Project 3

By: Eric Dockery

1. ***Bronze challenge for the end of chapter 4 page 106:***

The Problem

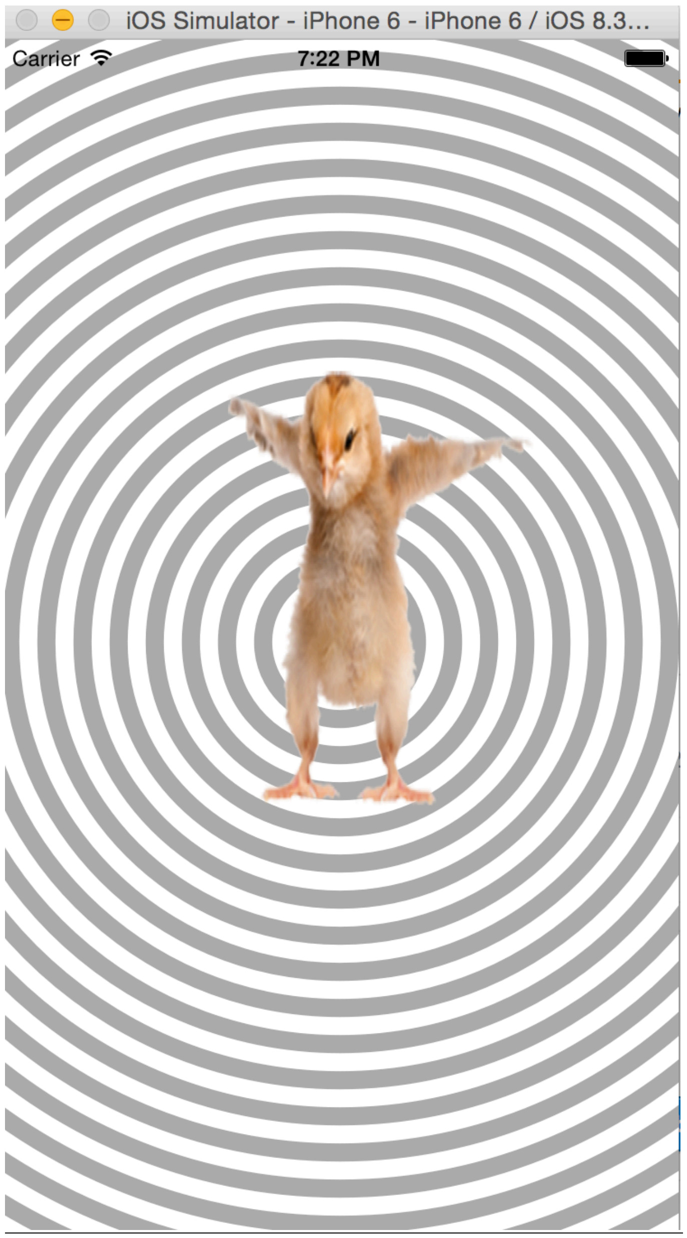
The challenge is to load an image from the filesystem and draw it on top of the concentric circles, such as the example given in the book on figure 4.21. Find a PNG image file with some transparent parts that would be especially interesting and drag it into your Xcode project. Creating a UIImage object from that file is one line:
UIImage *logoImage = [[UIImage imageNamed:@"logo.png"]];
In your drawRect: method, compositing it onto your view is just one more:
[logoImage drawInRect:someRect];

The Solution

The Question really gives the answer for this problem. For my solution I just went online and found a image of a baby chicken. Then to add the baby chicken to the view I created a rectangle for the image to sit inside. This was accomplished by adding these two lines of code after including the image in my project target locations.

```
UIImage *logoImage = [UIImage imageNamed:@"chickens.png"];
[logoImage drawInRect:CGRectMake(100, 134, 200, 300)];
```

Screenshot



2. ***Bronze and Silver challenges at the end of chapter 6 page 138:***

Bronze challenge problem
Give the UITabBarController a third tab that presents a quiz to the user. (Hint: you can reuse files from your Quiz project for this challenge.)
Bronze challenge solution
This challenge should have been simple but due to some refrencing issues from my quiz app (project 1), there were issues on getting this challenge to work. I eventually got the solution by recreating the quiz application inside of the project without importing the files from project 1. To accomplish this

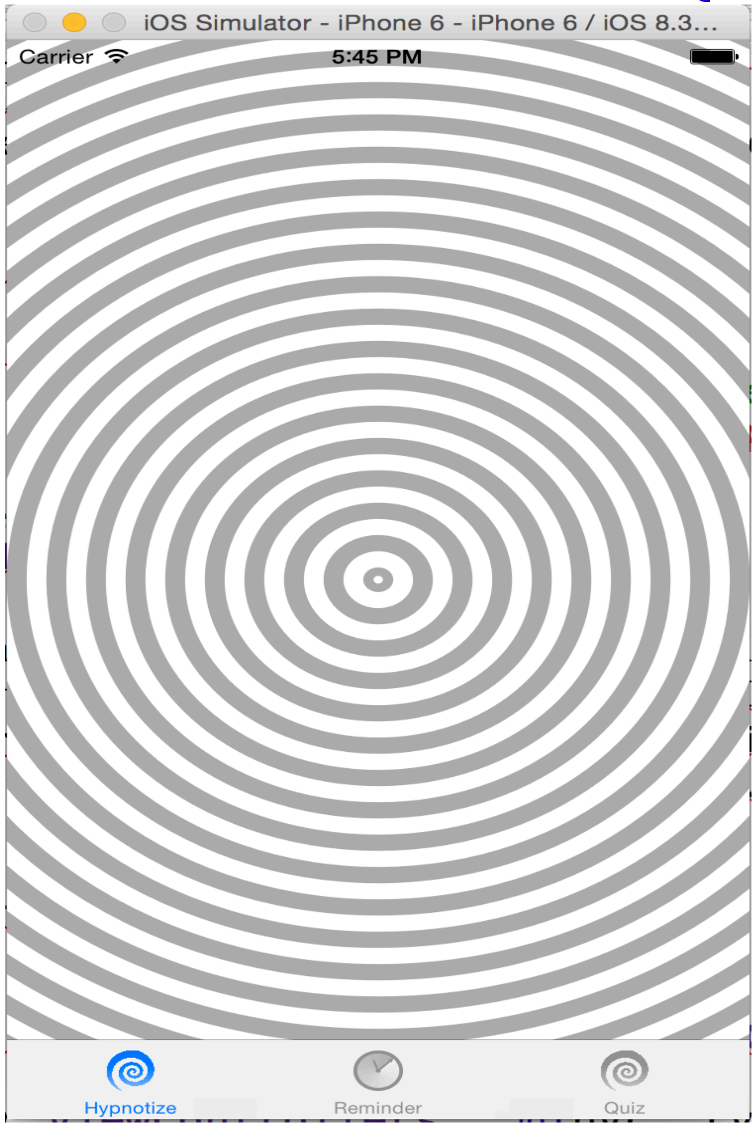
solution you needed to add the QuizViewController.h, QuizViewController.m, and the QuizViewController.xib files from project 1. Inside the QuizViewController.m file on the `if (self) {` line you add the tab information.

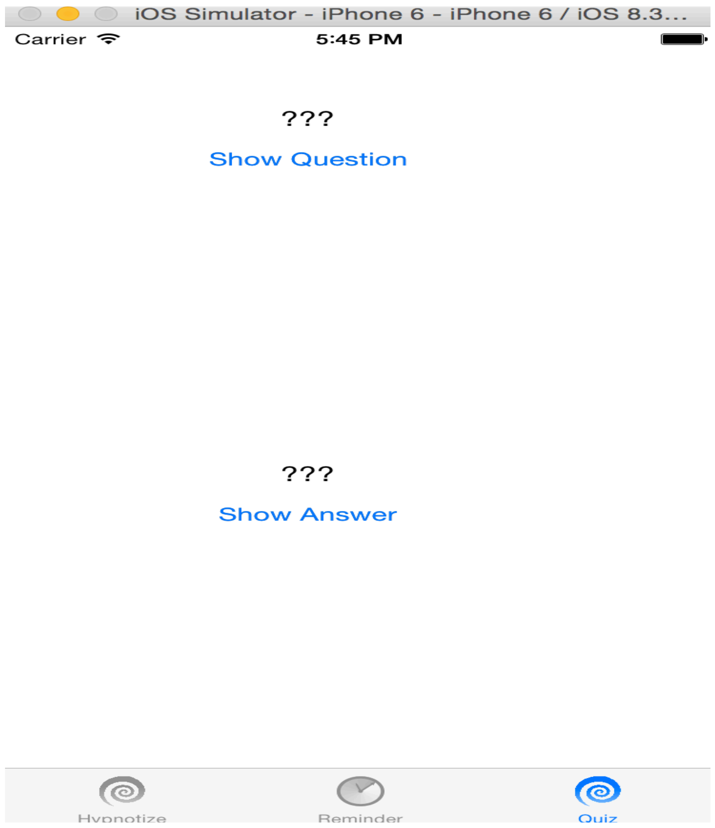
```
// Set the tab bar item's title
        self.tabBarItem.title = @"Quiz";

        // Create a UIImage from a file
        // This will use Hypno@2x on retina display devices
        UIImage *image = [UIImage imageNamed:@"Hypno.png"];

        // Put that image on the tab bar item
             self.tabBarItem.image = image;
```

I chose to just reuse the hypno.png image due to having trouble finding a image the correct size as the other png files. Then on the AppDelegate.m file you add a QuizViewController pointer object allocating and init the pointer. Then you add that pointer item to the UITabBarController.viewControllers array.

```
//quiz file
    BNRQuizViewController *qvc = [[QuizViewController alloc]
init];

    UITabBarController *tabBarController = [[UITabBarController
alloc] init];
```

```
tabBarController.viewControllers = @[hvc, rvc,qvc];
```





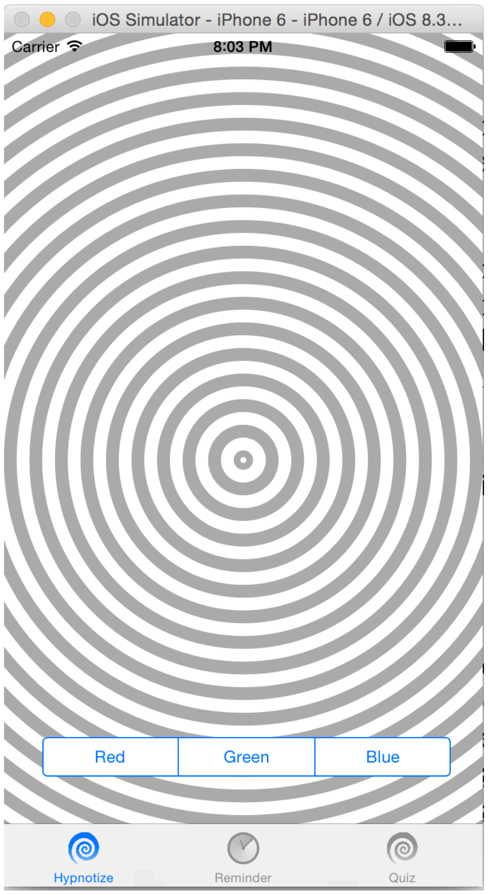
__Silver challenge problem__

Add a UISegmentedControl to HhypnosisViewController's view with segments for Red, Green, and Blue. When the user taps the segmented control, change the color of the circles in HypnosisView. Be sure to create a copy of the project and work from that copy while attempting this challenge.

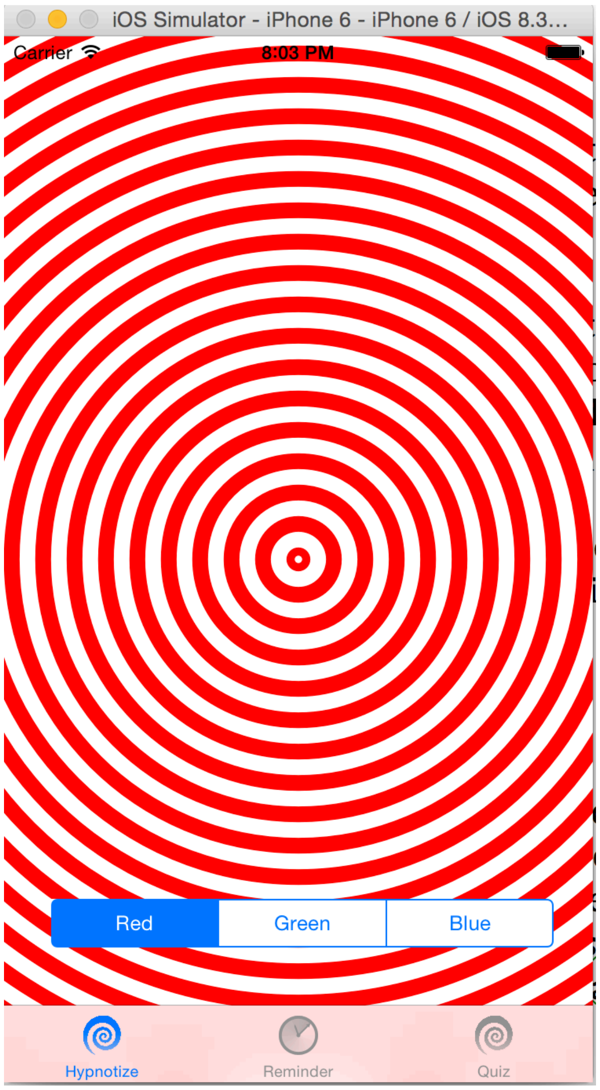
__Silver challenge solution__

This for me was a very difficult challenge. To solve this challenge I had to add an IBAction to change the color using an if else statement to change the color to red, green, or blue. The UISegmentedControl was easy to allocate and initialize using the native call as described in the challenge question and I set the array using @[@"red", @"green", @"Blue"]] . Then setting the segmentedconrol to the right position using the .frame command and the CGRectMake(30, 550, 320, 31); to try and get the most even placement in the lower portion of the screen. Adding a tagetpositionas self and an action to change the circle color. This challenge was very difficult for me personally trying to get the correct commands for the challenge.

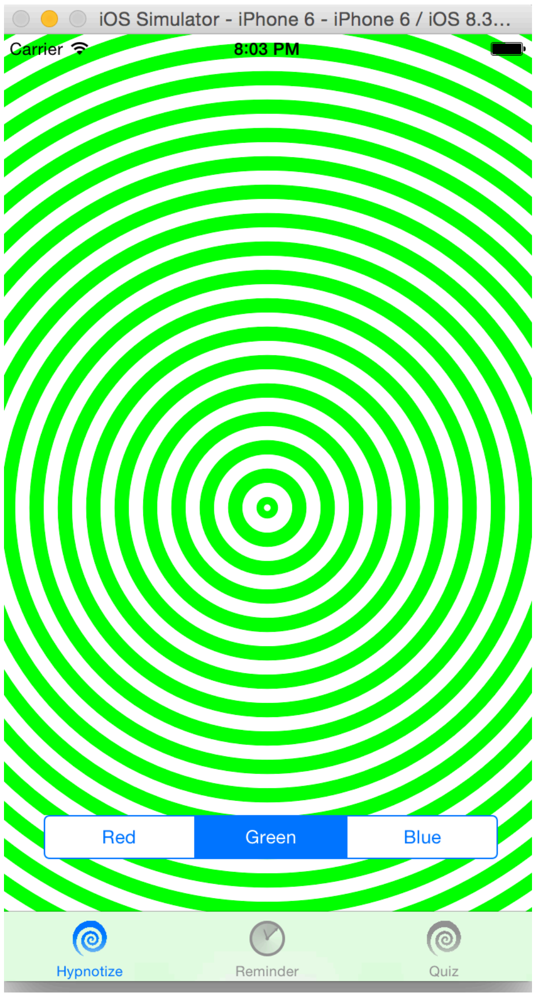
Normal Start



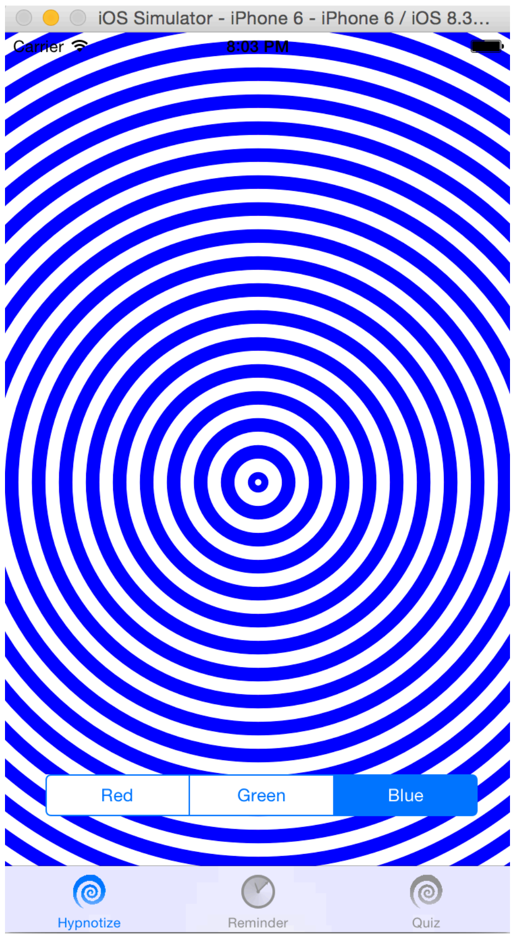
Red Selected



Green Selected

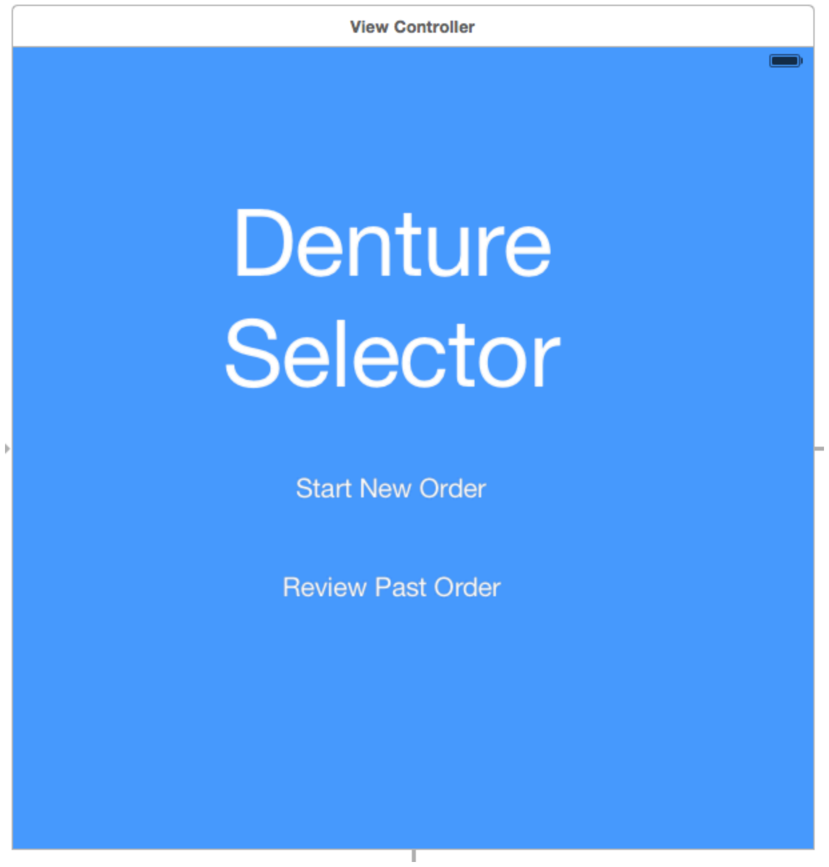


Blue Selected

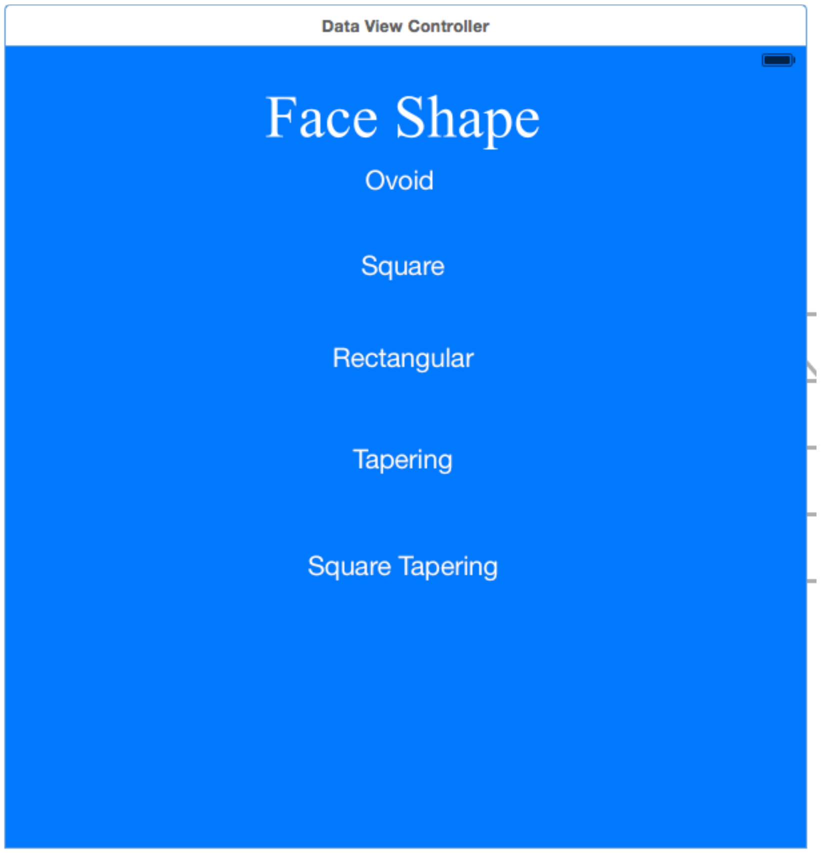


3. ***A brief description of my final project:***

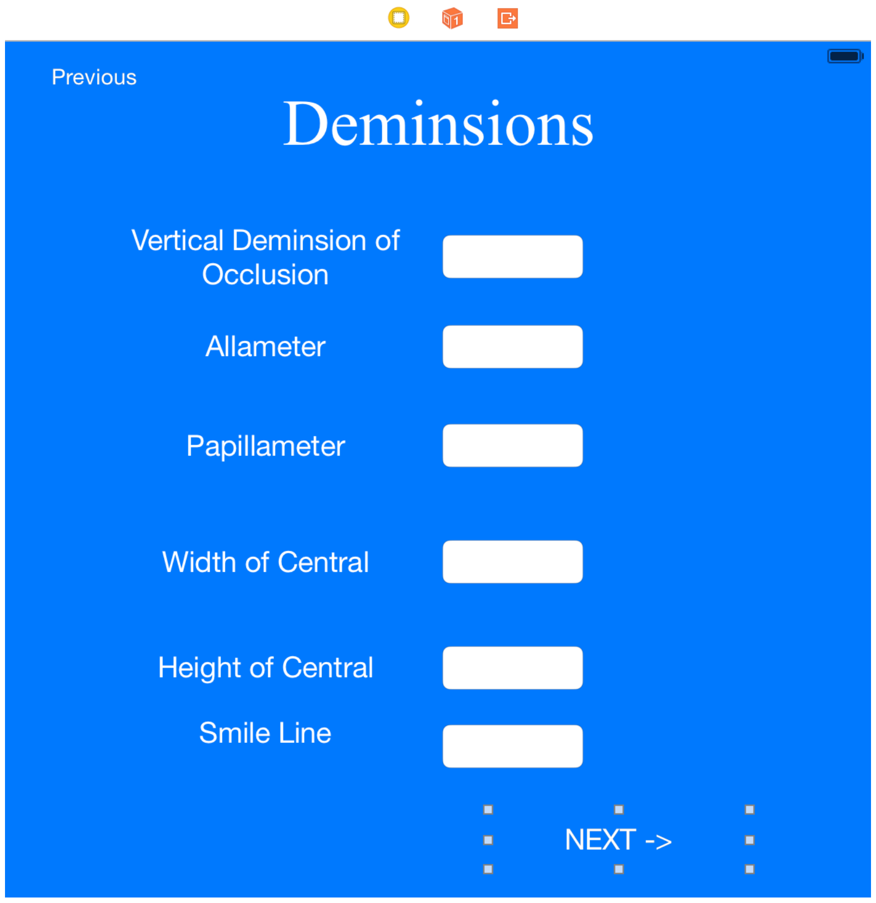
For my final project I would like to create a dental application that would allow a dentist to select the correct tooth order for a patient. This app would have multiple UIViews and layers. The first part of the app would ask the user if they want to create a new order or review their past orders. This will be two buttons that will change the view of the app to the next appropriate storyboard.



If the Start New Order is selected then the user will be shown the storyboard for selecting the face shape of the patient. They will have 5 different buttons to choose from Ovoid, Square, Rectangluar, Trinagluar, and Square Tappering buttons.



All of these buttons will go to the next storyboard but will change the face shape label on the deminsions storyboard to the face shape they select. On the deminsions storyboard the user will have to input values for vertiacal deminsion of occlusion, allameter, papillameter, width of central, height of central, and smile line. Note there will only be two mandatory values filled out:vertical deminsion of occlusion and smile line. The other data will help choose the best options for the teeth selection program but those two are mandatory to calucate the minimum requirements that can be given for a selection.



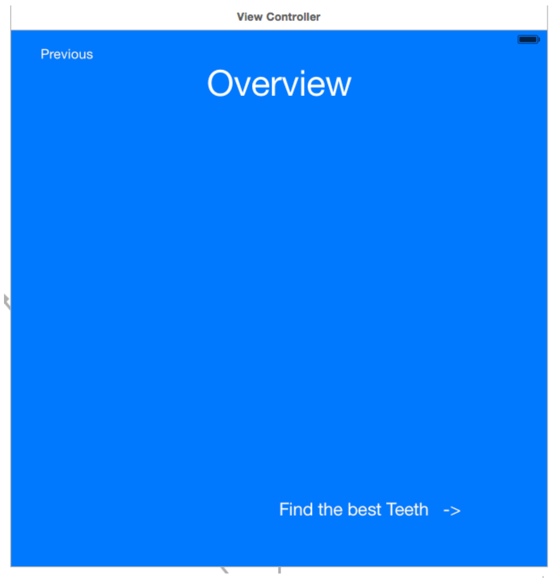
Previous

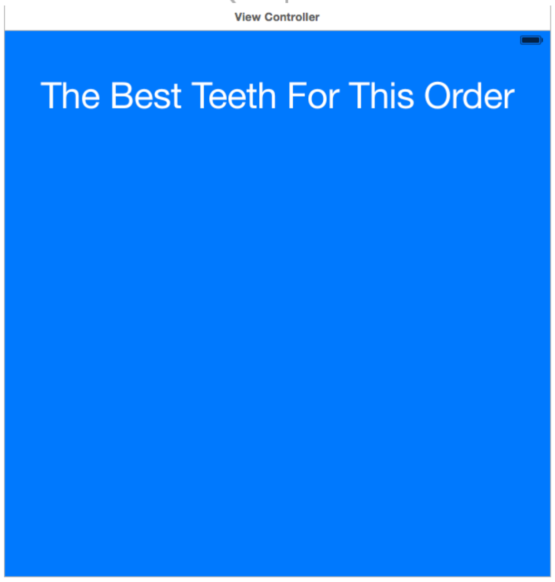
Deminsions

Vertical Deminsion of Occlusion	
Allameter	
Papillameter	
Width of Central	
Height of Central	
Smile Line	

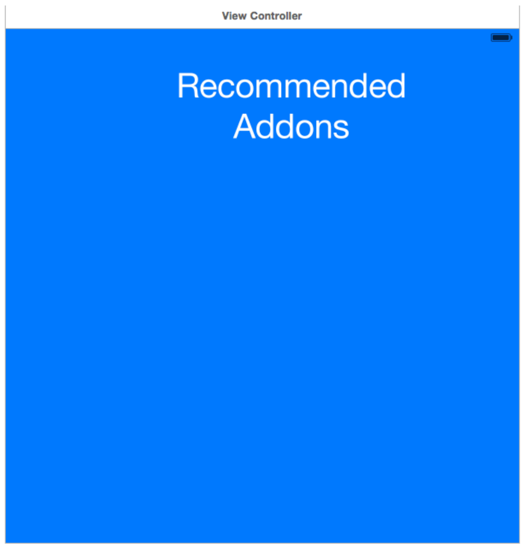
NEXT ->

When they click the next button at the bottom of the page their will be a background function that will calculate the best 5 Upper Anteriors that fit the given deminsions.

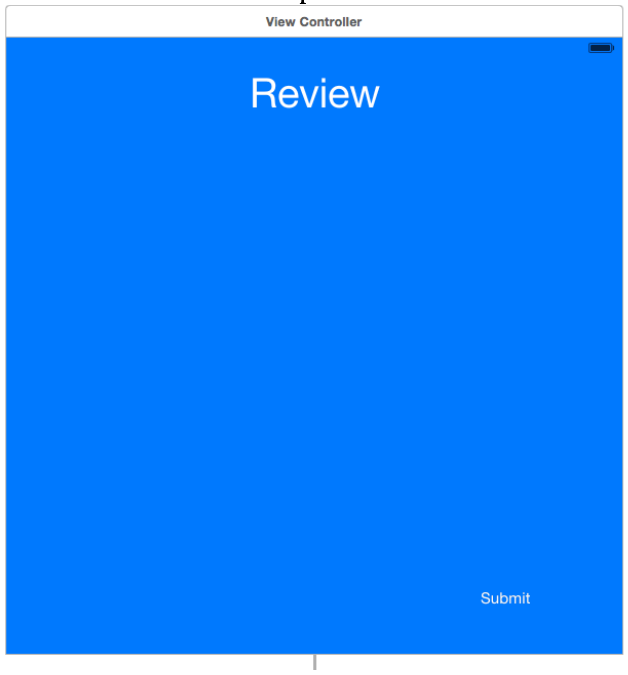




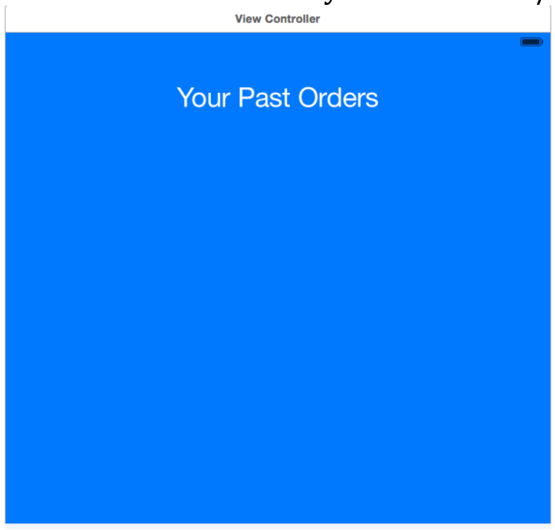
When the user selects the appropriate choice they will be brought to the next storyboard that will auto populate the best Lower Anteriors, but still allow the user to change the selection if they do not like the automated one. The next page will show posteriors and allow the user to view the suggested posteriors for this set which the user can select a different option if they want. On the next storyboard there will be a suggested articulations and mould combinations that will be selected from another function. It will list the upper mould number, articulates with lower mould, combines with 33, 20, 10, or 0 degree posterior mold.



The user will select the appropriate choices and then the shade availability view will be shown there are multiple options for this and I am currently trying to get a image for each shade. The last part of this will be a verify view that the user will verify their selection and will confirm. This will send a order number and a specific code for the order.



This will send a email to the users account for the order and will display the order details. The order number and date will show in the review order page. The system will also send the information to an email for the lab that would design the denture and the details so that they would be able to have the order on hand and ready when the cast/mold is delivered to the lab.



Note the images aren’t finished products for any of the views.