Brian Bauman

CSC 471

**Final Project Documentation**

*Project title:*

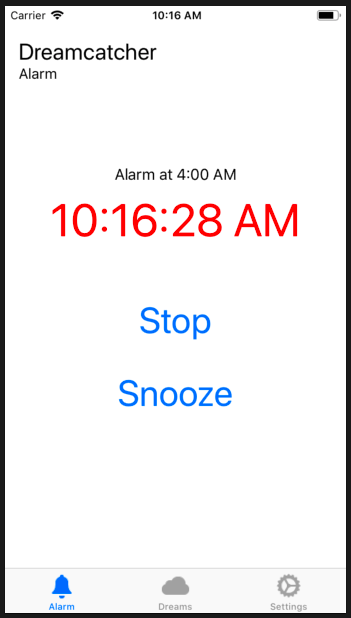
Dreamcatcher

*Project description:*

This app enables easy access to recording and reviewing dreams you’ve had, replacing the need for a dream journal or diary.

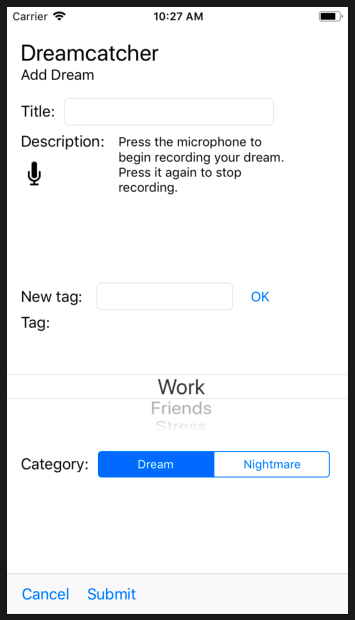
*Project documentation:*

(1) Alarm screen:

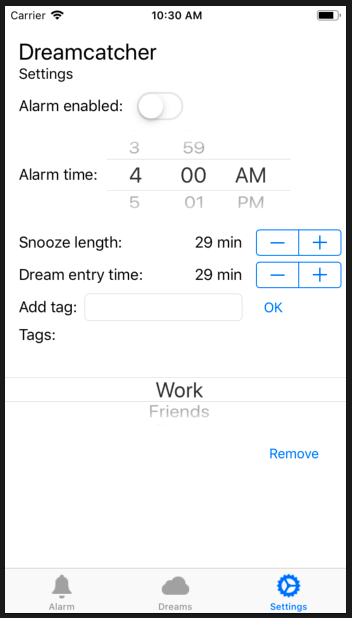
The app opens on the alarm screen, shown below. This screen displays the current time as well as the time of the currently set alarm (“Alarm disabled” is instead displayed if the alarm is disabled). The “Stop” and “Snooze” buttons are disabled until the alarm is triggered. When the alarm is triggered, the text begins to flash and a loud, obnoxious noise plays. Pressing the “Snooze” button resets the alarm time to a point in the future (configured on the Settings page). Hitting the “Stop” button takes you to the Dream Entry page.

(1) Dream Entry screen:

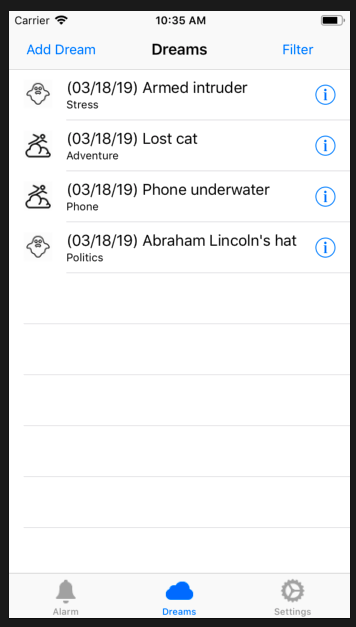
The Dream Entry screen (below) can be accessed from (a) the “Stop” button on the Alarm screen or (2) the “Add Dream” button on the Dream List screen. This screen serves to allow the user to record a dream. The options are straight-forward — the user enters a title, description, tag, and category of the dream. The title is entered as a text field. The dream is entered using voice-to-text. First, the user clicks on the microphone icon, then begins to speak. The words they say are transcribed and appear as they are spoken in the text view to the right. When the user is finished speaking, they click the “recording” icon to stop the recording. The dream’s tag is chosen using a picker. The user may also choose to enter a new tag here if an existing tag is not sufficient. If a new tag is entered, it will appear in the tag picker and must be chosen there. Lastly, there is a segmented control that allows the user to categorize the dream as a “dream” or a “nightmare”. When finished, the user hits “Submit” to record the dream. If any fields are not completed correctly, an alert will display informing them of what needs to be changed. If the “Cancel” button is selected, the dream is not recorded.

If the Dream Entry screen is accessed from the Alarm screen, a timer will require the user to complete the dream entry in a certain number of minutes (configured on the Settings screen). If the timer expires, the user is returned to the Alarm screen. This is to prevent the user from falling asleep while entering their dream.

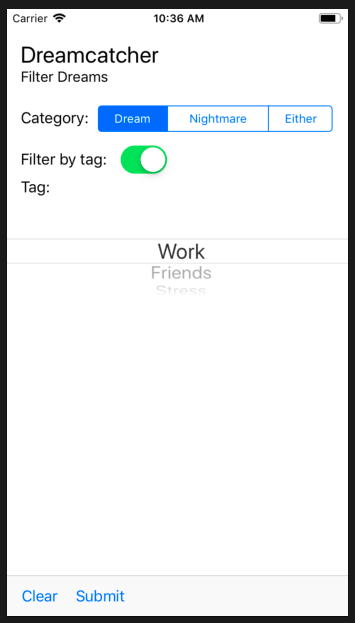
(3) Settings screen

The Settings screen is accessed from the tab controller at the bottom of the app. This screen allows the user to configure the Dreamcatcher app. On this screen, the user can enable or disable the alarm, configure the time at which the alarm will fire, set snooze and dream entry times, and add or remove dream tags. All of the buttons remain functional regardless of whether the alarm is enabled. The maximum time allowed for snoozing or dream entry is 30 minutes.

(4) Dream List screen

The Dream List screen displays all recorded dreams in a table view, sorted by the date they were entered. The cell lists the dream’s date, title, and tag. Clicking on the detail accessory button brings up an action sheet containing the description of the dream. The image to the left of the dream indicates whether the dream is a “dream” (a man flying through the clouds) or a “nightmare” (a spooky ghost). At the top, a user also has the option to navigate to the Dream Entry screen (via the “Add Dream” button) or the Filter Dreams screen (via the “Filter” button).

(5) Filter Dreams screen

The Filter Dreams screen allows you to filter the items in the Dream List screen and is accessed from the “Filter” button on the Dream List screen. The user has the ability to filter on the dream category (“dream”, “nightmare”, or both) and the dream tag. If the user doesn’t want to specify a dream tag, they can turn the “Filter by tag” switch off. At the bottom, the user may hit “Submit”, which will filter the dream list and return them to the Dream List screen, or “Clear”, which will clear all filters and return to the Dream List screen.

**Final Project Discussion**

*Features used:*

In this project, I used a number of features discussed in class and a handful of features that were not covered.

The in-class features that were used include:

1. Tab Bar Controller

The entire app is controlled via a tab bar controller that separates the three main screens (Alarm, Dream List, and Settings). This allows the user to quickly switch between the three most commonly used areas in the app. The tabs are labelled with descriptive images that enable more convenient navigation.

(2) Navigation Controller/Table View Controller

The Dream List screen is a table view that presents each recorded dream in its own cell. This is a dynamic table view that can handle as many dreams as the user wishes to enter. There are two prototype cells — one for dreams and one for nightmares. Each provides slight customization to the aesthetics of the cell. Both prototypes are styled as “Subtitle” cells with an accessory detail button on the right that allows more information about the dream to be accessed. Buttons at the top left and top right of this screen provide navigation to other screens, from which the user can return to an updated table view controller.

(3) Segues

There are many segues in this project — almost on every screen. These are pretty standard segues and most do not pass data and are simply dismissed to return to the previous screen. Some pages, such as the Alarm screen, do pass data forwards (a dream entry timer, in this case).

(4) Assorted user input and display

The application uses a large assortment of user input features covered in class, such as text fields, switches, buttons, pickers, steppers, and segmented controls. These all allow the user the ability to customize the functionality of the app and to input structured data when recording their dreams. I won’t spend too much time discussing the implementation of these, as they are largely self-explanatory. In addition, there are many labels and images used to convey information to the user, such as the current time and information about recorded dreams.

Features not discussed in class include:

1. Voice-to-text

The process of entering your dream’s description is entirely controlled with your voice. On the Dream Entry screen, a microphone image can be clicked (resulting in a recording image replacing it). Afterwards, the app listens and records the words you say, displaying them as you speak. When finished, hitting the record button ends the recording. This feature was a bit tricky to implement, but the internet provided a lot of help. I had to customize some code I found online, which all used the Speech framework. Also, the use of the microphone requires the user’s permission the first time they hit the microphone button.

(2) Playing sounds

The alarm itself seemed pretty half-baked if flashing text were its only feature. As such, the Alarm screen plays an obnoxious alarm noise when the alarm goes off. This feature required the use of the AVFoundation framework, which was much more straightforward to implement than the Speech framework. By including an MP3 file of the alarm noise, an AVAudioPlayer handles the playing/stopping of the alarm sound.

*Biggest challenges:*

My biggest challenges in this project largely had to do with the features not discussed in class. Both required some research and a lot of debugging, as there were no in-class examples on which to base the design. Luckily, sites like StackOverflow provided a lot of assistance, as nothing I was doing was truly brand-new. The ability to quickly test the design on the Simulator and on my actual device was also very helpful. The other big challenge that I ran into was the layout and formatting. I feel like I spent a lot of time making sure that the app displayed properly on devices large and small. I wonder if there are better ways to approach this problem, as my strategy required a lot of starting over. I am certainly glad I waited until the very end to bother with the layout at all.

*Limitations of the app:*

I think this app is very, very close to being a usable product. As I see it, there are two main limitations. The first is that the alarm will not sound unless the app is already open. The ability to have your app wake itself up from the background seemed tricky and was not discussed in class, so I didn’t even attempt to implement that. In addition, I’m not 100% sure it’s possible as described in my proposal. The second limitation is the inability to edit or delete existing dreams. Though this might be a more rare use case, I can definitely imagine wanting to go back and add detail to a dream that was already recorded. This feature would definitely not be hard to implement, but would require some thought behind is user interface.

*Limitations of the SDK:*

I didn’t find too many limitations of the development environment. In all honesty, it has become one of my favorites and is very intuitive. The inability to wake an app that is in the background (or potentially not even open — as described above) could be a limitation of the iOS environment that makes this app less desirable, certainly. Also, the layout features seemed to list a ton of warnings that were sometimes tough to remove without sacrificing the desired appearance. I truly think that this isn’t an issue with the SDK, but rather my own skills. With more practice and exposure, I’m sure that this becomes more of second nature.

*Overall experience:*

Overall, I very much enjoyed getting to learn how to develop for iOS. I had never made a phone app before and, in an age where everything is an app, taking this class was a great excuse to learn. I thought that the UI of the IDE was very comfortable. I really liked how intuitive the visual design aspects were, with drag-and-drop functionality that you normally see in an application like Photoshop. As mentioned above, having to design for many screen sizes was a bit tough, and I can only imagine how much more troublesome it would get when you consider iPads, different screen orientations, etc. However, I’m sure that it becomes similarly fluid once you learn the ropes. I was pleasantly surprised with the many things we were able to accomplish in only 10 weeks! What a fun class.