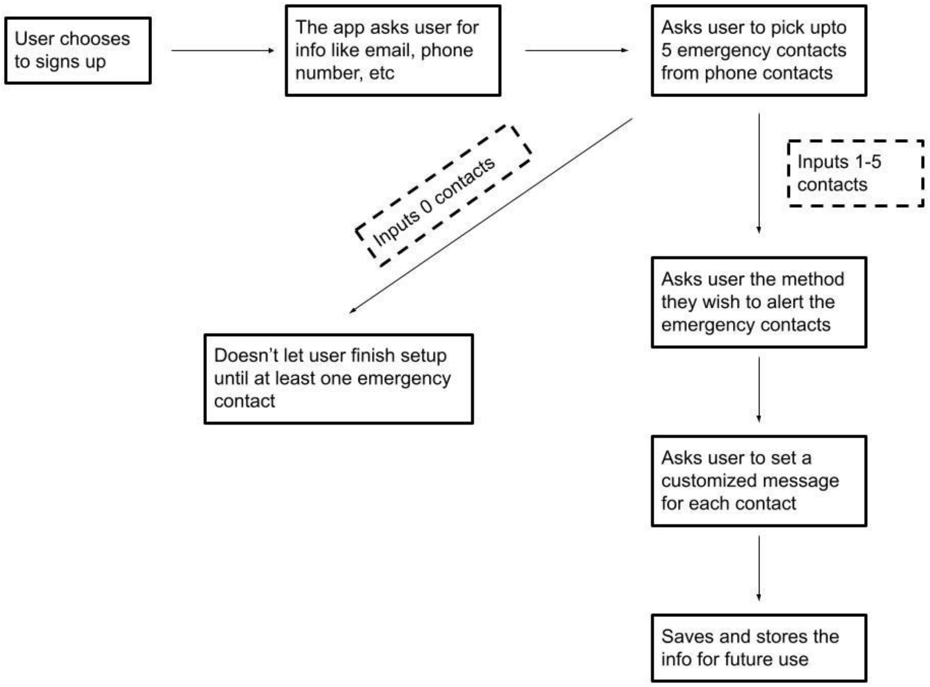
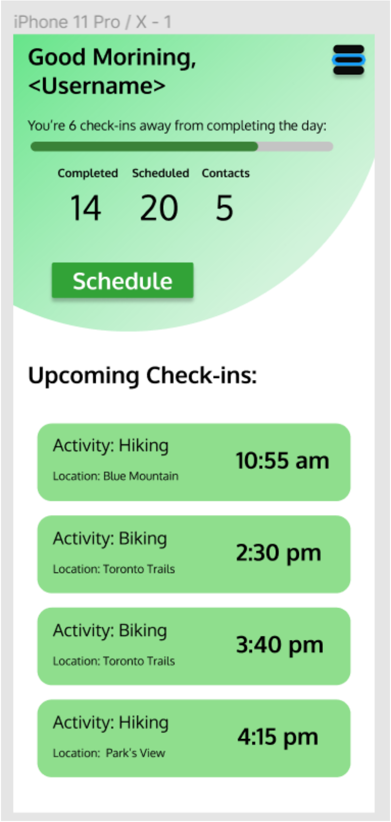

User chooses to signs up
The app asks user for info like email, phone number, etc
Asks user to pick upto 5 emergency contacts from phone contacts
Inputs 0 contacts
Inputs 1-5 contacts
Doesn't let user finish setup until at least one emergency contact
Asks user the method they wish to alert the emergency contacts
Asks user to set a customized message for each contact
Saves and stores the info for future use

Figure 8: Homepage design - Kian Ashrafganjouei


iPhone 11 Pro / X - 1

Good Morining, <Username>

You're 6 check-ins away from completing the day:

Completed	Scheduled	Contacts
14	20	5

Schedule

Upcoming Check-ins:

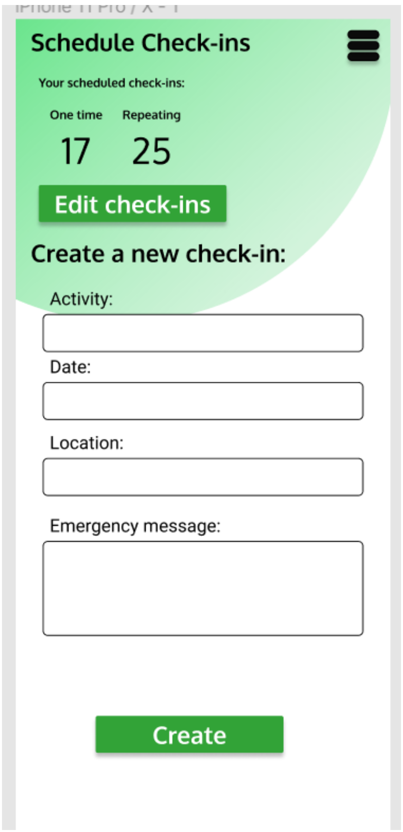
Activity: Hiking 10:55 am
Location: Blue Mountain

Activity: Biking 2:30 pm
Location: Toronto Trails

Activity: Biking 3:40 pm
Location: Toronto Trails

Activity: Hiking 4:15 pm
Location: Park's View

Figure 9: Check-in schedule page - Kian Ashrafganjouei



Schedule Check-ins

Your scheduled check-ins:

One time	Repeating
17	25

Create a new check-in:

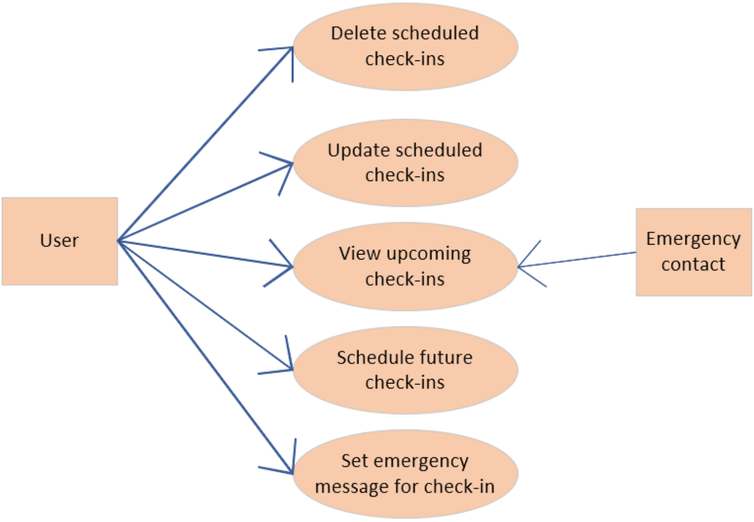
Activity:

Date:

Location:

Emergency message:

Figure 10 : Scheduling check-ins use cases - Kian Ashrafganjouei



User
Delete scheduled check-ins
Update scheduled check-ins
View upcoming check-ins
Schedule future check-ins
Set emergency message for check-in
Emergency contact

Figure 11: App Overview for Signing Up (1. Home Page, 2. Sign up, 3. Emergency contacts) - Iman Jama

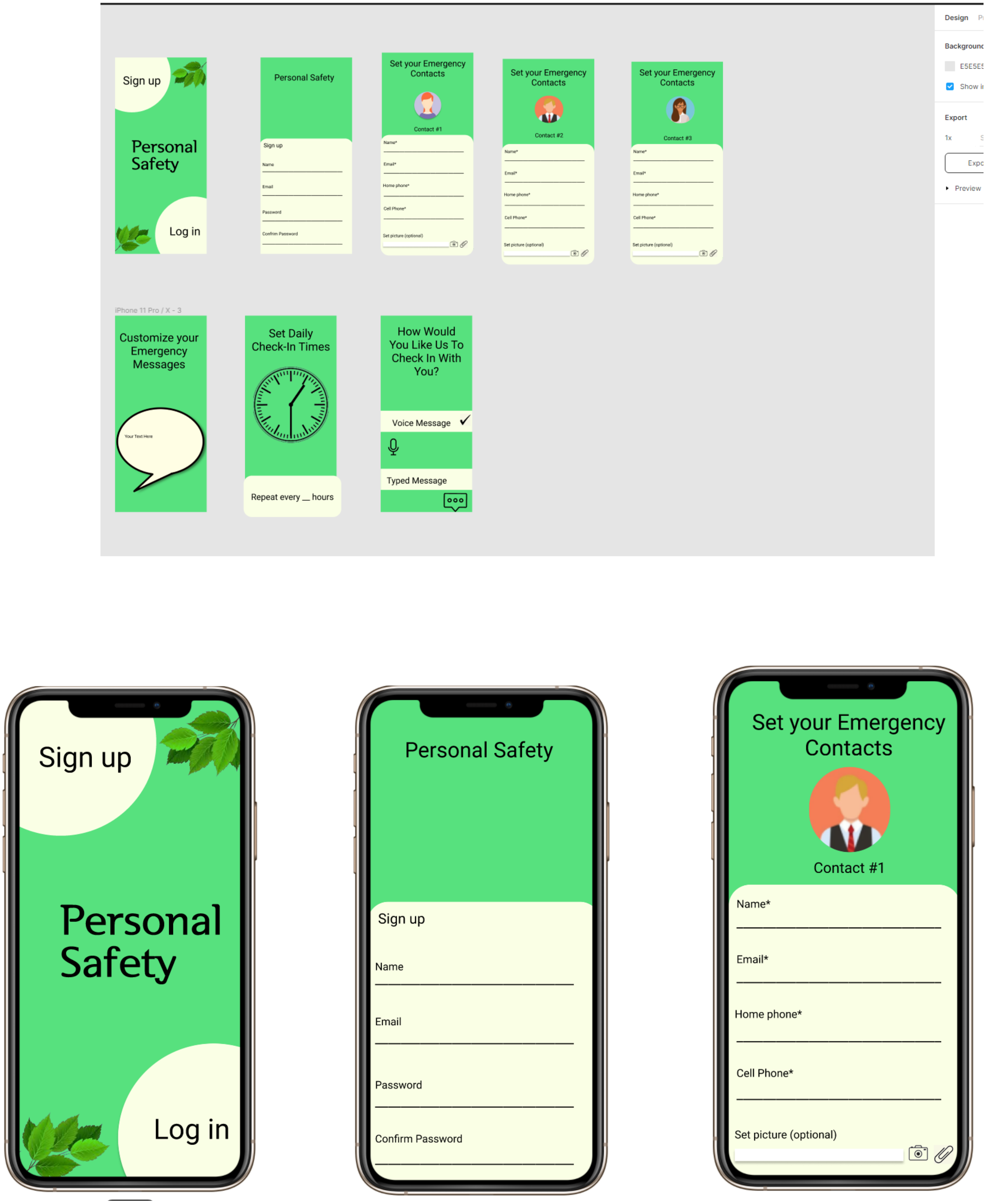


Figure 12: Unscheduled check-in -Darby Martin

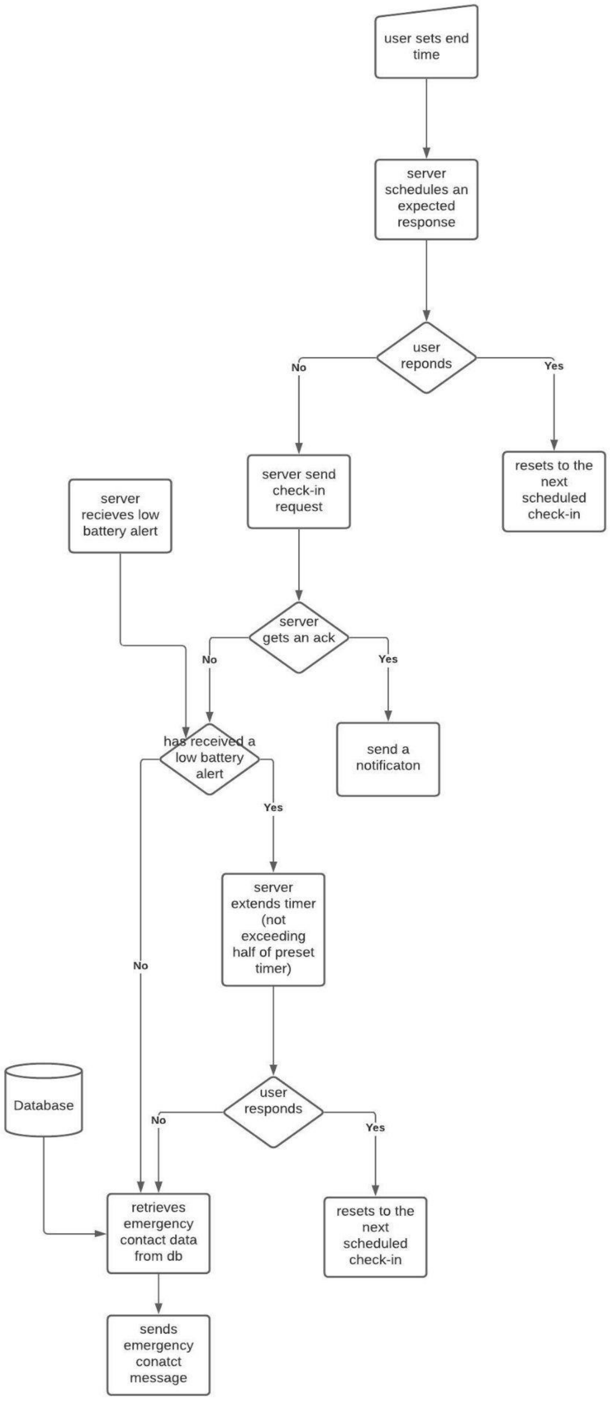

user sets end time
server schedules an expected response
user reponds
No
Yes
server send check-in request
resets to the next scheduled check-in
server recieves low battery alert
server gets an ack
No
Yes
has received a low battery alert
send a notificaton
Yes
No
server extends timer (not exceeding half of preset timer)
Database
user responds
No
Yes
retrieves emergency contact data from db
resets to the next scheduled check-in
sends emergency conatct message

Figure 13: Data in db- Darby Martin

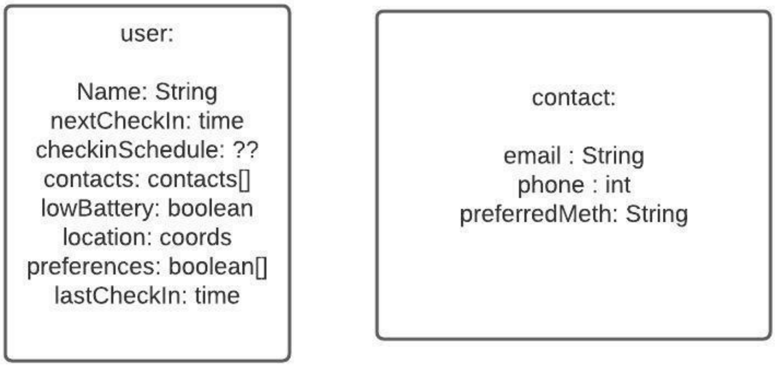
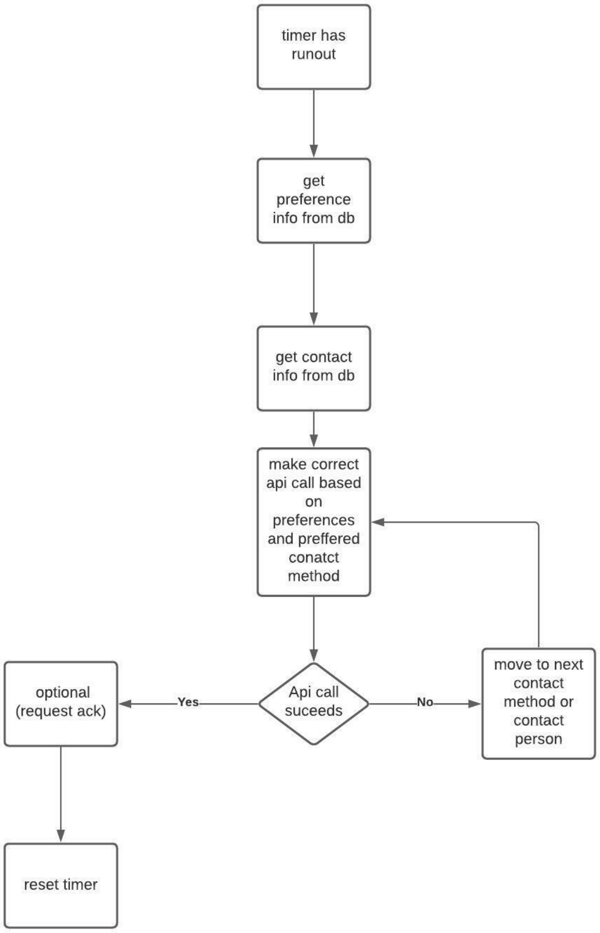


Figure 14: Api call process- Darby Martin


timer has runout
get preference info from db
get contact info from db
make correct api call based on preferences and preffered conatct method
Api call suceeds
Yes
No
optional (request ack)
move to next contact method or contact person
reset timer

Analysis and Evaluation

Table 1: Pro Con Analysis of Each Design Concept

Design (Figure number)	Pro	Con
1	● Gives an overview of the most essential functionalities ● Uses standardized flow chart symbols	● The process is really complex
2	● This flowchart touches upon a topics that were emphasized by the client (reset timer, checks before sending message to emergency contact)	● The client had mentioned they wanted to avoid “button-like” interactions
3	● Easier for us to implement since none of us have ever made a relational database ● Will likely be under 25MB	● Less efficient than design 4 ● Harder to manage
4	● More efficient ● Easier to use once we get used to it ● More structured ● Opportunity to learn new skill ● Will likely be under 25MB ● Easier to manage	● More time consuming to learn (none of us have made a relational database)
5	● The process relies on audio interactions	● We would have to learn how to deal with audio inputs
6	● Very straightforward and simple to implement	● Solely relies on ”button-based” interactions
7	● Turns a complicated procedure into a simple visual	● After the user attempts to proceed with 0 emergency contacts, the flow does not allow for recovery
8	● Corresponds with the color scheme requested by the client ● User-Friendly ● Lists the activities that an average user would partake in	● Missing an edit check in option

	● Shows everything important on one page	
9	● Corresponds with the color scheme requested by the client ● User-Friendly	● Uses complex shapes
10	● It specified how the user has the ability to customize every aspect of the check-ins	● Emergency contacts can only view upcoming check-ins ● The emergency contacts would need to download the app
11	● Matches the color scheme and nature theme requested by the client ● User-Friendly ● Not a very time consuming design to implement	● Too simple ● Missing an edit check in option
12	● Takes into account if the user's phone dies ● Gives a slight time buffer before contacting	● May cause false alarms ● If user missed a check-in, message won't be sent to emergency contacts unless battery is low
13	● Breaks data down into classes ● Very readable	● Simple idea ● Don't know how to implement
14	● Gives flexibility to users and emergency contacts	● Requires many user preferences to be set up

After a lengthy discussion about each design, we came to a consensus on a few ideas. We will be trying to implement a relational database (Design #4 Database Design 2) since they allow for relationships between the data and have better efficiency. None of us have prior experience with relational databases and due to time constraint, we may have to implement a non-relational database. Either implementations would likely be under 25MB which is one of our target specifications listed in deliverable B.

All of our designs for the logic of the backend are all very similar. The back-end designs revolve around providing the user the ability to customize the check-in schedules at each stage. We have a general consensus and understanding of how we will be developing the backend code based on all of our flow charts (1,2,5,6,7,10, 12, 14). We have also agreed on the basic visual of the app. The user interface designs rely on simplistic and elegant aesthetics. The designs have the theme of nature and use the green color extensively (9,11) which were some of the design aspects mentioned by the client in our first meeting. We wish to add more detailed designs and graphics to future prototypes. The client has stated that the visuals were not a priority so we will focus on it after all the other high priority features have been implemented.

The only thing that we have not completely decided on is the platform we will be using to build our app. We have started looking into different platforms like Android Studio and Xcode but we are still uncertain about which will be the best. We have been looking into features that are free and paid. This decision will determine the cost of our project.

Final Group Design Concepts

Afer a group discussion, we came up with the following modifications to the originals concepts we had individually come up with:

- Backend
    - Design # 1: This design incorporates several methods to check-in (buttons, audio, and typing). We'd like to give the user the option to choose which check-in method they like while setting up the check-ins. In this design, when the check-in is successful, the app automatically resets the timer. We'd like to implement a feature where the user can specify if the check-in will occur again rather than it automatically.
    - Design #2: In this design as well, when the check-in is successful, the app automatically resets the timer. We'd like to implement a feature where the user can specify if the check-in will occur again rather than it automatically.
    - Design #3: This design explains how to customize the check-ins. We'd like the user to be able to access the check-in customization page at any page just by accessing the menu tab and clicking on the check-in customization option.
- Frontend
    - Design #8: add the option to edit the check ins from main screen
    - Design #9: add option to add reminders (notifications) for check-ins, add option for how long to wait after a missed check-in before sending a message, add option to specify if the check-in will occur again or if it's only this one time.
    - Design #11: an overview of what the entire app looks like, add a "my profile" option where the user can access their settings to make changes. Covers all the important information to set emergency contact, but make home phone optional rather than required.
- Database
    - Design #4 - would consider removing emergency_contact_#_id and checkin_info_#_id.

New Visual Representation of the Final Group Design Concepts

Updated Backend:

Figure 15: Updated Figure 1: Case: User Attempts to Check In - Sabina to include checking whether a timer needs to be reset

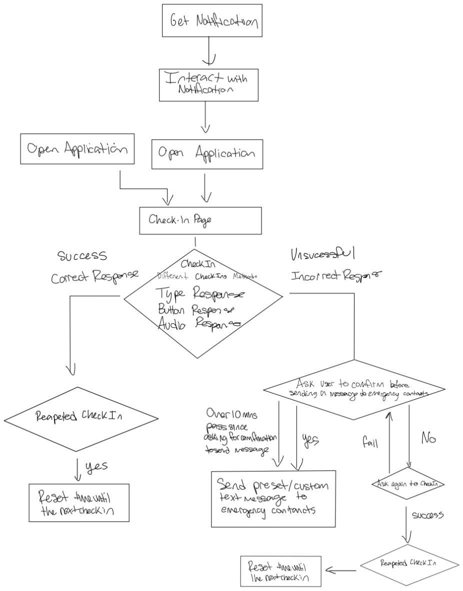
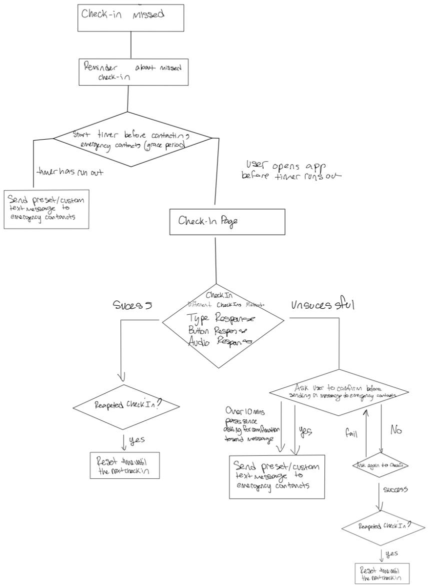

Get Notification
Interact with Notification
Open Application
Open Application
Check-In Page
Success
Correct Response
Check In
Different Check Ins Methods
Type Response
Button Response
Audio Response
Unsuccessful
Incorrect Response
Reapeted Check In
yes
Reset time until the next check in
Ask user to confirm before sending a message to emergency contacts
Over 10 mins pass since asking for confirmation to send message
yes
fail
No
Send preset/custom text message to emergency contacts
Ask again to Check In
success
Reapeted Check In
Reset time until the next check in

Figure 16: Updated Figure 2: Case: Missed Check-In - Sabina to include checking whether a timer needs to be reset


Check-in missed
Reminder about missed check-in
Start timer before contacting emergency contacts (grace period)
timer has run out
User opens app before timer runs out
Send preset/custom text message to emergency contacts
Check-In Page
Sucess
Check In
Different Check Ins Methods
Type Response
Button Response
Audio Response
Unsucessful
Reapeted Check In?
yes
Reset time until the next check in
Ask user to confirm before sending a message to emergency contacts
Over 10 mins pass since asking for confirmation to send message
yes
fail
No
Send preset/custom text message to emergency contacts
Ask again to Check In
success
Reapeted Check In?
yes
Reset time until the next check in

Updated Frontend:

Figure 17: Updated Figure 8: Homepage design - Kian Ashrafganjouei

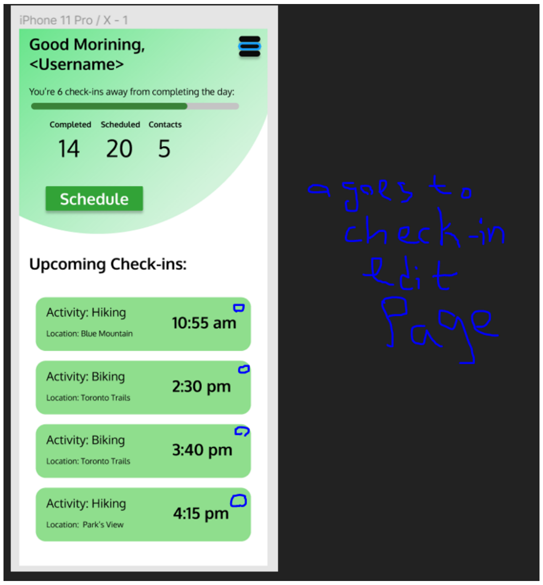


Figure 18: Updated Figure 9: Check-in schedule page - Kian Ashrafganjouei

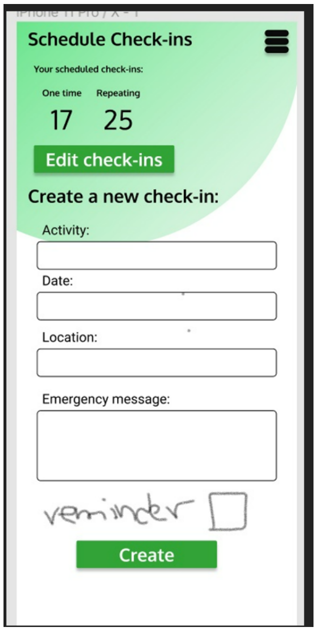
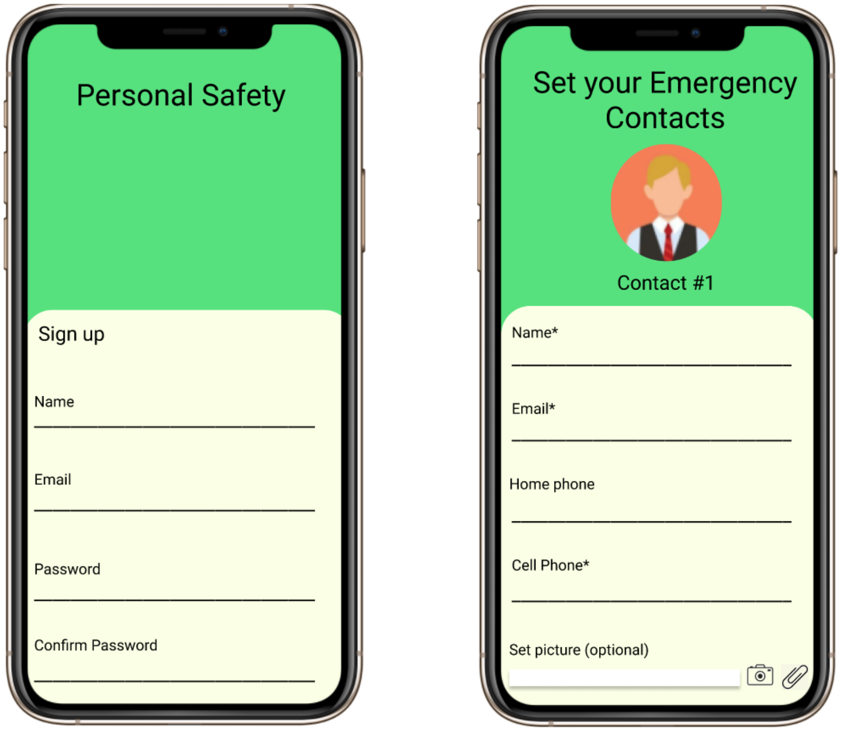
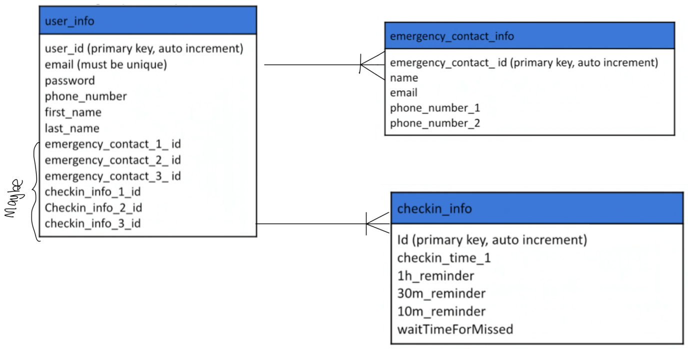


Figure 19 : Updated Figure 11: App Overview for Signing Up (1. Home Page, 2. Sign up, 3. Emergency contacts) - Iman Jama



Updated Database:

Figure 20 : Updated Figure 3: Database Design 1 (non relational)- Sabina



user_info

user_id (primary key, auto increment)
email (must be unique)
password
phone_number
first_name
last_name
emergency_contact_1_ id
emergency_contact_2_ id
emergency_contact_3_ id
checkin_info_1_id
Checkin_info_2_id
checkin_info_3_id

emergency_contact_info

emergency_contact_ id (primary key, auto increment)
name
email
phone_number_1
phone_number_2

checkin_info

Id (primary key, auto increment)
checkin_time_1
1h_reminder
30m_reminder
10m_reminder
waitTimeForMissed

Final Group Design Concepts and Target Specifications

The following explains our concept's relationship to the target specifications.

- Our target specifications required us to develop this product with the IOS platform in mind. We've based our user interface designs around IOS devices accordingly.**(#14)**
- Our target specifications specify that the time between check-ins will be measured in hours and will be decided by the user. We designed the app with the option to edit the time between check-ins. **(#4)**
- Our target specifications specify that our database size should be under 25MB, the updated database design is simple and lightweight to meet this restriction. **(#6)**
- Our UI is simple, which will hopefully allow us to get a Customer Satisfaction Rating for over 7, allow the app size to be under 50MB, allow the app to require under 3GB of RAM, and keep the loading time under 3 seconds . **(#1 , #11, #10, #8)**
- Our backend designs allow for external communication using text, meeting our target specifications **(#13)**
- The target specifications explain that the app will offer customizable notifications. Therefore, we've created an option to receive notifications for the check-in while creating the check-in. **(#3)**

Table 2: Target Specifications

	Metric	**Unit**	**Target Specification**	**Marginally Acceptable Values**	**Ideal values**
1	Download Size	MB	The app size is under 50MB	<100MB	<50MB
2	Price	CAD$	The app is free to use	0$	0$
3	Notifications	List	The application will offer several, customizable notifications	N/A	N/A
4	Time between Check-Ins	Hours	Time between check-ins is decided by the user	N/A	1-24 hours
5	Compatible Regions	Countries	The app works in Canada	Canada	North America
6	Database Size	MB	The size of the	<40MB	<25MB

			database should be under 25MB		
7	Cost for Creating App	CAD$	The app will cost no more than $50 to create	<50$	<100$
8	Loading time	Seconds	The app will take less than 3 seconds to load	<4 seconds	<3 seconds
10	RAM Requirement	GB	The app should use 3GB of RAM	<4GB	<3GB
11	Customer Satisfaction Rating	Number (scale 1-10)	From a scale of 1-10, the customer satisfaction rating will be 8	7	>8
12	Production Lifespan	Years	The app has a production lifespan of 5 years	>3 years	5 years
13	External Communication	Communication Methods	The app will be able to send text messages	Text message	Text message and email option
14	Compatible Operating Systems	OS	The app will function on iOS	iOS	iOS and Android

Conclusion

The design process gave our team members the opportunity to create flowcharts, schemes, UI components, and processes. By the time we had finished our individual designs and discussed them as a group, we had developed the inner workings of several subsystems. The team also evaluated the target specifications set in the previous deliverable to align them with the generated designs. We have created a final design that we will show our client. Improvements will be made after the client feedback and implement them in our prototype.