

# Final Report

## Studio Buddy - SP12 Music Practice Scheduling App

Spring 4/29/2023

Advisor: Sharon Perry

Team:

Will Worthan,

Alex Ingram,

Kaela Callum,

Nicholus Tinto

Website: <https://studentweb.kennesaw.edu/~kcallum1/>

Presentation: <https://studentweb.kennesaw.edu/~kcallum1/Presentation>

GitHub: <https://github.com/worthanwill/SP12-StudioBuddy>

# Table of Contents

<b>Overview.....</b>	<b>3</b>
<b>Background.....</b>	<b>3</b>
<b>Definitions and Acronyms.....</b>	<b>4</b>
<b>Requirements.....</b>	<b>5</b>
1. A login page and registration page.....	5
2. Professor Landing Pages.....	5
3. Student Landing Pages.....	6
4. Scaling Tempo Algorithm.....	7
5. Exercise Progress Tracker.....	7
6. Video Recording.....	8
7. Calendar.....	8
<b>Development.....</b>	<b>8</b>
<b>Project Planning &amp; Management.....</b>	<b>14</b>
<b>Design.....</b>	<b>16</b>
Prototype.....	16
Architecture.....	22
<b>Results.....</b>	<b>23</b>
<b>Test Plan &amp; Report.....</b>	<b>26</b>
<b>Conclusion.....</b>	<b>30</b>

# Overview

Studio Buddy is an app developed using React Native to run on both iOS and Android devices to be used by the KSU school of music to schedule practice exercises for the students. The main use of the app allows for professors who own studios to post practice for students within the studio. Professors can then track the progress of all the students in the studio to see if they have completed their practice.

This project was started in Fall 2022 with the help of Douglas Lindsey, a trumpet professor in the Bailey School of Music at KSU. The project was created with the requirements he had listed in mind. Due to issues with the previous group's code, the project had to be restarted to resume work on the app.

## Background

Douglas Lindsey wanted an application that could implement the GOLD method within his studio classes. The GOLD method involves assigning exercises at a lower tempo and then slowly the student will work their way up to the target tempo. There is an application that implements this, but it was not designed in an ideal way. The project was suggested by Douglas Lindsey to create a flexible and easier to use app

for him to be able to use in his trumpet studio classes. It was then picked up in fall 2022 by a group in the CS senior project class, and has now been continued by our team in the spring 2023 CS senior project.

## Definitions and Acronyms

Cloud Firestore	A flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud.
Gold Method	A way of practicing that has the user start at slower tempos and build up towards the actual tempo.
Professor	The user who will be the owner of Studio Semesters
React Native	An open-source software framework that is used in developing applications for Android and iOS.

Student	The user who will be joining Studio Semesters
User	The active person using the application.

## Requirements

1. A login page and registration page
  - 1.1. The login page can allow the user to register an account using an email and password.
    - 1.1.1. Each user will either be a student or professor.
  - 1.2. The login page can allow the user to input their email and password to authenticate them.
  - 1.3. The login page can allow the user to use a “forgot password” feature.
2. Professor Landing Pages
  - 2.1. The professor page will allow the creation of a Studio Semester.
  - 2.2. Studio Semester will have a professor landing page.
    - 2.2.1. Studio Semesters have the attributes: Registration Code, Creator.

- 2.2.2. Studio Semesters contains all registered students and the Creator can view all the students.
- 2.2.3. Studio Semesters allows the Creator to review a student's progress.
- 2.2.4. Studio Semesters allows the Creator to remove students.
- 2.2.5. Studio Semesters allow the Creator to view uploaded exercises.
- 2.3. Exercises will have a professor landing page.
  - 2.3.1. The Exercises page will allow for the creation of Exercises and the ability to update currently made Exercises.
  - 2.3.2. Exercises will have the attributes: Name, Description, Sample Video, Starting Tempo, Goal Tempo, Weekly Repetition, Category (Etude, Fundamentals, Solo Rep.).
  - 2.3.3. The Exercises page will allow the professor to add any of the made Exercises to any of their Studio Semester.

### 3. Student Landing Pages

- 3.1. The student page will allow the student to enter a registration code, the code will make the student enter a Studio Semester.
- 3.2. The student page will allow the student to view their Studio Semesters.

3.2.1. The Studio Semesters page will allow the student to view the professor and other student's contact information within the Studio Semester page.

3.2.2. Studio Semesters will contain exercises for the Student to complete.

3.2.3. Studio Semesters will allow for the organization of the Exercises by type, date, and completion.

3.3. Exercises will have a student landing page.

3.3.1. Students will be able to see the attributes of the Exercise.

3.3.2. Students will be able to mark exercises as completed.

3.3.3. The system will keep track of all Exercise progress.

#### 4. Scaling Tempo Algorithm

4.1. Within the Exercise on the Student view, it will allow a Student to follow the Gold Method with practicing the Exercise.

#### 5. Exercise Progress Tracker

5.1. The tracker will be displayed to students to view their overall progress for the week in completing the assigned Exercises.

## 6. Video Recording

- 6.1. Students can create a video from the exercises page.
- 6.2. The Professor will be able to view videos made by the Students on an Exercise.
- 6.3. The Professor will be able to comment on each video uploaded.
- 6.4. The Professor will receive a notification when a video has been uploaded and the Student will receive a notification when a comment has been made on their video.

## 7. Calendar

- 7.1. The Calendar will be displayed to students showing what is due for that week / month

# Development

In the beginning the plan was to have a meeting every week to discuss what needs to be done or turned in for that week. Along with that, we planned to speak with Dr. Lindsey, The professor we are making the app for, every 3 weeks. This was to make sure that he was up to date on what was going on. Our first meeting with Dr. Lindsey was to touch base and make sure that we had clarity on what was most important for him to have on the app. We established a plan on what we wanted to



put in the app and the time frame for which it would be added. If we were not able to implement some of the things on the list the next group will have at least an idea of what we wanted to do next. Some of the things that we wanted to accomplish were: exercise progress tracker, google drive/dropbox compatibility, and deployment on iOS and Android.

We quickly ran into some complications with the code that the previous team was using. The app was dependent on Firebase, which is a cloud computing service provided by Google. Firebase has many desirable and easy to implement features that compliment the requirements of the app very well. However, Firebase was not implemented correctly on the previous team's version of the app. Since the previous app failed to compile, the vast majority of the code provided by the previous team simply did not work, as it was not tested. So, as a team, we decided that it would be best to start over and attempt to reimplement the previous team's ideas into a new version of the app with working backend support. As a result, this decision reduced the scope of the project by a significant amount. Instead of adding more robust features to an already existing app, we shifted our focus to reimplementing the basics and providing working backend support and database construction.

Older versions of React and React Native are largely based in JavaScript and use a class-based API to write functions and methods to execute Java code and render screens. However, newer versions of the app released after 2019 instead use a new

API called “hooks”. Hooks are entirely function-based and do not use classes, and are designed to be easier to read and write complex functions (such as functions that use a cloud-based backend service). Since we decided that we wanted to make an app with more forward-facing techniques that are in demand in the programming job market, we used hooks instead of the more familiar and well-documented class-based API. Additionally, while hooks can be used with JavaScript with no issue, we decided to program in TypeScript instead. TypeScript is syntactically very similar to JavaScript, but allows for better compatibility with more forward-facing services like Firebase. TypeScript is also gaining popularity as an alternative or even replacement for JavaScript in the mobile development market, which is another reason why we decided to program in TypeScript instead of JavaScript like the previous team did. However, TypeScript took a lot of independent study in order to understand, since resources for programming in TypeScript are not as plentiful as JavaScript.

Initially, we thought of scrapping Firebase entirely. While Firebase’s functionality was a good pairing for our app’s design, setting up Firebase on a React Native app requires a large number of dependencies and multiplatform knowledge that alternate backend services do not need. However, we spent a lot of time learning the intricacies of React Native and TypeScript, and towards the back end of our development cycle, we felt confident enough to try and reimplement Firebase on the

app, and we indeed have a working version of the app with extensive Firebase support.

Our Firebase project includes two main modules: Authentication and Cloud Firestore. Authentication handles all user sign-in, registration, and user information storage. Currently, the app supports email sign-in and saves a persistent authentication state once the user logs in. All users have access to all Studios for testing purposes, and new security rules can be added to Firebase in order to restrict certain users' access to the Firestore database. The Firestore database itself is a NoSQL database (meaning it doesn't use SQL), but it has a similar structure to other common database management systems that use SQL. Instead of tables with shared keys, data is stored in Collections, which contain Documents. Documents have fields that can be written to and read from, and can also contain nested Collections.

We set up our Studio Buddy database with the following structure. Studios are stored in a collection called "studios". Each individual studio in the collection has its own Document, which contains two fields: "studioid" and "description". Each studio also contains a nested Collection called "exercises". "exercises" contains a Document for each exercise stored in it. Each exercise has a number of fields that are read from in order to display all relevant information, including a title, description, due date, start and end tempos, and an optional video link for reference.

Many of the main functions of the app, particularly those that interact with Firebase, use hooks extensively. Data is usually stored in “state hooks” as opposed to variables. Unlike normal TypeScript variables, state hooks remain consistent throughout the entire component, as they can be instantiated before any rendering is done. Reading and writing to the database from the app is handled by an “effect hook”, which is a hook that acts as a nested function that performs side effects in a React component (i.e. displaying a new screen can trigger an effect hook that does something else at the same time). These effect hooks utilize methods provided in the `@react-native-firebase/firestore` package to read and write data to and from the database. For example, in order to display a list of studios and their information, an effect hook is triggered on screen render in order to search the appropriate Collection (“studios”) and all Documents in order to read their fields, which is then stored in a state hook. When the screen is rendered, the state hook is called to iterate through all of the stored data and display it on a given component. Writing to the database calls a nested function that uses state hooks, which store variables that are passed into `set()` or `add()` methods provided by the Firestore package to create new Documents with specified fields.

While we couldn’t add all of what we wanted compared to the beginning of the semester, the app provides functionality to read and write data to a cloud database that can be shared across all instances of the app. Professors can easily create new

studios and exercises to share with their students, and students can view those exercises and even create personal studios for their own private practice routines.

# Project Planning & Management

To manage the team we used a few methods. Our main form of communicating any due dates or issues was through Discord. We created a Discord server to use to be able to message each other about anything related to the project and to be able to meet in a voice chat each Tuesday at 6:00 P.M. Discord allows for the creation of channels, which acts like different pages to send messages in, which allows us to be able to organize our chats based on specific topics. This made it very easy to be able to find messages relating to specific parts of the project, unlike other forms of communication that will create large amounts of text on one page making it hard to find certain messages again.


We would also conduct some meetings during the time we were in the Senior Project class. Due to some class days being mandatory it was a good way to meet due to all of us being there at the class.

For meeting with Douglas Lindsey, we used Microsoft Teams. Due to KSU using Teams as a way to host online classes, we all had it and so did professor Lindsey. We were able to meet with him over a voice chat on Teams to discuss the requirements he was looking for on the project.

For version control we used GitHub. This allowed us to best manage the programming of the project. GitHub allows for the team to be able to push any

changes in the code to the GitHub repository. This will create a version of the project for the team to be able to view and run, but all the while maintaining a way for us to be able to restore previous versions that are stored in the repository.

The use of the Gantt chart allowed for us to reference parts of the project that were still needing to be completed and to keep track of what was being worked on.

 Senior Project Music App Gantt Chart

# Design

## Prototype

Our prototype consisted of mockup screens of how we were envisioning the app to look like. This contained mockups of the splash screen, login/registration page, home page, account page, studio page in the professor view and student view, exercise page in the professor view and student view, and the calendar page. We created these screens as a template to base our final product on, but none of these designs were finalized; they were made just to show the general idea we had for the screens. Currently at this time we were referring to professor users as teachers. Below are the screens we made.





Figure 1: Splash Screen Mockup



Login/Registration Pages Mockup	
<div><p><i>Studio Buddy</i></p></div> <div>Email <input type="text"/></div> <div>Password <input type="password"/></div> <div><input type="button" value="Log in"/></div> <div><input type="button" value="Register"/></div> <div><a href="#">Forgot Password?</a></div>	<div><p><i>Studio Buddy</i></p></div> <div>Email <input type="text"/></div> <div>Password <input type="password"/></div> <div>Confirm Password <input type="password"/></div> <div><input checked="" type="checkbox"/> Are you registering as a Teacher?</div> <div><input type="button" value="Create Account"/></div>

Figure 2: Login/Registration Pages Mockup

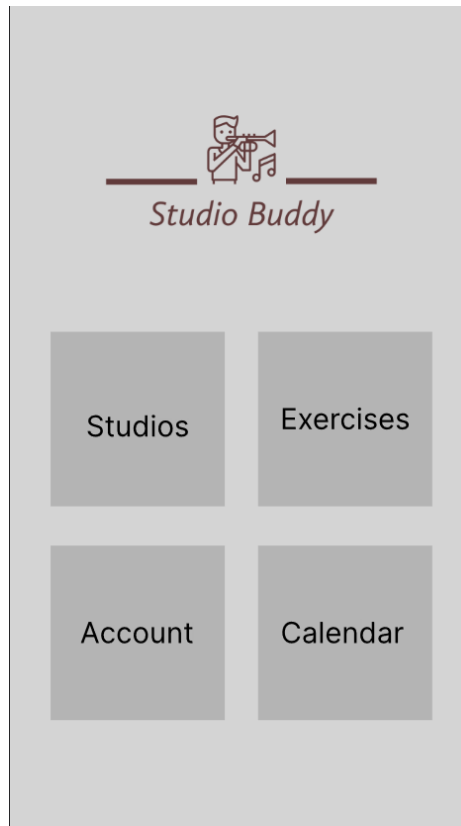


Figure 3: Shared Home Screen Page Mockup

My Account	Edit Account Info
Name <input type="text"/>	Name <input type="text"/>
Email <input type="text"/>	Phone Number <input type="text"/>
Phone Number <input type="text"/>	Birthday <input type="text"/>
Birthday <input type="text"/>	About <input type="text"/>
About <input type="text"/>	Password <input type="text"/>
Studio(s) <input type="text"/>	Confirm Password <input type="text"/>
<input type="button" value="Main Menu"/> <input type="button" value="Edit Account Info"/>	<input type="button" value="Cancel"/> <input type="button" value="Submit"/>
<input type="button" value="Logout"/>	

Figure 4: Shared Account Screen Pages Mockup

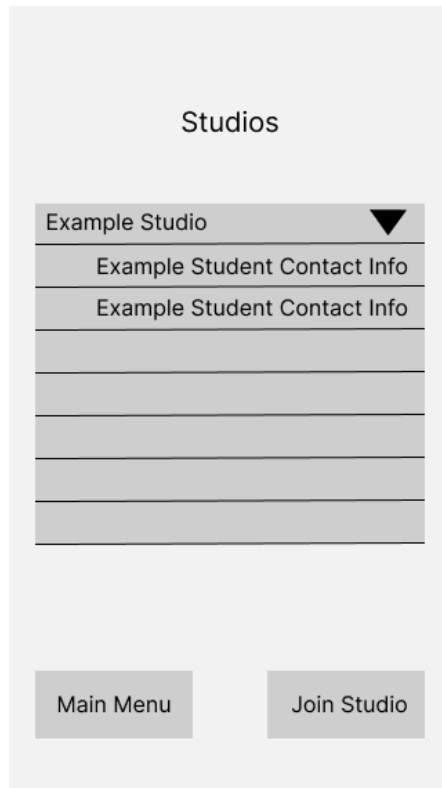


Figure 5: Student View for Studio Page Mockup

## Studios

Example Studio	Edit	▼
Example Student Contact Info		
Example Student Contact Info		

Main Menu

Create New Studio  
(Teacher Only)

## Create/Edit Studio

Name

Studio Key

Back

Create

Figure 6: Teacher View for Studio Pages Mockup

## Exercises

To-Do

Example Exercise	View

Completed Exercises

Example Exercise	3/22/2023	U

Main Menu

## Exercise Title

Video Example

Due Date  
Description

Starting Tempo  
Goal Tempo

Back

Log as Complete

Figure 7: Student View for Exercise Pages Mockup

The mockup consists of three panels. The left panel, titled 'Exercises', contains a 'My Exercises' section with a table listing 'Example Exercise' and several empty rows. Each row has 'View' and 'Edit' buttons. At the bottom are 'Main Menu' and 'Create New Exercise' buttons. The middle panel, titled 'Create/Edit Exercise', contains form fields for 'Title', 'Due Date', 'Description', 'Starting Tempo', 'Goal Tempo', 'Video URL', and 'Studio(s)', followed by 'Back', 'Create', and 'Delete' buttons. The right panel, titled 'Exercise Title', contains a 'Video Example' placeholder, 'Due Date' and 'Description' labels, 'Starting Tempo' and 'Goal Tempo' labels, and 'Back' and 'Edit Exercise' buttons.

**Exercises**

My Exercises

Example Exercise	View	Edit

Main Menu      Create New Exercise

**Create/Edit Exercise**

Title

Due Date

Description

Starting Tempo

Goal Tempo

Video URL

Studio(s)

Back      Create

Delete

**Exercise Title**

Video Example

Due Date  
Description

Starting Tempo  
Goal Tempo

Back      Edit Exercise

Figure 8: Teacher View for Exercise Pages Mockup

The mockup shows a 'Calendar' section for 'March 2023'. It features a large black rectangle with the text 'Google Calendar Integration'. At the bottom is a 'Main Menu' button.

Calendar

March 2023

Google Calendar Integration

Main Menu

Figure 9: Shared Calendar Page Mockup

# Architecture

The main sections of the program consist of Studios, Exercises, Account, and Calendar.

The idea behind the program is that each user will be able to interact with these screens, but some users will act like a professor while others act like a student. All accounts are classified the same, but depending on who the user is they will interact with these screens appropriately.

Users who act like students will be able to view the studios page, join studios, view exercises and mark completion, edit their account, and view the calendar.

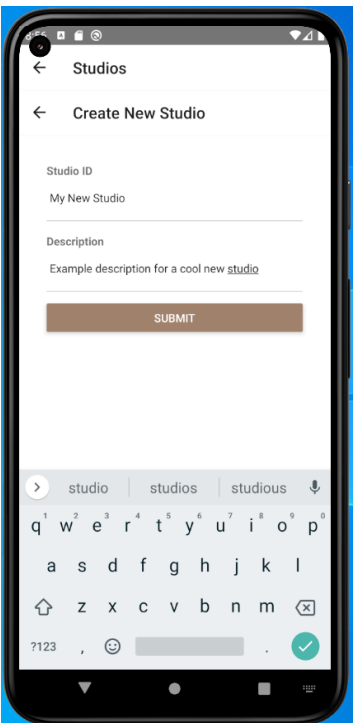
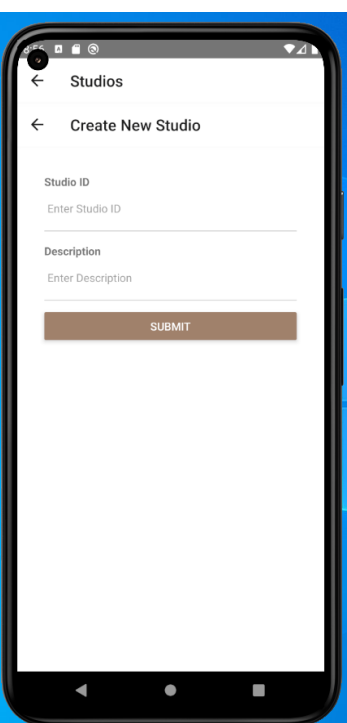
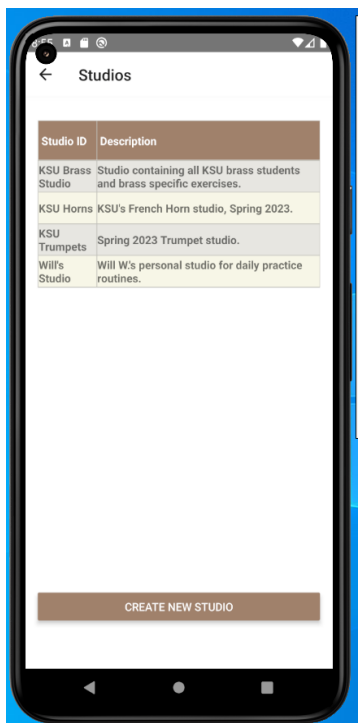
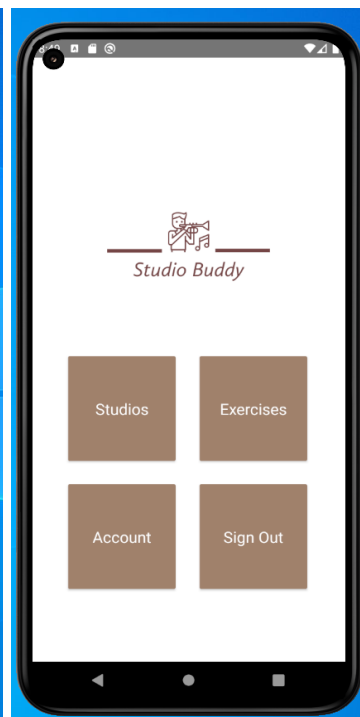
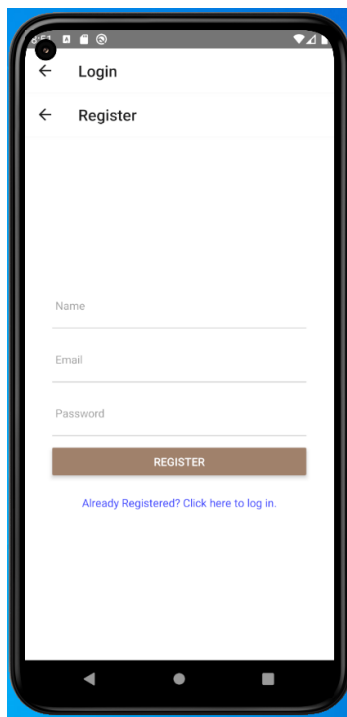
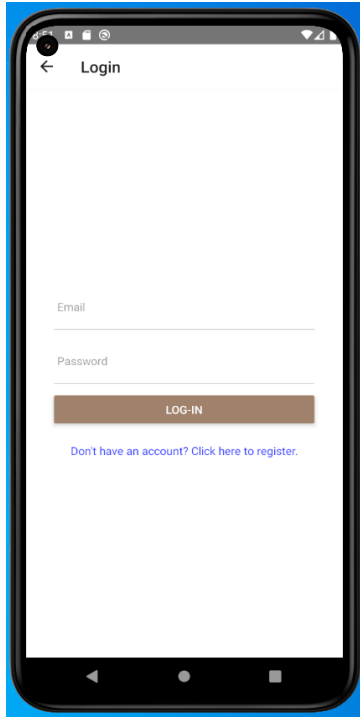
Users who act like professors create studios which will cause the user to create studios, view studios, create exercises, view completion of exercises, edit their account, and view the calendar.

Each user can act like both, but due to the actual users of the app some may be more a professor user than a student user. Some of the screens may not work fully due to time constraints.

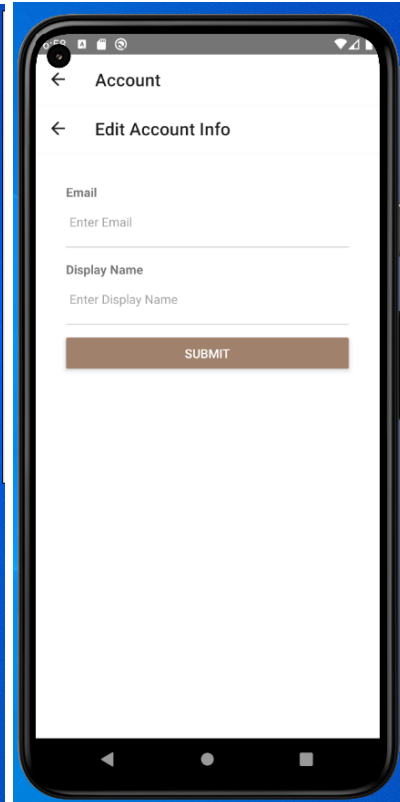
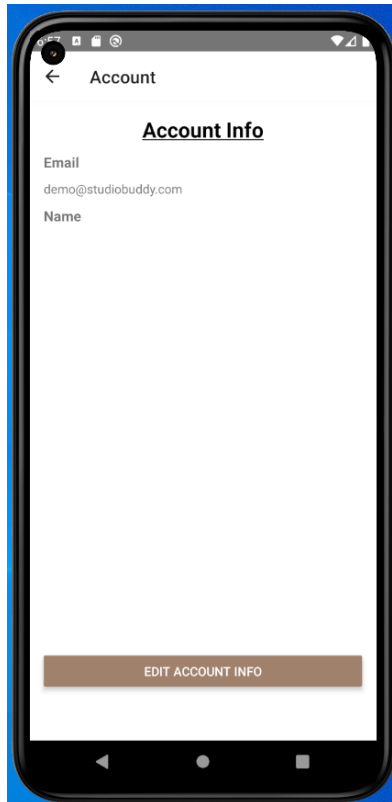
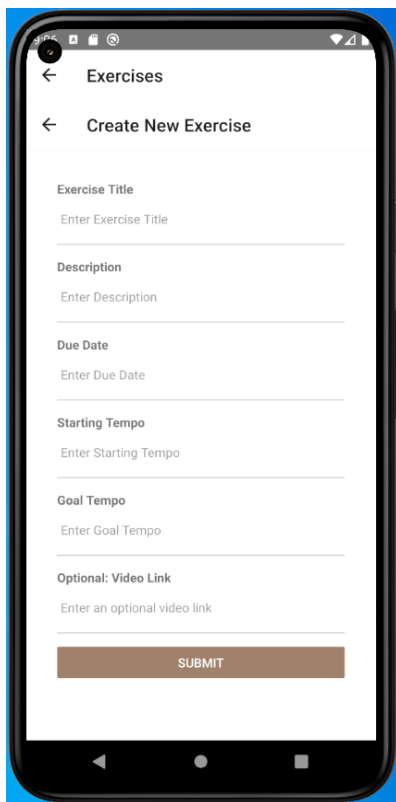
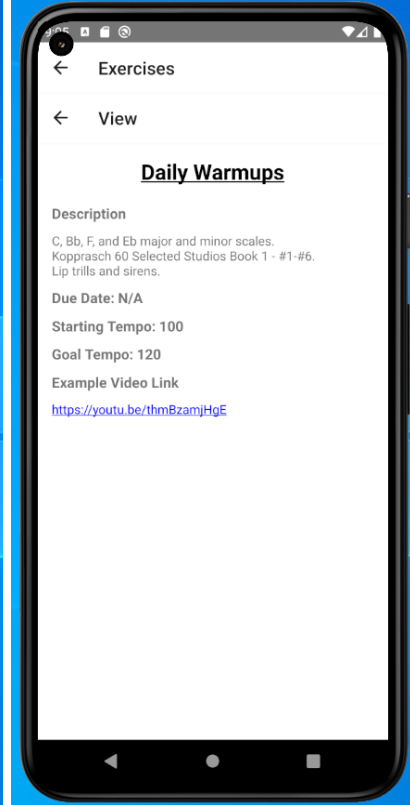
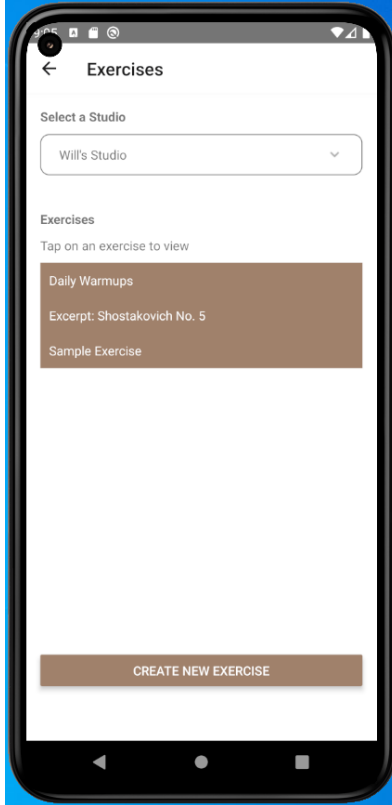
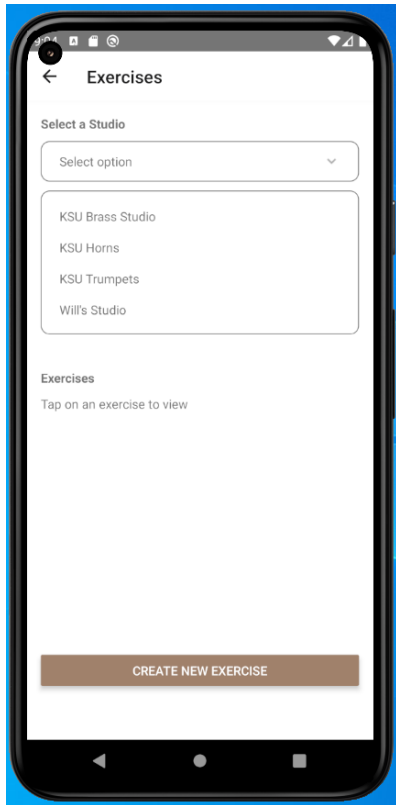
## Results

Our application was created using React Native with a Firebase backend. We were able to make a login for the teacher and the student. There is a main page where you can access your studios, exercises, and your profile. You also have the ability to sign out from there as well. The studio button will take you to the list of all your studios. There is also a tab to create a new studio. It only requires a studio id and a description. The exercise button will lead you to a screen where your studio is listed. Once you select a studio your exercises will show. There is also a tab to create exercises. The account view will lead you to a screen that displays all of your information. The student and teacher views are slightly different. The teacher view is the only one that will have the tab to create exercises as stated previously. We were not able to make the metronome and the google dropbox due to time.

Below are the screenshots of what the team has developed so far:







## Test Plan & Report

Section	Test	Test Result
1. Login/Registration Page	1.1 The login page can allow the user to register an account using an email and password	PASS
	1.2 The login page can allow the user to input their email and password to authenticate them.	PASS
	1.3 The login page can allow the user to use a “forgot password” feature.	FAIL
2. Professor Landing Pages	2.1 The professor page will allow the creation of a Studio Semester.	PASS* Landing pages are not differentiated by Student and Professors.
Studio Semester will have a professor landing page.	2.2.1 Studio Semesters have the attributes: Registration Code, Creator.	CHANGED Studios no longer require registration codes or creator IDs.
	2.2.2 Studio Semesters contains all registered students and the Creator can view all the students.	CHANGED Studios are shared by all users.
	2.2.3 Studio Semesters allows the Creator to	CHANGED Studios are shared by all

	review a student's progress.	users.
	2.2.4 Studio Semesters allows the Creator to remove students.	CHANGED Studios are shared by all users.
	2.2.5 Studio Semesters allow the Creator to view uploaded exercises.	PASS* *This is done on the exercises screen.
Exercises will have a professor landing page.	2.3.1 The Exercises page will allow for the creation of Exercises and the ability to update currently made Exercises.	PASS* *Landing pages are not differentiated by Students and Professors.
	2.3.2 Exercises will have the attributes: Name, Description, Sample Video, Starting Tempo, Goal Tempo, Weekly Repetition, Category (Etude, Fundamentals, Solo Rep.).	PASS
	2.3.3 The Exercises page will allow the professor to add any of the made Exercises to any of their Studio Semester.	PASS
3. Student Landing Pages	3.1 The student page will allow the student to enter a registration code, the code will make the student enter a Studio Semester.	N/A See above. Student landing pages are merged with Professor landing pages.
The student page will allow the student to view	3.2.1 The Studio Semesters page will allow	-

their Studio Semesters.	the student to view the professor and other student's contact information within the Studio Semester page.	
	3.2.2 Studio Semesters will contain exercises for the Student to complete.	-
	3.2.3 Studio Semesters will allow for the organization of the Exercises by type, date, and completion.	-
Exercises will have a student landing page.	3.3.1 Students will be able to see the attributes of the Exercise.	N/A See above. Student landing pages are merged with Professor landing pages.
	3.3.2 Students will be able to mark exercises as completed.	-
	3.3.3 The system will keep track of all Exercise progress.	-
4. Scaling Tempo Algorithm	4.1 Within the Exercise on the Student view, it will allow a Student to follow the Gold Method with practicing the Exercise.	PASS
5. Exercise Progress Tracker	5.1 The tracker will be displayed to students to view their overall progress for the week in completing the assigned Exercises.	FAIL

6. Video Recording	6.1 Students can create a video from the exercises page.	FAIL
	6.2 The Professor will be able to view videos made by the Students on an Exercise.	-
	6.3 The Professor will be able to comment on each video uploaded.	-
	6.4 The Professor will receive a notification when a video has been uploaded and the Student will receive a notification when a comment has been made on their video.	-
7. Calendar	7.1 The Calendar will be displayed to students showing what is due for that week / month	FAIL

## Conclusion

Overall we were able to do a lot more with the redesign of the program than we thought. The program can run most of the mandatory features that needed implementation. With more time we could finish implementing the exercise progress tracker, video recording, and the calendar. From there we could implement more quality of life features for the application to make it an overall better user experience.