Project 5

By: Eric Dockery

**Gold Challenge Chapter 10 page 210**

**Problem:**

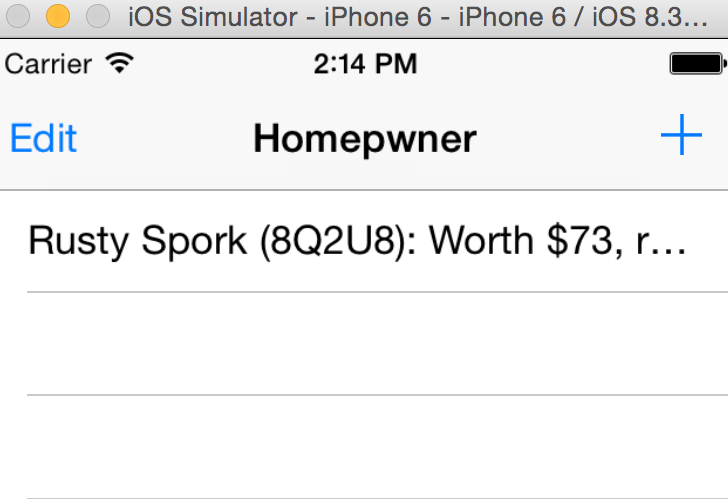
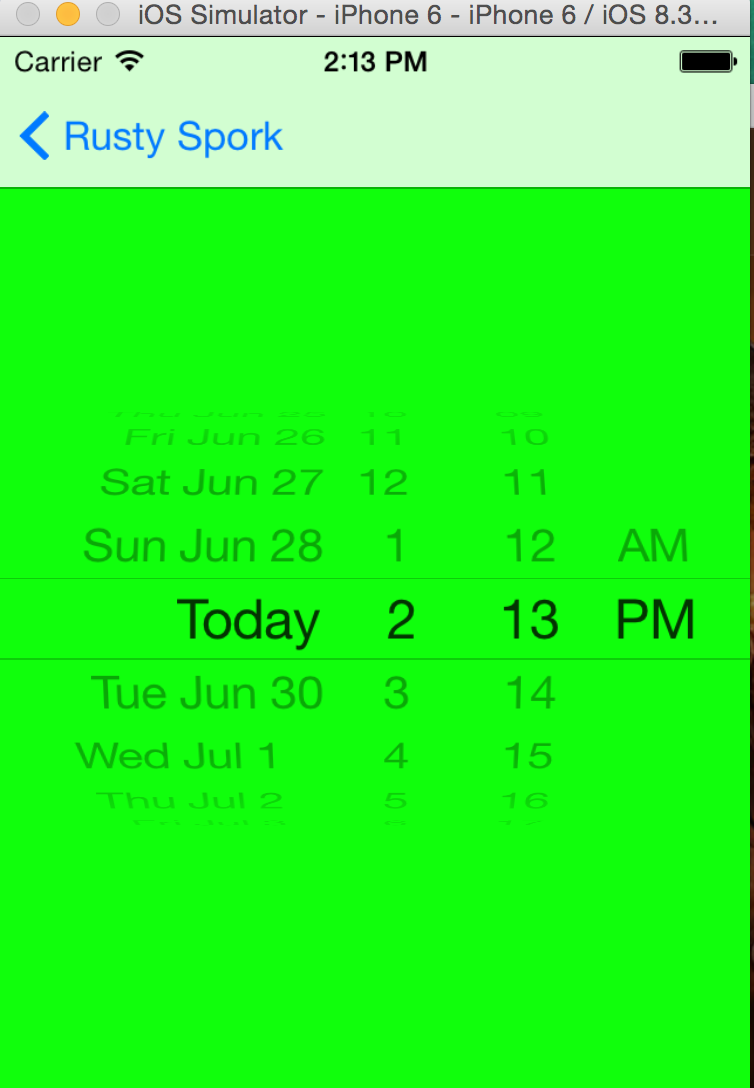
Right now, instances of BNRItem cannot have their dateCreated property changed. Change BNRItem so that they can, and then add a button underneath the dateLabel in BNRDetailViewController with the title Change Date. When this button is tapped, push another view controller instance onto the navigation stack. This view controller should have a UIDatePicker instance that modifies the dateCreated property of the selected BNRItem

**Solution:**

This was a relatively simple solution. The first step was to change the Item.h \*dateCreated value to be able to be read. That required you to go to the Item.h file and change the value from nonatomic, copy to nonatomic, strong and remove the declaration in the Item.m file. Next in the DateViewController.h file you needed to add the Item Calss reference and call the @property (nonatomic, strong) Item \*item. Then create your DateViewController files and add the UIDatePicker to the .xib. add in the viewWillDisappear commands to the DateViewController implementation file. Lastly, add the nesscary code to the DetailViewController implementation file.

Special note on this assignment: I accidently set a breakpoint on my button call

**Screenshots:**

****

**Bronze and Silver Challenges Chapter 11 page 230**

**Problem:**

Bronze:

UIImagePickerController has a built-in interface for editing an image once it has been selected. Allow the user to edit the image and use the edited image instead of the original image in DetailViewController.

Silver:

Add a button that clears the image for an item.

**Solution:**

**Bronze:**

For this challenge there were just a few lines of code to add. First was the imagePicker.allowsEditing =YES;

And then to allow this edited image to be selected you need to add in an if Statement to select the edited image if it was selected: if (info[UIImagePickerControllerEditedImage]) {

image = info[UIImagePickerControllerEditedImage];

} else {

image = info[UIImagePickerControllerOriginalImage];

}

**Silver:**

To solve this challenge you first have to add a button to the controller. Then add the button press to the delegate, I choose to use this code:

- (IBAction)removeimage:(id)sender {

//remove the image

[[ImageStore sharedStore] deleteImageForKey:self.item.itemKey];

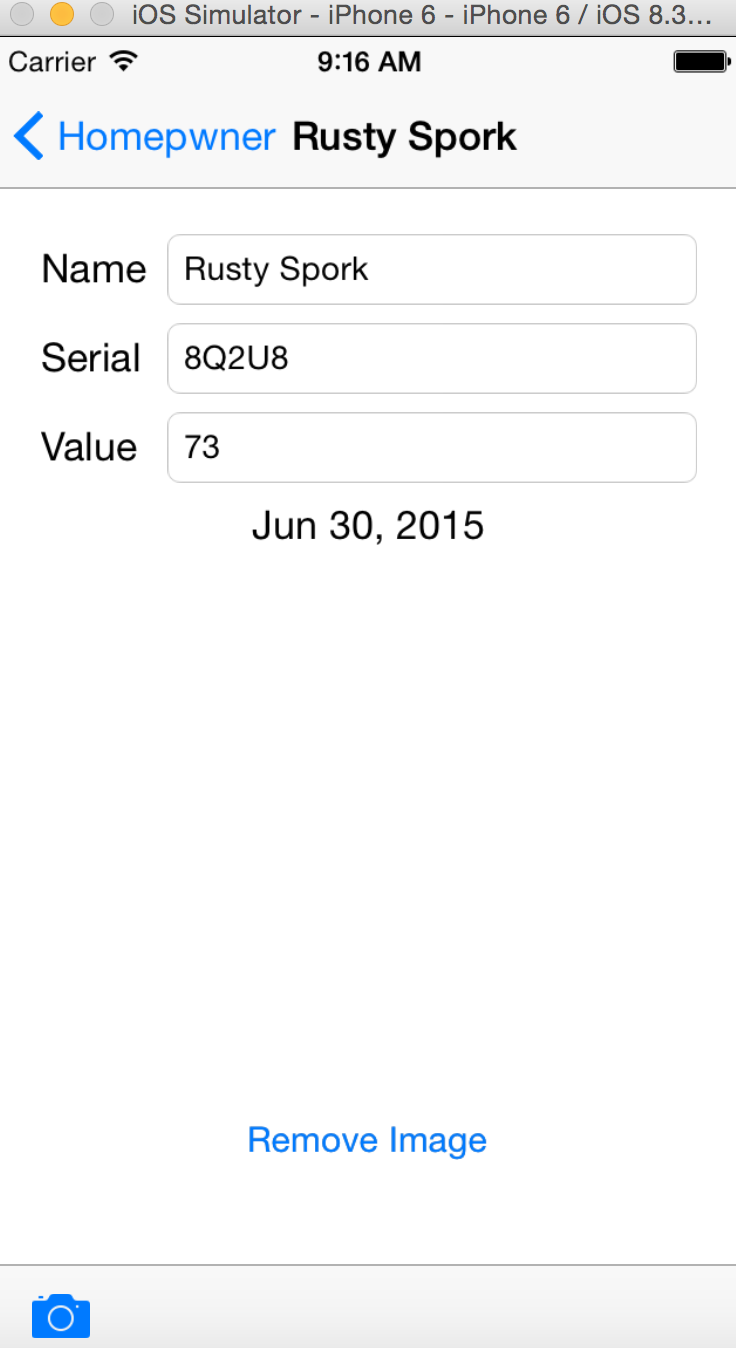
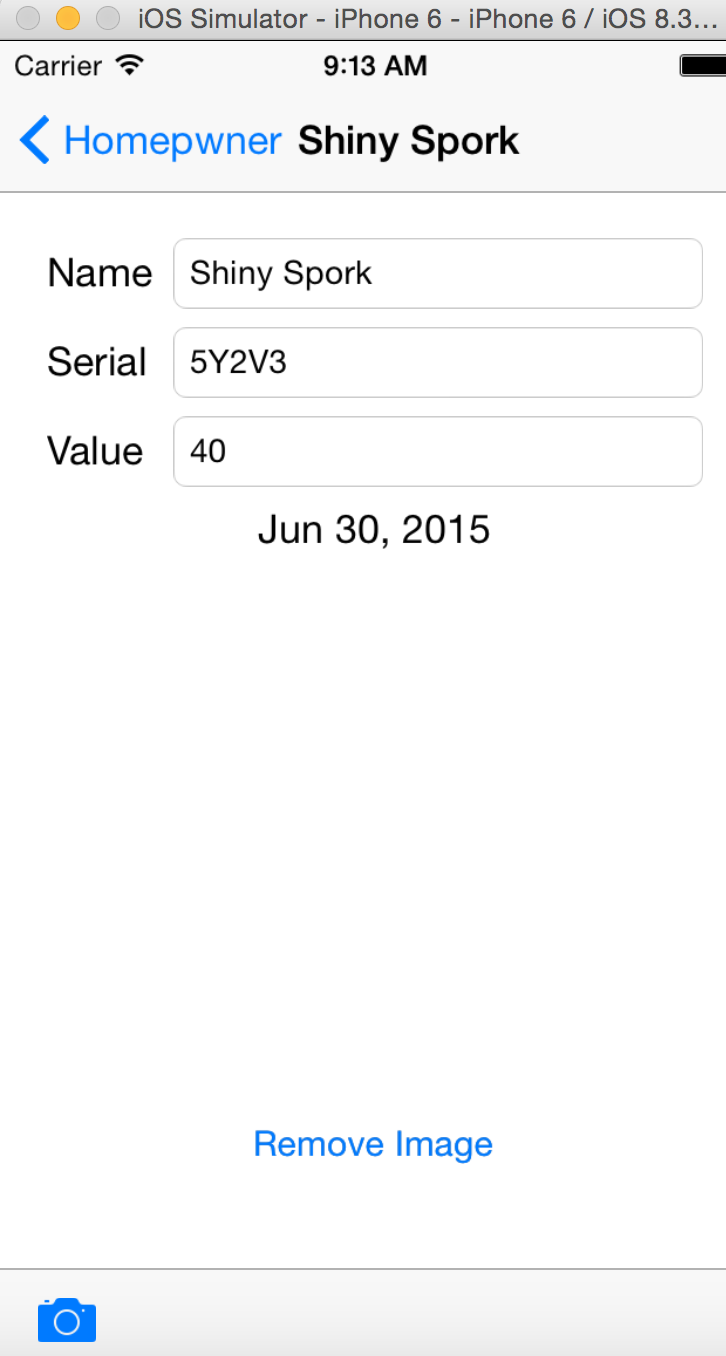
//update the view with no image

self.imageView.image = [UIImage imageNamed:@""];

}

Which gives the image a custom image that I created in Photoshop, telling the user there is no image selected.

**Screenshots:**

****

**Side Note:**

Just removing the image seemed a little basic so in photoshop I created a “no image selected image” I added this image to the app and coded it to show this image. I was going to add an if statement to not allow this to be selected, but I ran out of time.

- (IBAction)removeimage:(id)sender {

//remove the image

[[ImageStore sharedStore] deleteImageForKey:self.item.itemKey];

//update the view with no image

self.imageView.image = [UIImage imageNamed:@"no\_image.jpg"];

}

