Assignment 2: A ~~Tender~~ Hinged Attempt with RN

Due Friday, Feb 3rd, at 11:59 PM.

Overview

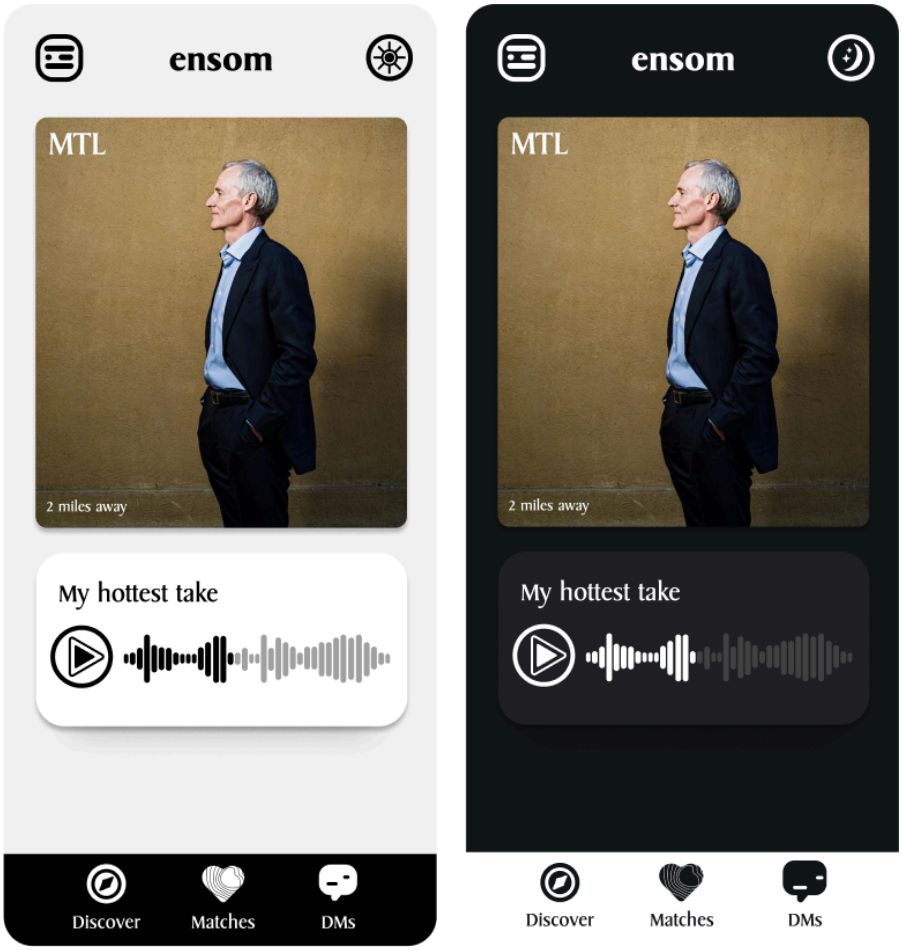
Imagine working in a team as a mobile developer. You have a team of designers with you. Their task is to design the final version of the application, and your task is to turn the designs into a functional app. Your designers are picky. For the next project, they want you to mimic their specifications exactly.

Your job is to take the design, break it up into pieces, and turn it into code. There is no need to do it in one run. We understand that working with React Native will look different than other frameworks / programming languages you might be used to, but we hope that this assignment will help you get familiar with the framework.

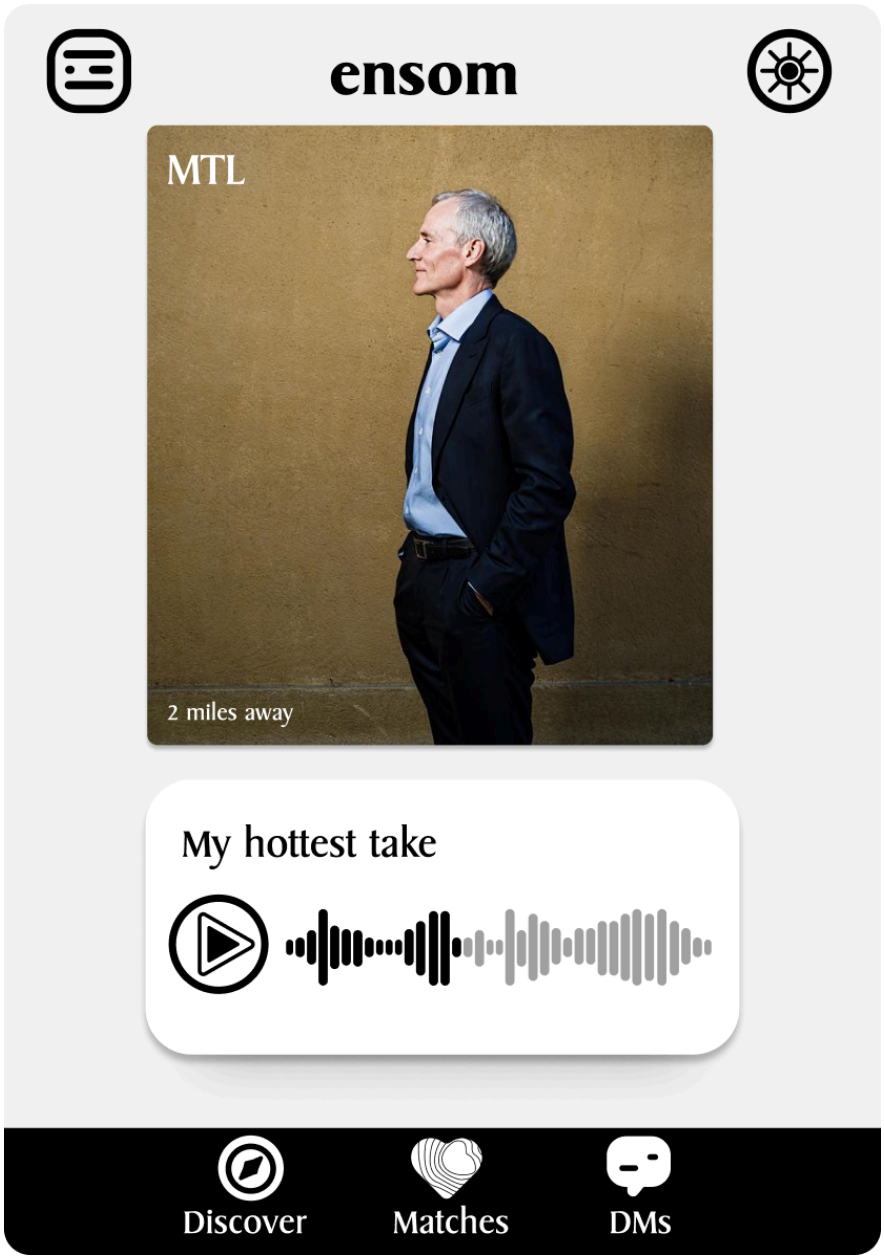
Design Details

Note: your version will deviate from the images below in having 1 mile away as the caption kayle instead of 2 miles away

To get your feet wet, your team of designers have provided these screenshots for you to mimic:



(Light theme on the left, dark theme on the right)



(mobile device interfaces first, and tablet interface next)

*Note: You only need to implement **ONE** of the themes!*

Alas! It's a hinge-inspired clone. Since this is your first project, the designers intentionally didn't set any strict sizing, padding, or margin constraints. They want you to interpret the interface seen above and create a responsive application based on it. Your app should look reasonable on different screen sizes (mobile and tablet), and should run on both iOS and Android. You may assume the app will only ever run in portrait mode.

Your designers have sent you the files of the images, icons, and colors they want you to use (look in your Themes and Assets folders).

Your designers have also left a chart on your desk. It summarizes the following specifications, item by item:

Item (with design value)	Description
Background Color (+0.5)	Use any light color of choice. There must be some contrast between the background and the action icons on the bottom of the screen.
Use of theme object (+1)	We didn't cover themes in class explicitly, but they are essentially JS objects that contain most of the properties that you'll need (e.g., specific colors, shadows, etc). Themes are incredibly important in modern UI design and we want you to get some practice with them! *Hint:* specifically, we're talking about the Themes object which you can import as such: `import { Themes } from "./assets/Themes"`
Header (+1)	*(contains app name and two adjacent buttons)* Should have a height of ~54px on Android and ~41px on iOS. *See Platform module.* For the title text font size, pick something reasonable (e.g., 32)
App Logo (+0.5)	Must be centered. Sizing is flexible.
Nav. Bar Icons (+0.5)	Should be aligned with the bottom image (of MTL in the images above).
Profile Card (+0.5)	Matches screenshot. Add a shadow all around (see themes object in assets/themes/index.js), AND corners must be rounded.
Profile Picture (+1)	Should scale with screen-size but must always be a rectangle (1:1.1 aspect ratio).rea
Profile Details (+1)	Please match the styling of the name, and description on the card profile. Although this is flexible, the font-sizes we used are 32 and 18.
Audio feature (+1)	Rectangle with all rounded corners (note: they're more rounded than the profile picture). Shadows around it as well. Please try to match the alignment of the elements!
Bottom Action Icons (+1)	The 3 action icons at the bottom of the screen must be in the same order as the screenshot. The exact size is up to you.

Note that each item includes a design value number. This is representative of how important each element is to the final design of the application and will be used for grading.

You're welcome to exercise creative freedom for any item that is not listed above. With that said, if you choose to add something new, make sure to report that you have done so and we will give you extra feedback on it.

Grading

A total of 8 value points can be earned by satisfying each of the design specifications printed above. You will then have an opportunity to earn 6 additional points, as detailed in the following breakdown:

Meeting the Design Specifications (matching the screenshot)	8 points total
Appropriate use of Stylesheets and Flexboxes	2 points
Application runs on iOS and Android (Mobile + iPad)	2 points
Appropriate use of components (basic and custom)	2 points
Autograder (extra credit) -> for Body and Header only	6 points
Total	**20 points**

Note that we want you to get exposure to the intricacies of putting various layouts together. We want you to feel comfortable moving things around and experimenting with different ways to present information in this "Hinge" clone. With that in mind, and up to our discretion, we will be rewarding bonus points for novel solutions and extensions.

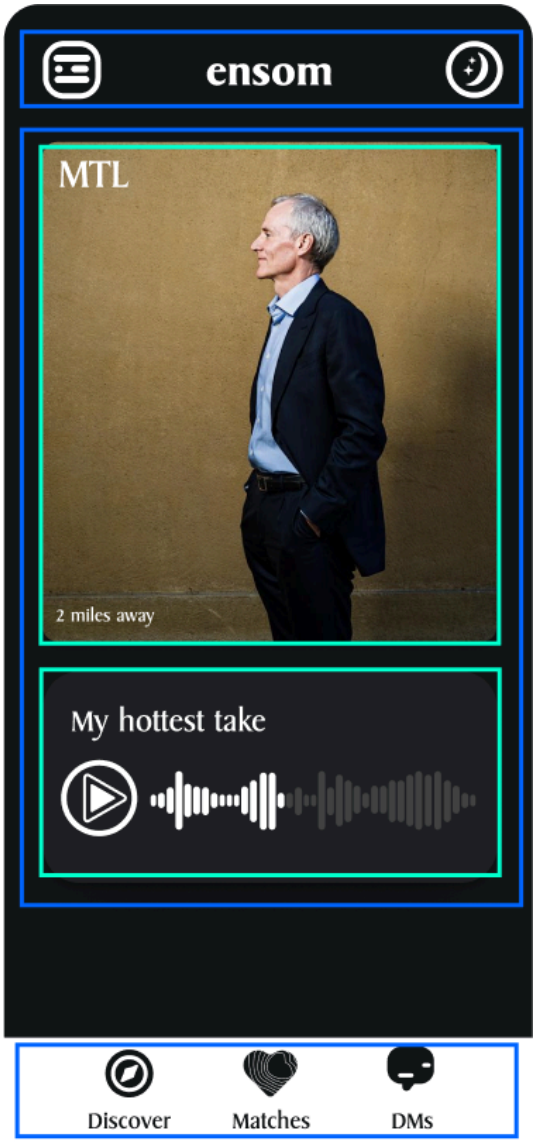
Extensions (for even more extra credit!)

Note: Please cite (both in your submission and code via comments) any code that you end up using/borrowing from the internet.

Here are some extensions that we're hoping you consider. Doing so can greatly expand your grasp of this React Native material early on in the quarter:

- Implementing both themes. Simply designing both will give you half a point (+0.5). A more advanced version is to implement themes using hooks which are a bit more of an advanced practice (this medium article is a great start for that). This will earn you +3 extra points.
  Note: Oh no, you've run out of your free member-only views on medium. I sure hope this link isn't a pdf of that article 😉

- Make the cards swipe-able (animation). This is a good opportunity to experiment with implementing a third party library: [React Native Tinder Swipe Cards](). +3 bonus points can be earned here.
- Enable the use of all of the buttons in the bottom action bar. Make it so that a “swipe right” or “swipe left” action takes the user to the next picture. Experiment with state to see how to achieve this (See Lectures from week 3). Getting every button working is worth +2 bonus points. Get creative with the functionality of the other buttons.
  Note: This extension may depend on the previous one (swipe cards) depending on how you go about implementing the swipe gestures. See also [gesture handler]()
- In addition to what you have implemented so far (the designs above), if you feel like you might have a better design in mind/trend that you want to pursue (e.g., glass-morphism, neomorphism) feel free to reach out to us and we will reward you generously for your implementation! We suggest looking at [Dribbble]() for inspiration!



Implementation Details

Knowing that this was your first time working with React Native, your co-worker left you this short guide on how to best approach the project.

1. **Break up the design into smaller pieces.**
   Trying to turn the design into code all at once could turn out messy. Before starting to write code, look at your design and try finding different

layouts and boxes that you can use to split your project into parts. Below, we have split the screen into 3 (suggested) main parts. **You can split each of these 3 main parts into more parts. Instead of just drawing boxes, split the large boxes to smaller ones. Your aim is to have every element of the design in a box by itself.**

(light theme broken down on the left, dark theme broken down on the right)

The three main parts are 1) the navigation bar, 2) the profile picture card + audio feature (*note: these are two parts that are grouped with each other!)*, and 3) the bottom buttons bar.

As a quick side-note, the navigation bar contains the app logo and the two adjacent buttons. We think that you should create this component from scratch using a View, text, and two Buttons/Images. When you implement navigation solutions you will find that the navigation bar is a given and comes naturally. For the time being, it's okay to mock up what you see above. In practice this gives you flexibility to style your application uniquely.

2. For each of the three parts, figure out the components you want to use

Once you have figured out the different elements of the design, and roughly how your layout is going to look like, try to figure out the different components you will be using. This is helpful in starting the coding process.

For reference, the most important components for this project are the basic components: View, Image, Text, Button — though you are **NOT** limited to using only these.

3. Write the render function piece by piece

Choose one of the 3 main parts that we highlighted. Start by creating the parent *View* for this part, then style it, and recurse on its children. Keep in mind that along the way of styling/creating children, you may need to change the style of the parent. This is totally expected since you are relying on flexboxes.

4. Create Components:

Once you are done rendering any of the three main parts (navigation bar, the profile picture card + audio feature, or the bottom buttons bar) turn that part into its own custom component. This will let you organize code by moving relevant pieces into a specific file. See the lecture from week 3 for an example.

5. Autograder portion:

This is extra credit and is meant to grade your coding style. Grading the UI visually is one thing but making sure you're making your code as *modular* (i.e., you can reuse your components) and *abstract* (coding the building blocks for you app) as possible is another thing. In this portion, we grade what we believe is the best solution one can have for this assignment codewise. We encourage you to look at the test suite that's shipped with the starter code as it should inform you about the overall structure we're looking for.
To run the tests, you can use the following command `yarn run test`

Resources

Lastly, here's a link to the GitHub repository with the starter files https://github.com/acui51/cs47-a2-starter-w23. To begin coding: go to the GitHub repository, download the repository (click the green "Code" button), open terminal and `cd` into your new repository, and run `yarn install.`

You will be writing your components inside the app/components folder!

Then, you can open the project using your code editor, poke around the starter files, and begin coding! If you need help getting the assignment or submitting it, follow the GitHub guide or use Ed to reach out for help.

Most of your edits will happen in `App.js`, but make sure to check out the assets folder as well. Check the imports that are used at the top of `App.js` if you need a reference for how to use assets!

Remember to run `yarn install` before running `npx expo start`.
More resources on styling in RN: Styling cheat sheet

Submission

To submit, you will need to update your Assignment 2 GitHub repository by **Friday, February 3rd, at 11:59pm**. Please see the GitHub handout for more information.
Also, keep an eye out for the pinned Ed post for A2, we will be posting relevant updates there!

Tips:

-I am getting some error about an SDK error?

Run this command: `npx expo client:install:ios`

-You ~~can~~ should import the colors, icons and profile pictures (yes, we provided you with more than just MTL) by importing the respective objects. Here's an example of how you can do that with icons
`import { Icons } from "./assets/Themes"`
To know how these objects work and what you can import/do with them, we recommend either printing them out and seeing what they have (using the console.log() function) and/or by going to the respective files (e.g., the Icons object in ./assets/Icons/index.js)
-What should I name the files that contain my components?
We recommend:

→ app/components/
|
|-> header.js
|-> body.js
|-> footer.js (we did this for you!)

A note about the autograder: it actually looks for the components named the same as the files rather than the files themselves.
Hint: look into the index.js bit of the export/import doc.

-SafeAreaView might be helpful! ;)

-How do I set an image as a background?

React Native offers a very helpful element for this use case, called ImageBackground