12-1-2023

ProcrastiMate: Your Study-Buddy for Getting Things Done

George McAllister Coppel

ProcrastiMate: Your Study-Buddy for Getting Things Done

by

George McAllister Coppel

An Honors Capstone

submitted in partial fulfillment of the requirements

for the Honors Diploma

to

The Honors College

of

The University of Alabama in Huntsville

12/01/2023

Honors Capstone Project Director: Dr. Jacob Hauenstein

George Coppel Nov. 24, 2023

Student (signature) Date

Jacob H 12/1/23

Project Director (signature) Date

Letha Etzkorn Digitally signed by Letha Etzkorn Date: 2023.12.05 14:00:01 -06'00'

Department Chair (signature) Date

Sean Lane Digitally signed by Sean M Lane Date: 2023.12.07 11:10:02 -06'00'

Honors College Dean (signature) Date

Honors College

Frank Franz Hall

+1 (256) 824-6450 (voice)

+1 (256) 824-7339 (fax)

honors@uah.edu

Honors Thesis Copyright Permission

This form must be signed by the student and submitted with the final manuscript.

In presenting this thesis in partial fulfillment of the requirements for Honors Diploma or Certificate from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department, Director of the Program, or the Dean of the Honors College. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

George McAllister Coppel

Student Name (printed)

George Coppel

Student Signature

December 1st, 2023

Date

Table of Contents

Abstract

There are many students with busy schedules and lots of things going on in their lives which makes it difficult to keep a consistent schedule from week-to-week. Unfortunately, one of the easiest things for students to forget when their schedules start to get especially busy is studying. Even a few minutes each week is shown to significantly help students keep up in their understanding of the course load by giving themselves time to think through and process the information on their own outside of class. To help students avoid procrastinating on their work and study, the ProcrastiMate to-do list/study planner application was created. Developed for Android devices, ProcrastiMate goes beyond conventional to-do list applications by integrating task management with a specialized "study" mode, catering specifically to students facing challenges in maintaining consistent study schedules. As the foundation of the application, React Native ensures cross-platform accessibility, while Google Firebase's FireStore cloud database enhances the user experience with robust remote storage and synchronization across devices.

The application's primary objective is to facilitate effective task management and promote structured study routines. ProcrastiMate stands out with a dynamic ordering algorithm suggesting optimal task sequences based on user-defined priorities. The incorporation of a gamified "study" mode helps to encourage students to study by rewarding them. Every week that the student completes their goal, their score goes up. Miss a week, and the score resets back to zero.

ProcrastiMate, though functionally complete, is slated for further improvements as well as the release of a web version to allow for greater accessibility. While ProcrastiMate may not entirely solve procrastination, it stands as a valuable tool for motivated students, with the study streak feature offering a unique motivational aspect to supplement an otherwise traditional and easy-to-use to-do list application.

Introduction

In the ever-evolving landscape of mobile applications, the need for tools that not only enhance productivity but also foster personal development is paramount. The ubiquity of smartphones has transformed them into indispensable companions, capable of assisting users in managing their daily tasks and achieving their goals. ProcrastiMate, a mobile application targeting Android devices, endeavors to contribute to this space by introducing a versatile to-do list application that goes beyond the conventional by integrating task management with a unique "study" mode, designed specifically for students who struggle to maintain a consistent study schedule.

As a tutor for the University of Alabama in Huntsville, I have spent hundreds of hours working with students who are struggling with their classes. Some have specific questions about one or two topics, some are completely lost for one reason or another, and some simply seem to struggle with keeping up in their course(s) even though they are going to class and feel like they are understanding the material well enough until an assignment or exam comes around. After working with so many students and thinking through where their problems are coming from, I feel like there are many students, especially in the 100 and 200 level classes here at U.A.H., who are struggling with time management and scheduling. Specifically, keeping a consistent weekly schedule seems to be a serious issue holding back many students, at least in the computer science and mathematics departments.

As the backbone of this project, React Native is used to allow for cross-platform code sharing, ensuring accessibility to a broad user base. The integration with Google Firebase's FireStore cloud database further enhances the user experience by providing a robust and scalable remote storage solution. These tools enable real-time synchronization of data, allowing users to seamlessly transition between devices while ensuring their information remains up to date.

The primary objective of this application is to facilitate effective task management while promoting a structured approach to study routines. Users can create and organize tasks, assigning deadlines and additional attributes to optimize their workflow. What sets this application apart from other to-do lists, however, is the incorporation of a dynamic ordering algorithm that suggests an optimal sequence for completing tasks based on user-defined priorities. This feature not only streamlines the workflow but also ensures that users can tackle their responsibilities in a strategically efficient manner.

Moreover, the inclusion of a "study" mode addresses the specific needs of students. Users can set weekly study goals, and a built-in timer allows them to track their study sessions. Drawing inspiration from successful language-learning apps like Duolingo, the application introduces a gamified element by rewarding users with a "study score" upon successfully meeting their weekly study goals. However, not meeting the study goal for a week will result in the user's score being reset back to zero.

Throughout this paper, we will delve into the project requirements, the application features, and provide a showcase of the application in action. As the sole developer, I will

then reflect on the successes and failures encountered during the development process, conduct a self-assessment of the project, and ultimately draw conclusions about the impact and future of ProcrastiMate.

Project Requirements

- An Android APK will be created for the application.
- The application will use React Native and Google Firebase's FireStore cloud storage system.
- Users can create their own account with an associated email and password.
    - An email cannot be associated with multiple accounts.
    - All data will be tied to the user's account and only accessible by them.
- The primary use of ProcrastiMate is as a to-do list application.
    - Users can add tasks to a dynamic list.
    - Each task can have the following information associated with it:
        - Name (required)
        - Priority (optional)
        - Difficulty (optional)
        - Estimated Time to Complete (optional)
        - Deadline/Date (optional)
        - Location (optional)
    - If the user gives a task a priority, difficulty, and estimated time, then the app will create a valuation for that task.
        - The valuation will be used to generate a suggested order for the user to complete the tasks in. If a valuation cannot be generated, the task will be placed at the back of the list.

    - The user can sort tasks by the suggested order or by the highest/lowest priority, difficulty, or estimated time.
    - Tasks in the to-do list are not connected to the study goal in any functional way. However, the intention behind the study goal is that it would be used in conjunction with the to-do list. Users can add their tasks to the to-do list for organization, then use the study goal and study sessions system for motivation by gamifying work/study.
- Users can create a weekly study goal for themselves.
    - This is the number of minutes they would like to spend studying per week.
    - The goal can be modified at any time.
    - The user can achieve their goal by completing “study sessions”.
        - Study sessions will use a timer that the user can start, pause, resume, and stop.
        - Each session’s time will be added to a weekly total. When the total reaches the student’s goal, the goal is completed for that week.
        - The next time the user logs into their account after midnight on the next Sunday, their study goal time will be subtracted from their current total and the ability to reach the goal and increment the study score will be unlocked again.
    - Every week that the user reaches their goal, they will have their “study score” incremented.

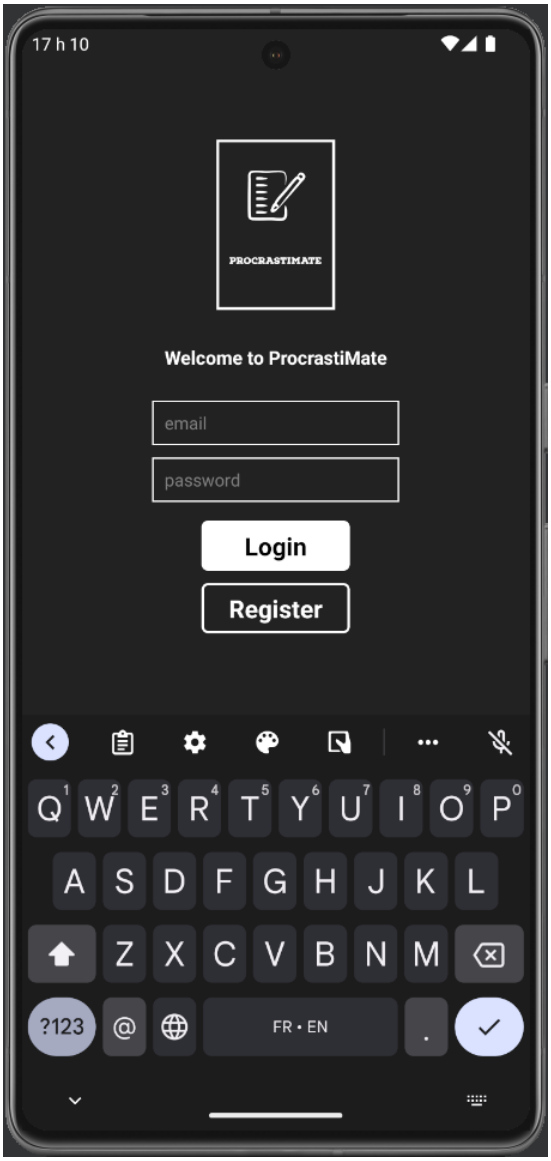
- All user data (email, password, study goal, tasks, settings, etc.) will be stored remotely.
    - Users can sync their data across multiple devices using the same account.
    - Potential for multi-platform support.
- Stretch Goal: Web version of the application
    - Built using normal React (versus React Native).
    - Connect to the same Firebase system for shared authentication and cloud storage as the mobile app.
    - Unlikely to complete the entire thing, but having a basic website connected to the same Firebase systems with the user's tasks viewable after logging in may be possible if the main features have no significant roadblocks during implementation.

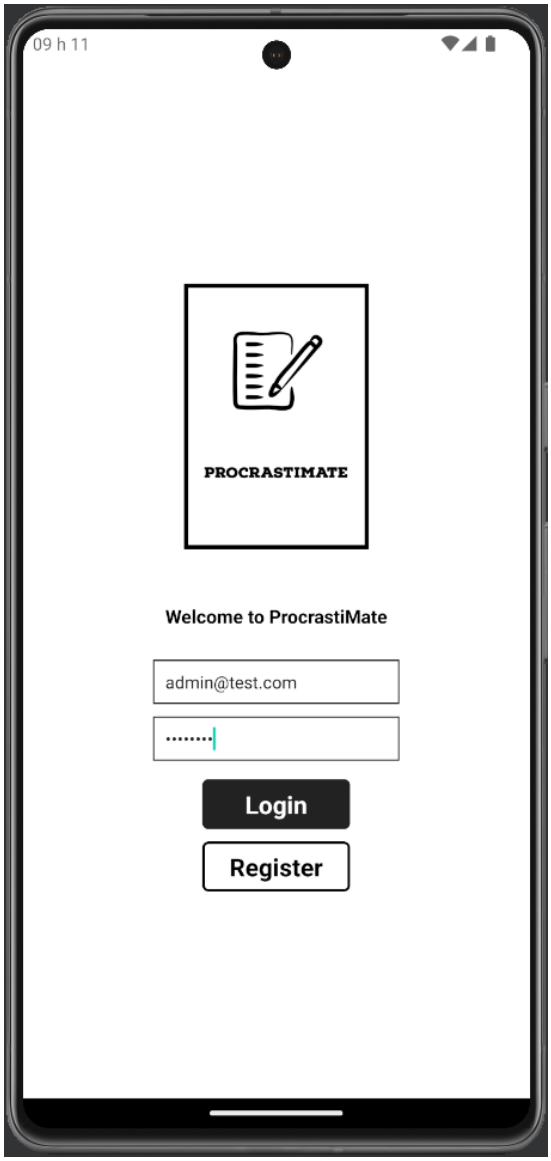
Features and Showcase

This section will review each of the requirements listed in the previous section, but it will also go into more detail about the implementation of the features. Images will also be included for each main page as well as for many of the larger features. The source code for the mobile app and for the web version of ProcrastiMate can be found on GitHub using the following link to my GitHub page or by searching for the repositories titled “ProcrastiMate” and “ProcrastiMate-Web” under GCoppel.

For each of the five main pages, screenshots of both the light mode and dark mode will be provided. There are pop-ups on many pages which will only be shown as dark mode, but light mode is supported throughout all areas of the application. The five sections of the application are: Login, Home, Study, Session, and Settings.

Login:





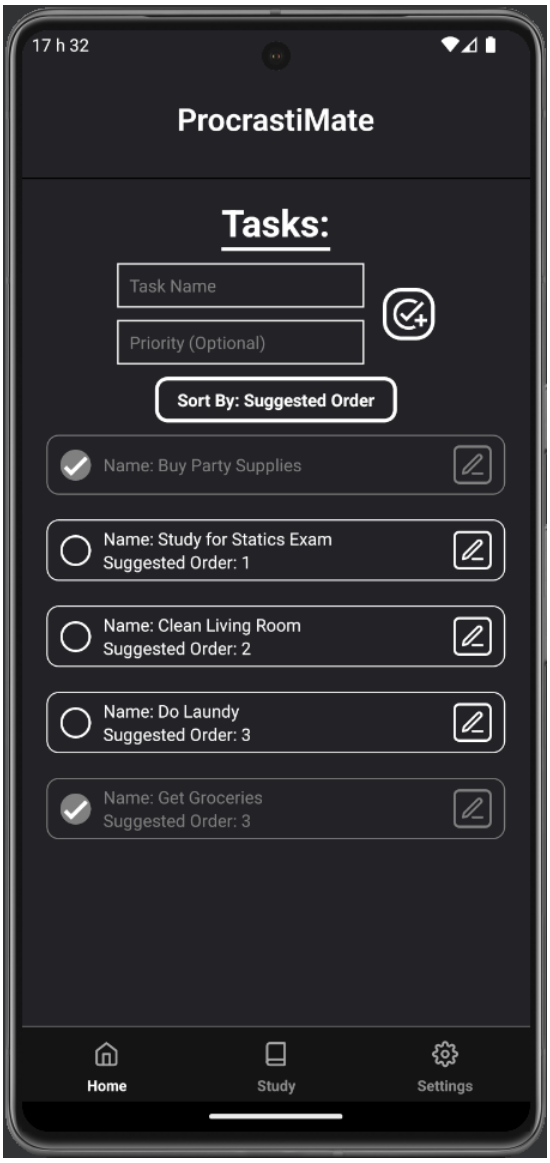
The “login” screen handles registering new users and returners logging into their account. This screen does appear every time the app is opened, but asynchronous storage is enabled for authentication, meaning that the textboxes should automatically fill with whatever the user last used to log in. However, because the user does need to successfully

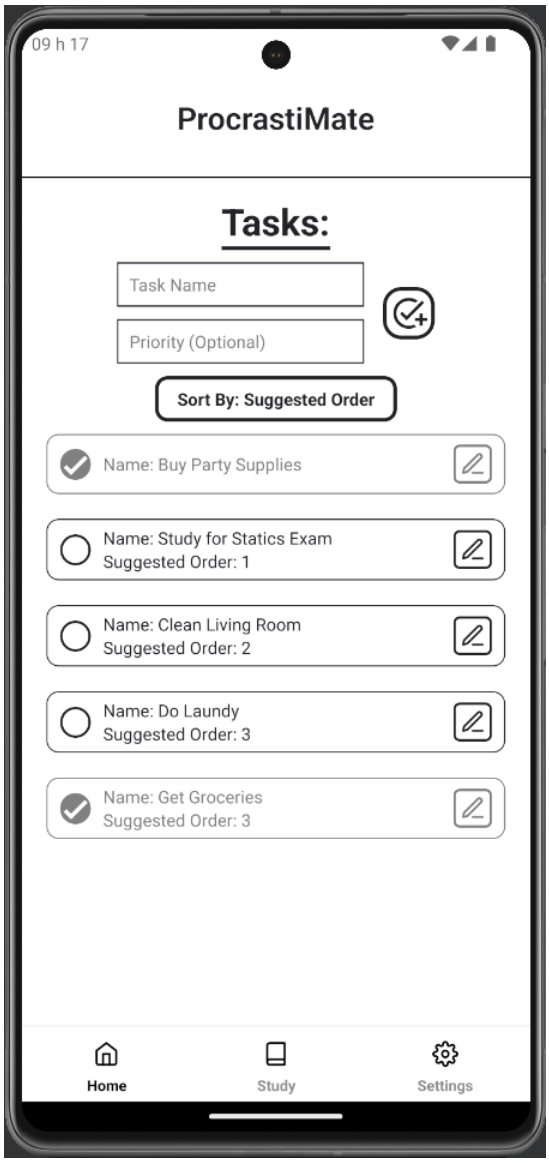
log in before they can progress to the rest of the app, if they are offline then they cannot proceed to the app.

Firebase authentication is used to manage user accounts. When a new account is created, the FireStore remote database has a new document created for the user to store all of their information. A user can, therefore, be logged into multiple devices (at once even) and have their data stay synchronized. The application does not update in real-time because this would require cloud processing to be done efficiently, but it does refresh every page when you open/return to it. So, if you have two apps open and logged into the same account, switching away from then back to a page will refresh it to show any changes.

When a user tries to register for a new account, a popup will be shown to either confirm the user's registration of the new account or to let them know that the account couldn't be created and the reason why. There are no password requirements other than a 25-character maximum and 4 character minimum, but the email address must be a valid address and must not already be associated with another account. If the user's login attempt is successful, they will be directed to the homepage where the to-do list lives.

Tasks (a.k.a. Home):





The homepage contains the to-do list which is the core of ProcrastiMate. Here, users can create new tasks using the textboxes and '+' button at the top which adds the new task to the list and to the remote database. Tasks are organized in a scrollable length with no restriction on size (other than the available memory on the user's device). When the user navigates to the homepage, there is a short delay in loading the task list data from the

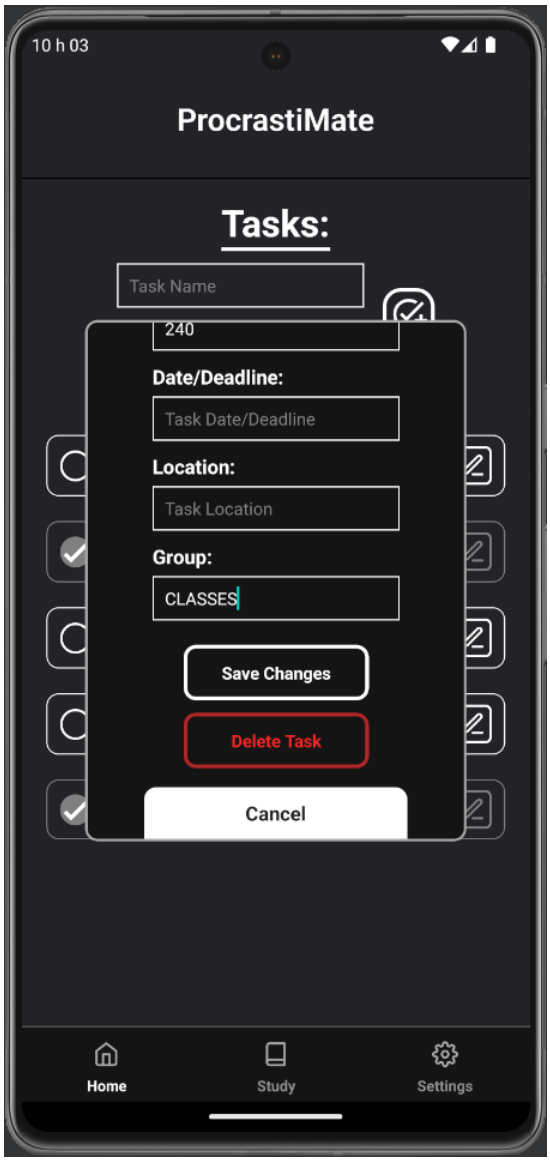
remote database. Typically, this is less than half of a second with a good internet connection.

Once loaded, the tasks can be sorted using the “Sort By” button seen in the above image. By default, tasks are sorted by the “Suggested Order” which is based on the priority, difficulty, and estimated time for each task. At least one of these fields must be set to generate a suggested order value, but all are optional. In the example image, the bottom four tasks all have at least one field set, so they have an assigned value. It is possible for multiple tasks to have the same suggested order value, in which case ProcrastiMate has determined, using the information it was given, that they are equally important. The smaller the suggested order value, the more important. If the user has not given a priority, difficulty, or estimated time to a task, then it will be placed at the beginning of the list as can be seen with the “Buy Party Supplies” task. The other sorting options for tasks are highest priority, lowest priority, highest difficulty, lowest difficulty, highest estimated time, and lowest estimated time.

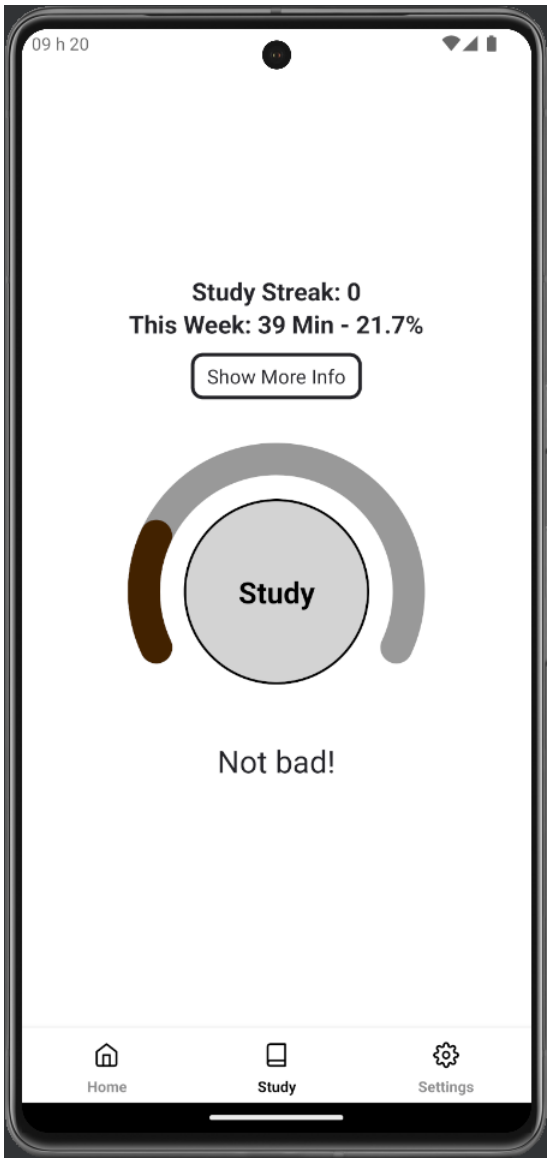
Tasks can be marked as “complete” by pressing the circular checkbox on the left side of a task item. Completed tasks are slightly greyed out to make them stand out less than the uncomplete tasks. Completed tasks are not sorted differently than uncompleted tasks. Once a new week has started, meaning any time after midnight on a Sunday in the device’s current time zone, any tasks that have been marked as complete will be automatically deleted.

To manually delete a task or make any changes, the user can press the pen button on the right side of the task item. This will open a popup with editable fields for all of that

task's data. At the bottom of the edit popup, there is a red button to delete the task and another to save the changes made.



Study:



The other big feature of ProcrastiMate is its study system. Through the “Settings” page, the user can set their weekly study goal which is the number of minutes they would

like to spendy studying/working each week. On this “Study” page, the user can see their progress towards completing their goal for this week.

At the top of the page, there are two bits of information: 1. The number of consecutive weeks that the user has completed their study plan (this is called their “Study Streak”), and 2. The number of minutes they have spent studying this week along with the percentage of total completion of their study goal that those minutes give them. There is also an arced progress meter in the center of the page which visually represents this percentage. The progress meter has a fill animation when you open the page which starts on the left side and eases into the final spot to show the user’s progress. As the bar’s fill percentage grows, the color of the bar shifts from a dark red to vibrant green.

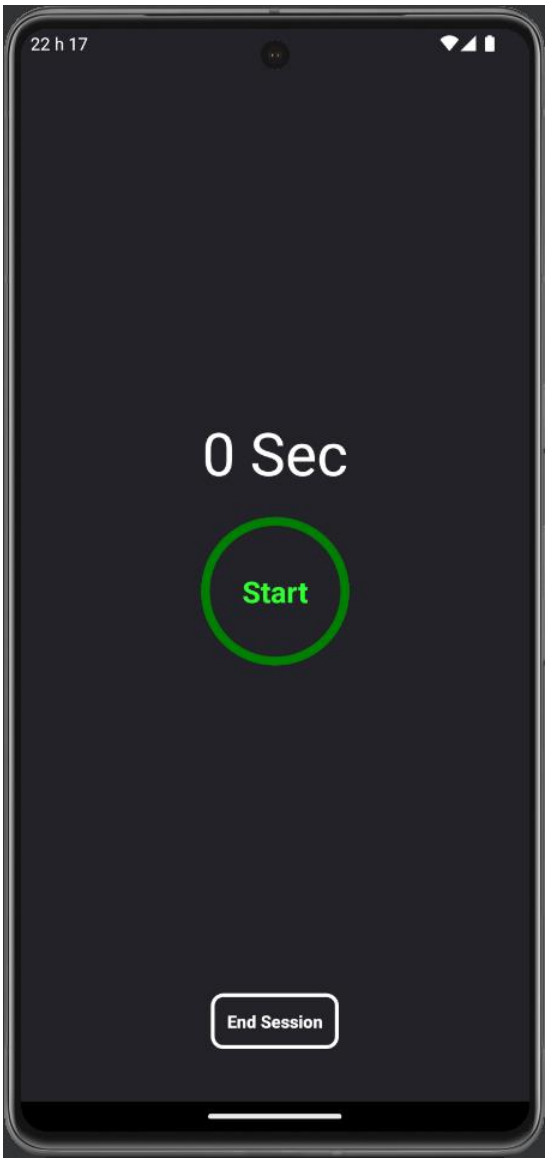
In the center of the page, and inside the arced progress meter, there is a button labeled “Study” which redirects the user to the “Session” page of the application. This is where the user can complete “Study Sessions” which add to the user’s total time and goal progression. More information on Study Sessions and the “Session” page can be found in the next section.

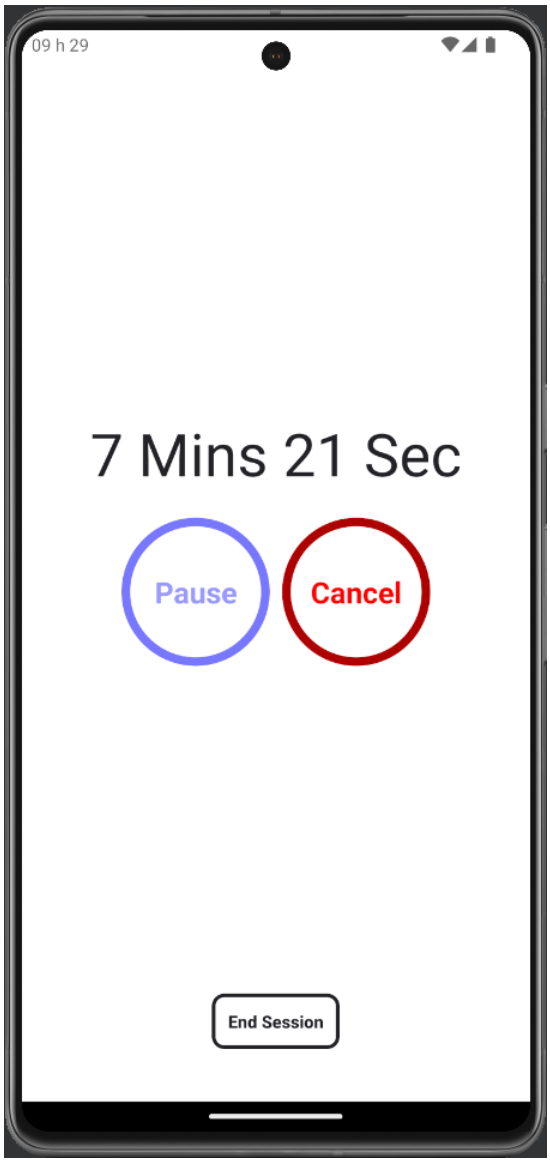
The last visible feature of the “Study” page is the “Show More” button which opens a popup where the user can review more information about their studying for this week. This includes the number of sessions they have done, the average length of each session, what their study goal currently is, and how much banked time they have for this week. Banked time is the extra time that the user spends studying after completing their study

goal for the week. Banked time can be carried over from week-to-week to ensure that there is still an incentive for users to study even after reaching their goal.

Like the tasks, study data is stored remotely using Google's Firebase FireStore cloud system. When the page is opened, there is a short delay before the data can be fetched and read in. During that delay, the fields all read '---' and the "Study" button in the center is replaced with a circular loading icon. The user will normally only see a flash of this icon before the data loads in, but, if they have a poor connection, then the icon will be there to let them know that their data is still loading in.

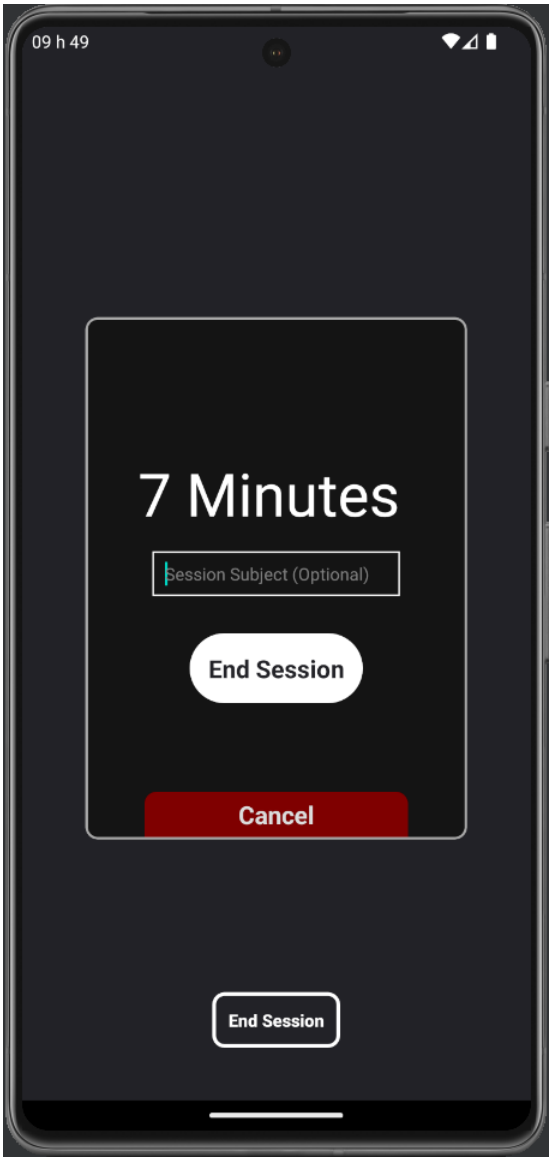
Sessions:





The “Sessions” page is accessible through the “Study” page in the main navigation trio. Here, users can start, pause, resume, and reset a timer which is used to count the number of seconds, minutes, and hours that the user spends studying/working. This session and its time get added to a list which is used to calculate how long the user has

spent studying that week. That total time is then used to determine the user's progress in completing their study plan.

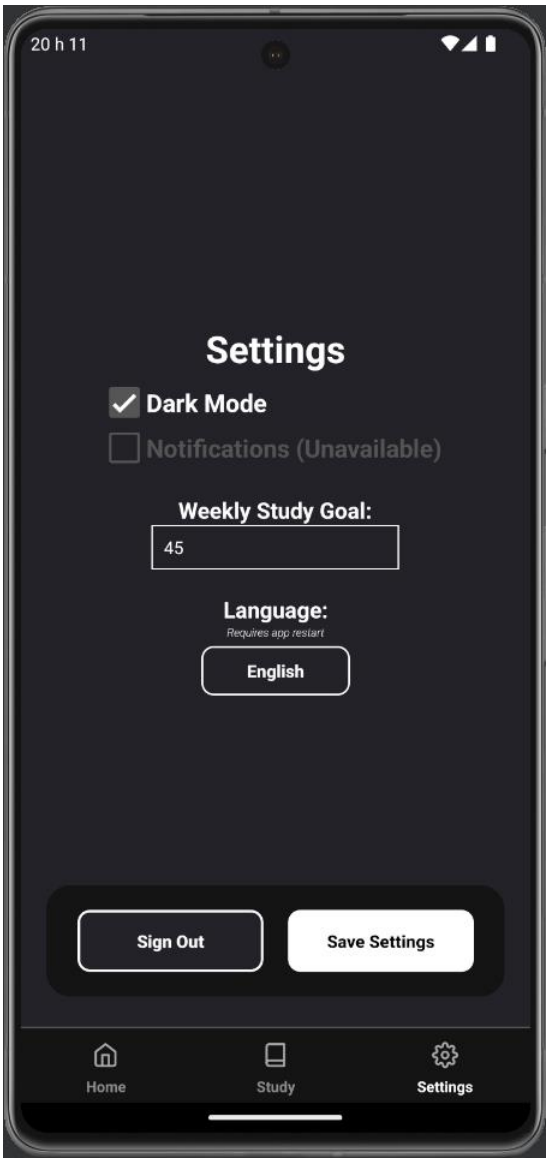


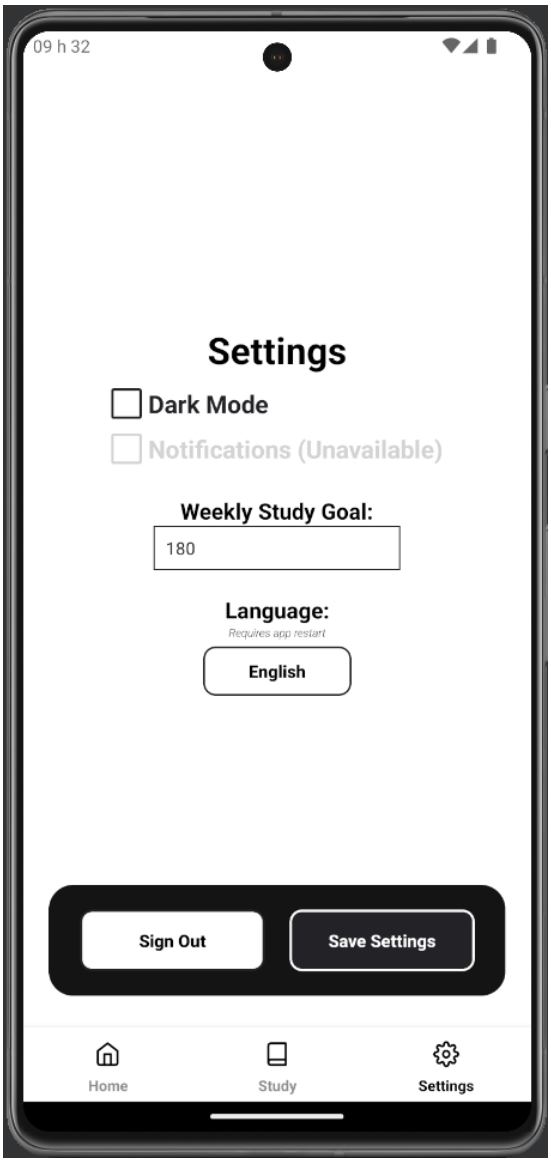
The "End Session" button on the bottom of the page is used to stop the session and add it to the list. Before the session is saved, a pop-up appears which asks the user to confirm that they would like to end the session. The popup displays the number of minutes that have elapsed on the timer, rounded to the nearest minute. Once the user confirms that they would like to end the session, the session is added to the remote database as a new entry in the session list. Additionally, all the other sessions are read in, and the time is totaled for each. If the length of all previous sessions is less than the user's weekly study goal, but adding this session's time is enough to reach the goal, then the user's study streak score is incremented. The user will then be returned to the homepage. If the user navigates back to the "Study" page, they will see that their progress has gone up and, possibly, that their study streak score has increased.

Because the study system works on a weekly basis, the user's progress must be reset each week. Therefore, like the completed tasks, when the user opens the application

for the first time at the start of a new week, the entire list of sessions is cleared. If the user has any banked time from the previous week, this is automatically added to this week's progress. If that banked time alone is enough to meet the weekly goal, then the study streak score is also incremented, and any sessions completed after that will be added to the user's banked time.

Settings:





The user has a few settings that they can change. The most important one is their Weekly Study Goal which sets the number of minutes they need to accumulate in study time per week in order to meet their study goal. The study goal can be anywhere from 1

minute to 999 minutes. Upon returning to the “Study” page after changing the goal, the progress percentage and animated meter will automatically update.

Additionally, if you have already completed your study goal for that week and you set the new goal to a higher number of minutes than you have accumulated, then your study streak score will be decreased by 1. This prevents users from using this feature to cheat and get more than 1 study streak point per week.

Another setting that the user can change is the app theme. By default, light mode is enabled, like most applications. Toggling the “Dark Mode” checkbox on inverts the color scheme on all pages of the application. The dark color is not pure black, but rather a dark grey with a light hint of blue. This is used as the background for the dark mode and the text color for the light mode. All text, icons, and buttons respond to the theme as can be seen in the screenshots included in this section on app features. The theme setting is saved both in the remote database and with local asynchronous storage. Remote storage is used so that the application can stay fully synchronized across multiple devices, even if the theme is not a practical feature. The local storage is so that there is no delay in enabling dark mode when starting up the app like there is when opening the “Home” or “Study” pages which rely on getting their data from the remote database.

The final setting that users *would* have had access to is notifications. This feature was part of the original requirements but had to be dropped due to unexpected technical difficulties and limitations. The setting to enable/disable notifications was not removed entirely in case this feature were to be added in the future, but it is greyed out and disabled so that users cannot turn on the non-existent feature. More information on the removal of

the notifications system can be found in the next section, “Successes, Failures, and Lessons Learned.”

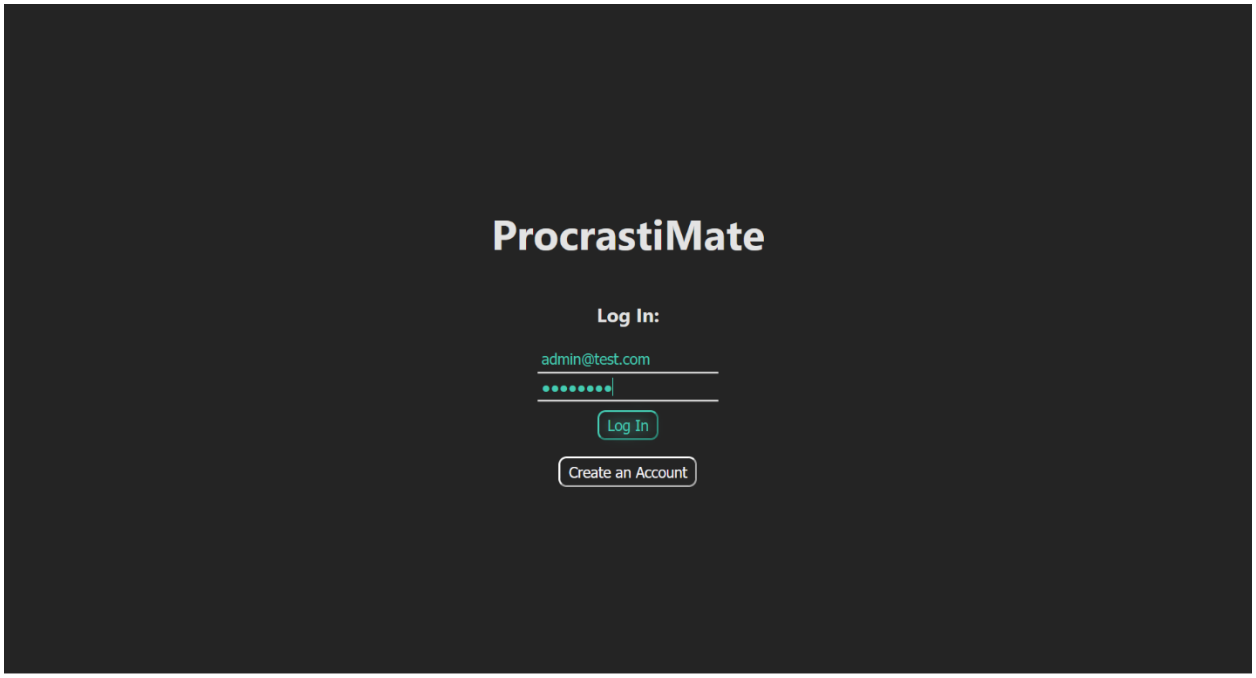
The final setting that the user can change is the application language. There are only two options: English and French, and this change, unlike the others, requires the app to be restarted before it will take effect. By default, the application uses English which is directly contained within the React Native text components. If the currently language is set to French, then the text components will pull from a JSON file containing the French versions of all the text items. Because ProcrastiMate currently only supports two languages, this feature could be simplified by removing the JSON management and putting both languages directly into the React Native components. However, if there were to be languages added later, then the only changes that would be needed are adding the new JSON file and adding a new button for the language in the settings. The active language is stored both locally and remotely so that, like with the dark/light mode setting, all devices using the account can stay synchronized and avoid the split-second delay before the non-default value can be loaded in from the cloud.

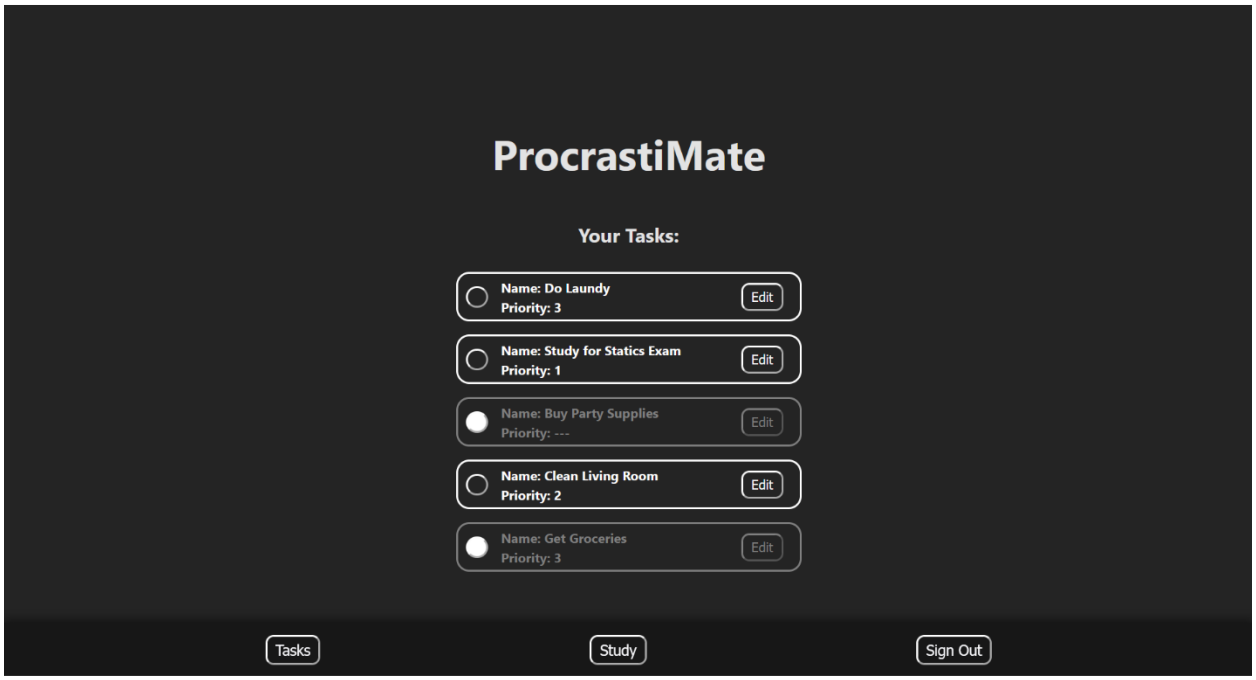
ProcrastiMate for the Web

The stretch goal for this project, once the Android application was completed, was to develop a web version of ProcrastiMate. This multiplatform support makes a lot of sense given that data is all stored in a remote database accessible from anywhere and because the mobile app, being made with React Native, is already written in JavaScript. This makes it fairly straightforward to develop a web application as much of the backend code from the mobile application can be directly copied.

The main challenge with developing a web version of the mobile app is the front-end. Despite using React Native, which shares many similarities with React in how it handles state, updates, and components, the way the two libraries display components is very different. React uses HTML elements at its core while React Native has its own library of components that can be rendered on Android and iOS devices locally. Because of this, nearly the entire front-end for the web version must be re-written. Additionally, packages which may exist for React Native, such as the animated progress meter used on the "Study" page of the ProcrastiMate app, are not always available for React, meaning that substitutions or custom versions must be made.

While not all of the ProcrastiMate mobile app features are available on the web version (yet), having the ability to view and add tasks to the user's to-do list and being able to view their study progress is enough to make the web version useful. On the next page are a pair of screenshots showing the login and tasks pages of the web version of ProcrastiMate. The functionality of the tasks page is identical to the mobile app version except without the ability to sort tasks.


ProcrastiMate
Log In:
admin@test.com
Log In
Create an Account


ProcrastiMate
Your Tasks:
Name: Do Laundy
Priority: 3
Edit
Name: Study for Statics Exam
Priority: 1
Edit
Name: Buy Party Supplies
Priority: ---
Edit
Name: Clean Living Room
Priority: 2
Edit
Name: Get Groceries
Priority: 3
Edit
Tasks
Study
Sign Out

Successes, Failures, and Lessons Learned

With the exception of the notifications system, all the requirements for the project were met or, I feel, even slightly exceeded. There were no big features added, but the quality of some of the items turned out better than I was expecting when planning out the project originally. Whether that be a bit of extra polish with animations/responsiveness or some small bonus features, everything that was implemented works very well. I was also able to get the stretch goal of a web version quite far along as well, which I feel is a big achievement for the planning and time management of this project.

For example, the to-do list tasks have extra data that can be set, more sorting abilities than were outlined in the requirements, and, most importantly, have asynchronous storage enabled so that tasks will persist even when the user's device is offline. When they reconnect, the data will be automatically uploaded to the remote database. The login screen, as well, has asynchronous local storage enabled to help speed up the logging in process by auto-filling the email and password fields with what the user last used. Even the settings use asynchronous storage to remember whether the user wants dark mode to be enabled or what language to use!

And, speaking of dark mode, that brings us to the one "failure" of this project: the notifications system. When doing the research for this project, the main reason I'd chosen Google's Firebase cloud system because I thought it would allow for semi-easy integration of notifications. I'd planned for users to receive occasional reminders about their tasks or

to complete their study goals, but I did not consider that this would require access to the user's data to be useful. I was able to get notifications integrated to the point where I could manually send a notification to *all* users via the Firebase console. However, when I started to get further into sending users individual notifications based on their tasks and study plan progress, things became substantially more difficult. It is possible to get notifications working this way, but doing so requires quite complicated (and expensive) cloud functions. The cloud would have to look through all of the user accounts, read their tasks, check how many days there are until the deadline, see if they've already completed their study plan, and, most difficult of all, determine which device (or devices!) the user is *currently* logged into, then send a notification to that specific device. This quickly proved to be too much for this project, the timeframe, and my personal goals. Therefore, as the notifications are not an essential feature, I made the decision to remove it from the application entirely.

However, because I removed a fairly large quality of life feature, I felt that I had to add a new one in its place. And that is where the aforementioned dark mode got introduced to the project. This feature was not in the original requirements, but I am very glad that I added it because it introduced me to a lot of React's state management tools that I likely would not have discovered or used otherwise. It may sound like a simple feature to implement (just change the background, text, and icon colors, right?), but it did prove to be a fairly long and complicated process. Undoubtedly, some of the complexity came from not planning for a dark mode when I wrote most of the frontend code, but there was significant

challenge to managing the logic for changing colors when using many components, many of which are nested inside of other components!

Another large feature that's about the same size and complexity as the app theme which I elected to add in lieu of the notification system was multi-language support. This bonus/replacement feature allows the user to choose between English and French for the language to be used by ProcrastiMate. This change, unlike the others, does require restarting the app (at least when going from English to French) to take effect due to the use of local JSON files to manage what text will be displayed. I feel that this feature, along with the dark mode, makes up for the lack of a notifications system, particularly when considering that one of my goals for this project was to gain experience designing responsive React/React Native applications. Being able to change the language and color theme are features included in most popular applications, so it is extremely beneficial to have researched how they are usually implemented and to then implement it myself.

While it, unfortunately, could not be included in the final version of the application, I do not feel that the notifications system was a complete failure. The purpose of ProcrastiMate, aside from being a tool that I genuinely believe could help people study and work better, was to give me an opportunity to learn more about the development tools that I am most interested in currently. Getting the experience of working with remote databases

and asynchronous data storage will undoubtedly be beneficial going forward, especially at a time where cloud computing is more popular than ever before.

The experience learning React Native, while I am unlikely to use it professionally anytime soon, is also a relief for me to have gained. Having not had much experience with web development beyond simple personal projects while pursuing my bachelor's degree, I wanted an opportunity to spend a good amount of time learning web/frontend tools like React. React, as the most popular foundational tool for web apps currently, is extremely valuable to have proven experience with. However, I have built websites before, and I wanted to do something more unfamiliar with this project, hence the reason I chose React Native. React Native is very similar to React, the web development tool, but it is used to build local apps for iOS and Android. By choosing React Native as my application's platform and Google's Firebase as its remote database, I am gaining experience with mobile development, web development, cloud authentication, cloud storage, local storage, and more.

Self-Assessment

The first few weeks of this project were extremely productive. So much so that I was already starting to think about the stretch goals less than a third of the way through my time. However, when I started to get more into managing both the remote and local asynchronous storage systems as well as making the app more dynamic and responsive, things quickly slowed down. I still feel that I managed to get everything I wanted to do completed, and my personal goal of learning more about React, mobile development, cloud functions, and what type of development I most enjoy was very successful.

Since building a website using mostly vanilla JavaScript over a year ago, I've believed that I prefer working with front-end development more than back-end development. I've never thought that I've particularly disliked back-end development, but I enjoyed working on the more visual projects much more, it seemed. The problem with this was that the back-end heavy projects were always for school or work while the more front-end focused programs were my own personal projects, and who doesn't prefer working on personal projects over assigned work? Because of this, I wanted to do a project where I would balance a near equal amount of front-end and back-end work. I think that plan turned out well, although there is a bit more front-end work that was needed due to my having to learn React and React Native. I still believe that front-end work is slightly more satisfying for me, although, as I expected, the back-end work can still be fun when it's interesting.

As for my success in learning React and React Native, I am very happy with my progress and newfound understanding. React is a very large platform with a lot of features, quirks, and intricacies, but I am confident that I have a good understanding of the library's fundamentals as well as a handful of its more advanced features. Reading through Meta's documentation and beginning tutorials was extremely helpful and a large part of why I started so strongly on this project in the first few weeks.

Another tool that I had no experience with was remote databases and, specifically, Google's Firebase tool suite which included their FireStore cloud database and authentication tools. Firebase is a popular tool used by many large companies including Sony, Venmo, and even Duolingo, coincidentally. It was fairly straightforward to get the authentication set up and working, but the FireStore system took a few weeks before I fully understood how to use it and keep it synchronized with the app's local storage. Now, I feel like I have a much better understanding of API usage and synchronization methodologies, thanks in large part to Google's documentation which I made an effort to use instead of simply trial-and-erroring until it worked. Of course, the notifications system, which was supposed to be powered by Firebase, wasn't a success, but I still learned a lot about how it works before coming to the conclusion that it wasn't realistically implementable given the timeframe.

I did my best to keep a steady momentum throughout the project by adding at least one significant feature/fix each week while still taking care not to burn myself out. Looking back at my commit history for the project, I am very happy with my consistency in working

on this project while balancing the rest of my workload throughout this last semester of college. I completed the project almost exactly when I had hoped to when planning the project requirements, and I was also able to get quite far into my stretch goal as well. And that's all with a few small features added into the mobile app which weren't part of the original requirements.

Conclusion and Next Steps

Even though the requirements for this project have all been completed, I am still planning to add a few more features, do a bit of polishing and code cleanup, and to finish the web version of the application within the next few months. This is already the largest personal project that I have made, and I want to make it as good as it can be so that it can be used, for now at least, as the showcase project(s) on my GitHub. The React Native mobile application is functionally complete and only requires a bit of polish in certain areas, and the React web application can borrow a lot of the functionality from the mobile app, meaning it is also nearly complete.

I am extremely satisfied with this project, how it turned out, and what I learned while completing it. There were a few hitches along the way, and some things did not quite go the way I had wanted, but the project was a big success overall. There is a real possibility that ProcrastiMate will find its way to the Google Play Store in the next few months, and, if it does, that I will also likely publish the web version as well by using Google Firebase's hosting service. I do not expect many users to download and use the application, but it could be a worthwhile experience just to have shipped an application.

The original purpose behind ProcrastiMate, other than being a project I can use to learn more about React, cloud tools, and my development preferences, was to be a tool for students to help overcome procrastination. I do think that the app could be effective for students who want to improve themselves, but it is not a tool that can solve the problem

entirely. The study streak feature, I believe, will certainly help students motivate themselves, especially once they've accumulated a decent score which they don't want to lose, but the student must have a good amount of discipline already for it to be effective.

As a tutor for the University of Alabama in Huntsville, I would like to see at least a few students try using the app and to get their feedback, but as this is my last semester, I doubt I will be able to get much engagement with it. That said, there are a few students who have been regulars over the past few semesters who I do plan on reaching out to and seeing their thoughts on the application. Even if it doesn't end up serving them well, I am very happy with ProcrastiMate from my perspective as the creator.