Genevieve Myers

Prof. Xiaoping

CSC371

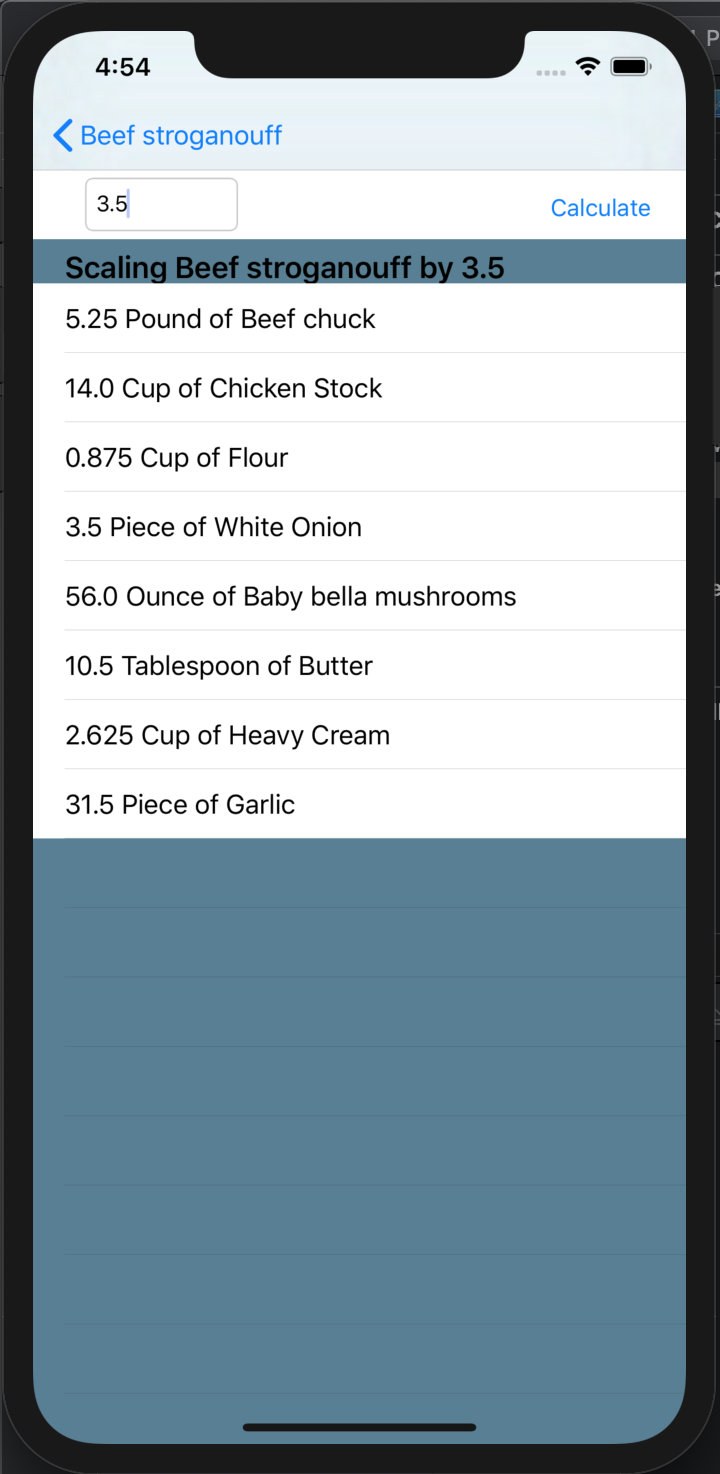
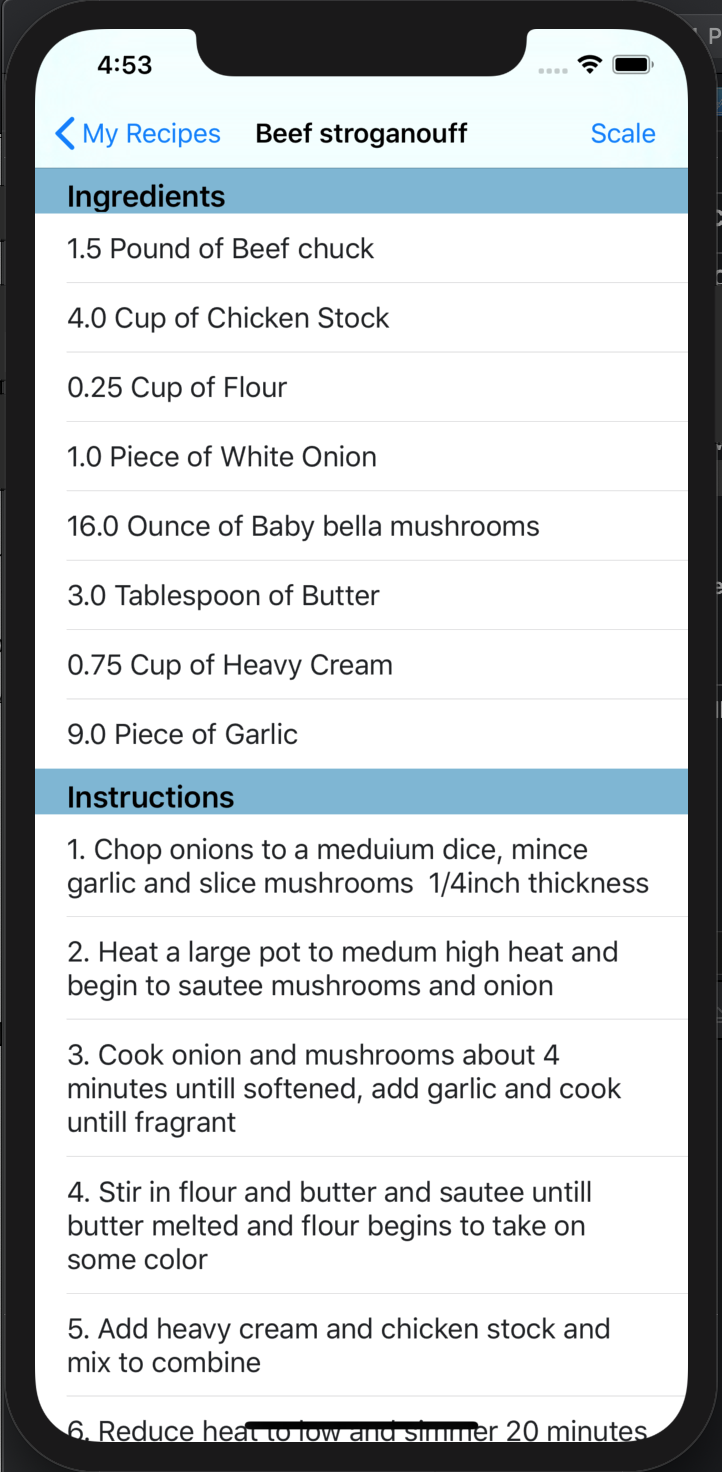
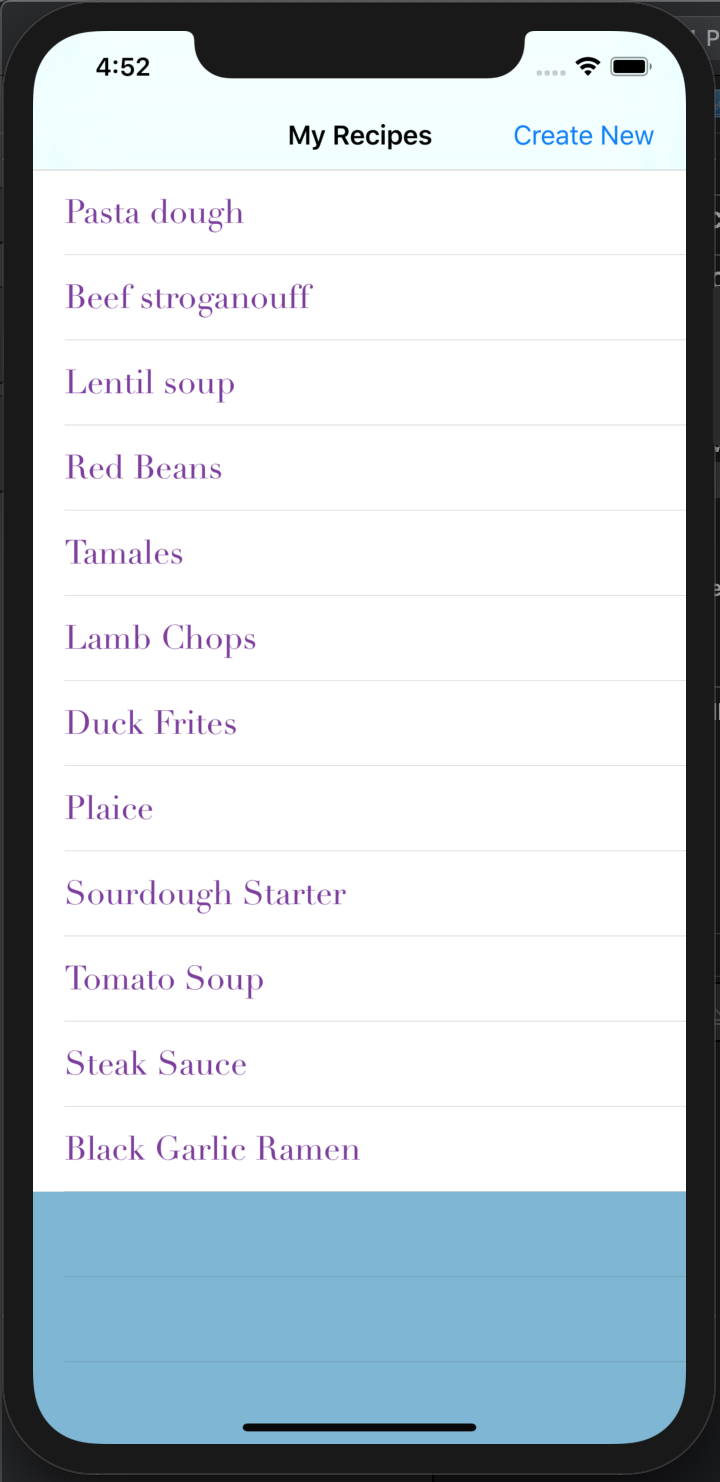
18 March 2020

Final Project Documentation

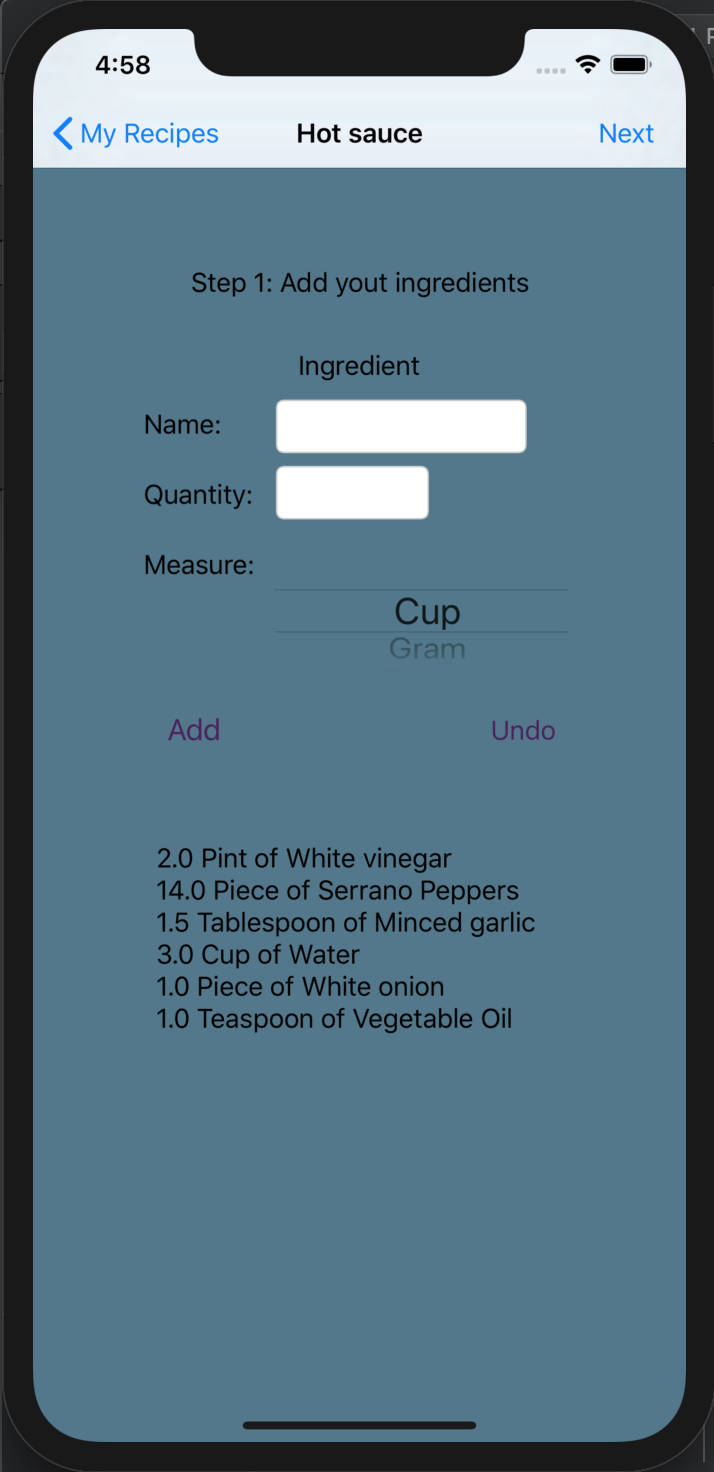
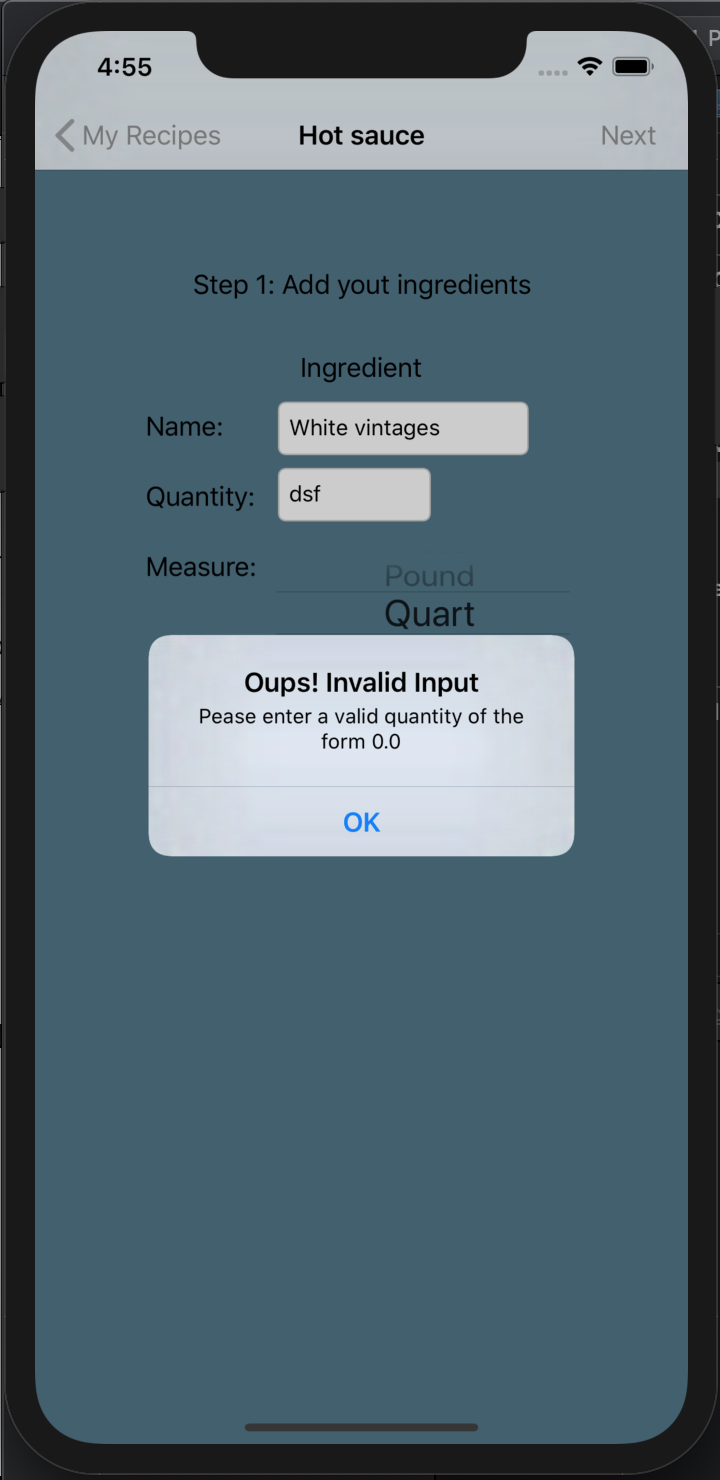
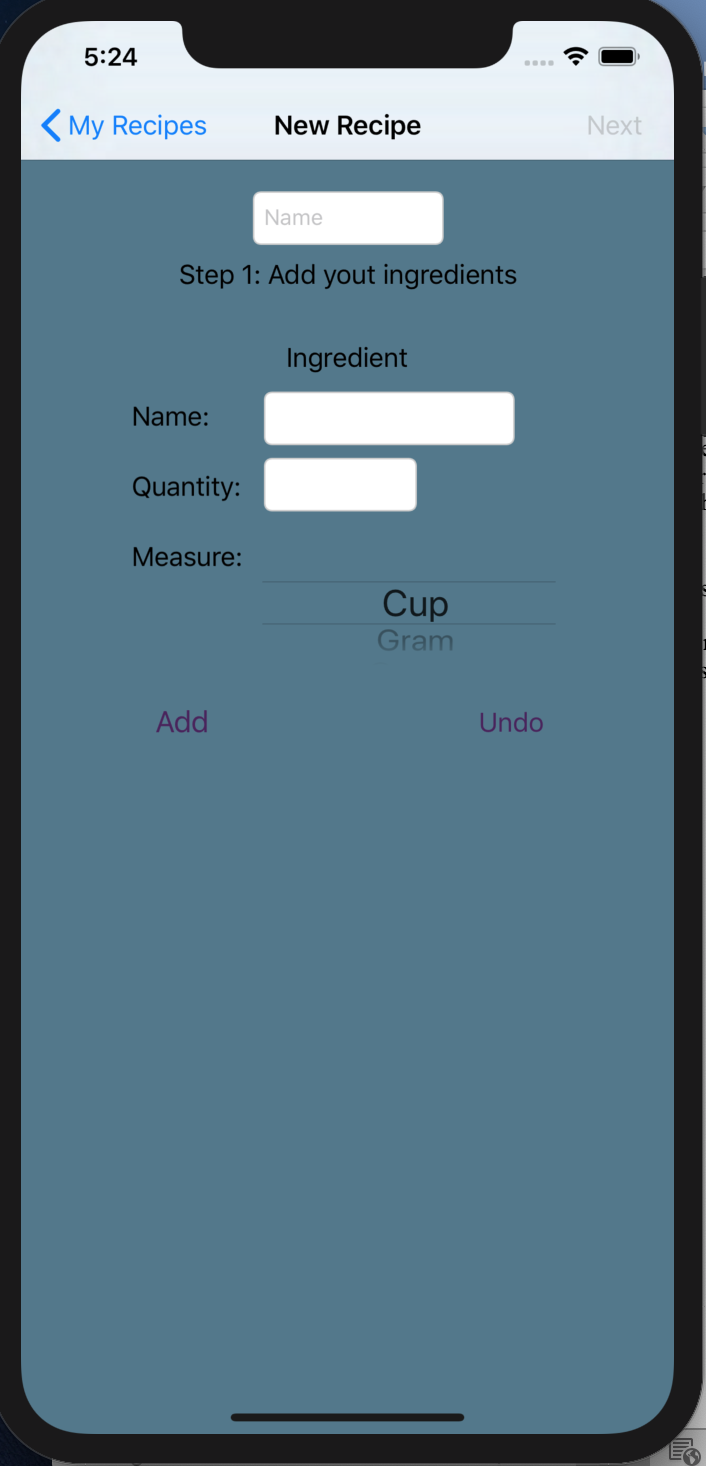
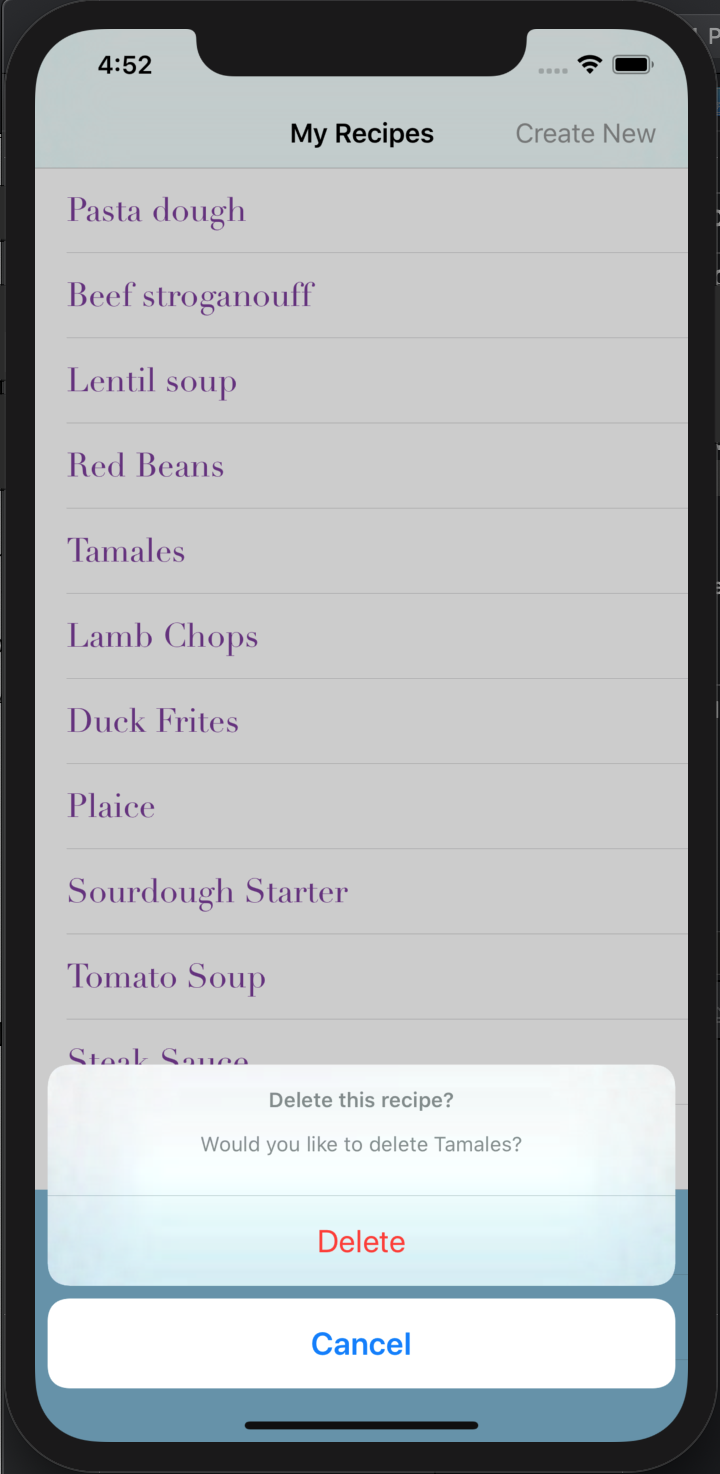
1. Final Project Description and Documentation
   * This app is intended to be a recipe app for users to store and view their own recipes. I have created some dummy data for our purposes and demonstration.
   * Users can view a list of their saved recipes delete unwanted recipes and create new recipes. When viewing a recipe users also have the ability to scale the ingredients of their recipe for bigger or smaller batches.

All Features:

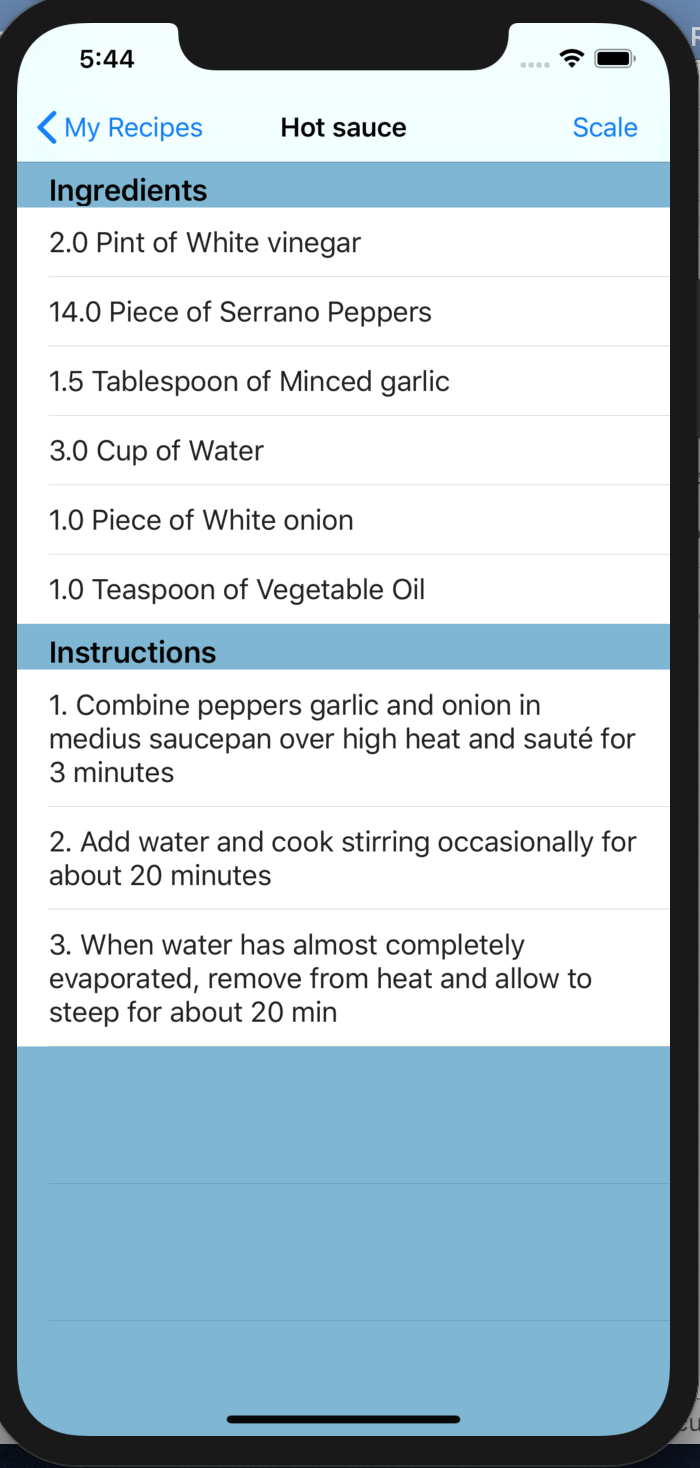
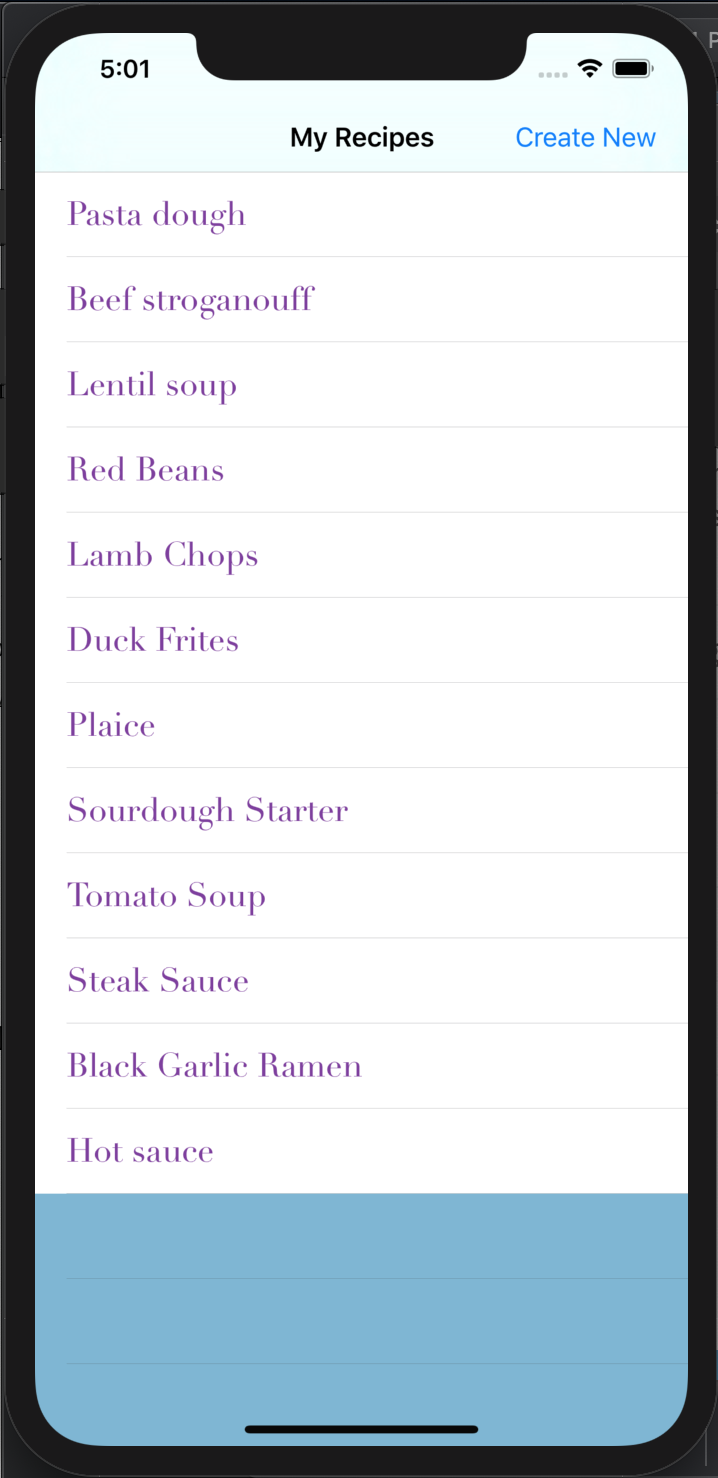
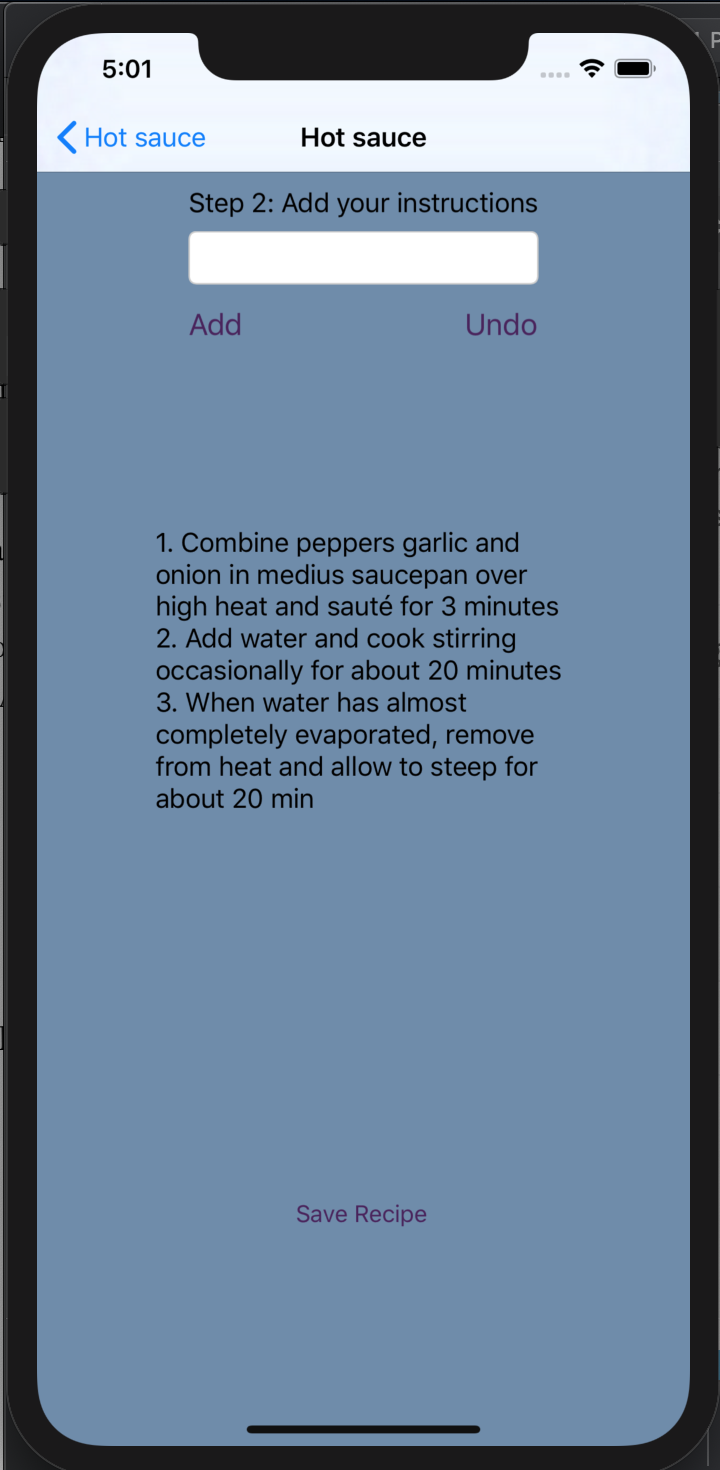
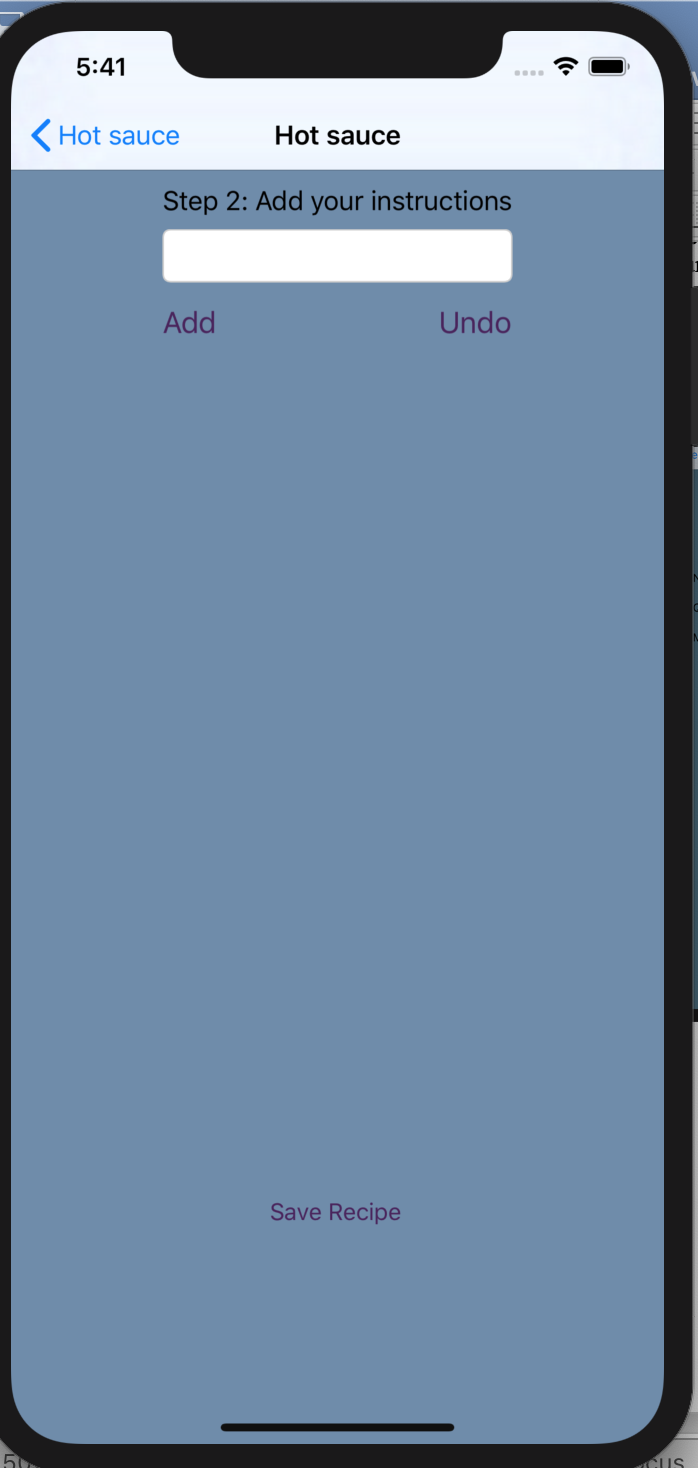
* When user opens app they are shown a table of all of their saved recipes, from here they have a few options depending what they would like to do.
* They may click on a recipe which brings them to the details view of this recipe, which displays the ingredients followed by the instructions for this recipe
  + If they would like to scale this recipe they may use the navigation bar “Scale” button which brings them to the Scaling view
  + In this view users may enter any Double value and hit “Calculate” to see the ingredients at this scale. If users enter an invalid scale they will be notified of their error and may try again, they may scale the recipe as many times as they desire
  + Form here they can navigate back to the recipe view and back to the many Recipes view page



* + If a user has created a recipe the no longer desire to save they have the ability to delete it. This is done from the home page by swiping left on the recipe which brings up a command sheet verifying the users choice. They may select “Delete which removes the recipe from the list and all records of it or “Cancel” to cancel the action
  + From this page users also can create new recipes by selecting “Create New” from the navigation bar. This will bring them the first of a to a two-step process for saving new recipes.
    - The first view allows users to enter a name for their recipe and input all of their ingredients. Each ingredient is required to have a name, a valid quantity and a selected form of measure, users are notified if their input is invalid and asked to try again. If an ingredient is added by mistake they can select “Undo” to remove their last addition. Once the recipe has a name and at least one ingredient the user can select “Next” to move to the final step entering instructions.



* This simple view allows users to add instructions for their recipe in order and displays their entries on the screen below. Once again, they have the ability to undo the last addition. Users are not however required to provide instructions if they choose not to, I decided to leave this up to user’s discretion as it is intended to be an app for personal use and some may not need instructions for their own recipes.
* Once the user is satisfied with their recipe they can hit “Save Recipe” which will return them to the home page with their new addition appearing at the bottom of their list of recipes where it can be viewed, scaled or deleted like any other!



1. Final Project Discussion
   * I implemented many of the API features we discussed in lecture to get this project up and running as I had envisioned. Although I didn’t use many features not discussed in lecture, I had to expand on what we learned in many ways to get the full functionality I was hoping for. My app is centered around a single navigation controller, with segues to several table views or view controllers depending on the user’s action.
   * Some of the biggest challenges I faced when creating this app was initially familiarizing myself the navigation view controller and table views and how to set them up properly and display my data. Even though we discussed them in lecture and went through examples, it took me a significant amount of time to get the table view for the recipe details working correctly. Initially tried to segue to a view controller with a two table views, one for ingredients one for instructions. I even created custom table view cells and classes for each however I was not ever able to get the data to display. I was hoping to implement a screen that didn’t appear to be so much of a table but still had lists of items. However, with all my efforts proving futile and with some further research it became clear the best option for me would just to create a second table view with two sections to display a recipe. Creating a table view with multiple sections is also somewhere I struggled a bit as we didn’t go this far into table views in class. I had a hard time figuring out how to access and display information in the correct place and how to differentiate the contents of a cell in one section from another.
   * In the end, being forced to really comprehend how table views work and what exactly is happening in these various methods was for the best. The table view displays the information in a clear and efficient manner and is probably sleeker than what I would have created in a view controller for a recipe. All this struggle also made it easy to implement the Scaling functionality which I had really hoped to include but wasn’t sure I would be able to implement in our time frame. It was yet again another table view, however this this time I was able to implement it in a fraction of the time and had a much easier time debugging.
   * One challenge that did arise in creating this third linked table view however was creating a navigation bar button and segue not from a cell in a table view. For some reason, in my storyboard I couldn’t access my navigation bar in my table view, I could create a button on a toolbar to segue to scaling but this bulked up the screen and didn’t really make sense when I already had a navigation bar with a perfect empty right button spot. To access the navigation bar here, I had to create the button in my Swift file and link in manually through a generic segue I created in my storyboard, it wasn’t that difficult but took some time and research to flush out.
   * I found the navigation bar to be a little bit of an annoyance in other areas as well, I was not able to easily link actions other than segues to it. This was not clear as Swift allowed me to create actions and link navigation items as outlets. However, it took some debugging to realize they were simply not being executed as I intended. I’m sure if I put in more time familiarizing myself with the navigation control bar I would have come to appreciate it more as with the table view, however I generally found its functionality to be frustrating and not intuitive. As with the button item in my table view, why would it not let me simply add it on my storyboard as in the view controller. Instead requiring it to be implemented in a more difficult and time consuming manner to a beginner. And here allowing me to code functionality and not give errors or warnings that simply would be bypassed as is not the correct way of handling navigation bar functionality.
   * This being said I think overall iOS SDK and Xcode is generally a relatively easy framework to adjust to and grasp the basics of quickly. Once you are trying to achieve more complicated tasks and have specifications not directly provided in their API features, tweaking and learning to use these features like you want becomes more of a challenge. This was not made any easier by the rapid pace of new releases and versions of Swift and Xcode with serious changes. I often would look to the internet when things were not behaving as I expected or couldn’t figure out how to correctly implement a feature and rarely would there be useful information pertinent to Xcode11 or Swift 4. Usually I would turn to developer documentation, but I also found Apple documentation to be painfully sparse. I do not mean this as they did not have documentation on the methods or classes I was searching, but this documentation it included little detail and often relied heavily on defining one thing by its related attributes and methods which can seem circular and not provide much insight to a beginner. This trouble finding adequate documentation or examples to help explain why certain things behaved the way they did or how to achieve a goal was common place across the final project and my programming assignments.
   * That being said, this was my first experience in app development and given that I feel I was able to accomplish a relatively high functioning app having only 10 weeks experience. Despite what I see as its shortcomings, the Swift and Xcode framework defiantly provides you much of the backbone to get up and running as a developer in a no time. Getting to know and manage the detailed inner workings of these provided features however proves to be the real challenge, as they handle and manage so much of it for you behind the scenes.
   * I am satisfied with the app I was able to develop using this framework, there was not much functionality I hoped to implement that I was not able to sort out eventually. One definite limitation that I would have flushed out if I had more time would be to give users the ability to edit an existing recipe. Now if they want to make a change they would have to simply delete and start anew. This also speaks to another area I think my app could use improvement, in the underlying data structures and way I choose to store my recipes. My specifically my ingredient storage as an array of tuples makes it somewhat difficult to access and change specific values. I created the class and some dummy data and figured out displaying my recipes and recipe detail view before figuring out how to take user input which in hindsight was not the best idea. It made the user input form for new recipes somewhat clunky editing a recipe even more challenging. Given more time I would have done some refactoring to come up with a more elegant recipe class which would make implementing more functionality far simpler. Nonetheless I am proud of all that I was able to accomplish and despite my somewhat clunky user input feel this is an app I would defiantly use.