You have free reign to create an application that manages some type of mutable information. That means that the data displayed by the app is expected to change. The data you load and display can come from anywhere and you can do whatever you want with it. For instance, you might display images from online somewhere, you might display stock exchange information, or you might display a information from twitter--or movies, or books, or amazon. It can be whatever information you want to display (really, have fun with it), as long as you use the interface elements as described below. You will need to get creative in order to incorporate ALL the design elements below. That is okay, some parts of the app will just be weird (but not all the app).

Create an iOS application in XCode that uses the storyboard and a TableViewController to load different views based on the data your are loading. The TableViewController must implement three different types of cells and load them dynamically (i.e., you cannot use a static table). View navigation can be hierarchical in any way you want--as long as it makes sense for the interface and the data you are displaying. When loading a new view controller your main view controller should hand off information to the controller that is getting created. The application should make use of the following design elements in one or more of the views:

Required for 5000 and 7000 Students:

- Automatic Layout
- Buttons, Sliders, and Labels
- Stepper and Switch
- Picker (you must implement picker delegate)
- Segmented Control
- Timer (which should repeat and somehow update the UIView)
- ScrollView (with scrollable, zoomable content)
- Image View
- Navigation Controller
- Collection View Controller
- Table View Controller with dynamic prototype cells
- The design should work in both portrait and landscape mode
- I should not be able to crash your app
- Your design must strictly adhere to Model View Controller programming practices
- Use lazy instantiation when possible

Exceptional Credit for 5000 students and Required for 7000 students:

- Implement a modal view and handle properly using custom protocols/delegation

Test your app running on the device, not the emulator to ensure it runs in all scenarios. Also see the grading rubric for how much each element is worth.

Turn in the source code for your app via upload or github link (if uploading, call the file "teamNameAssignmentOne.zip"). Use proper coding techniques and naming conventions for objective C and/or swift. Use whichever programming language you are most comfortable with. Include your team member names and team name in the comments of the canvas upload text.