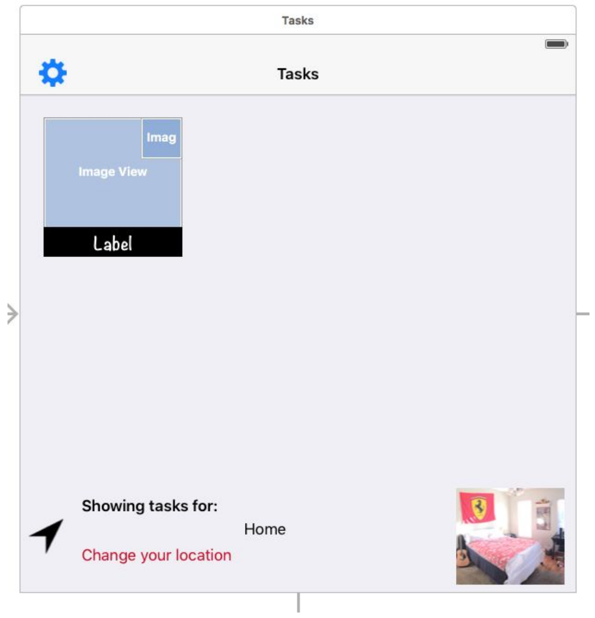
Clemson Life Step by Step Application Technical Manual

Group 1A- Greg Gettings, Jordan Marro, Brendan Giberson

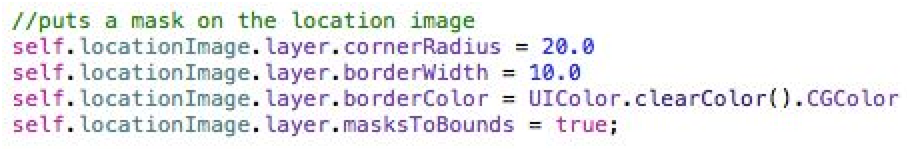
ViewController



This view is the initial view when opening up the app. It makes use of a *UICollectionView* of custom *UICollectionViewCells* defined by the file *CustomCollectionViewCell.* The code to round the cell edges and provide the black background color is contained in that file as well.

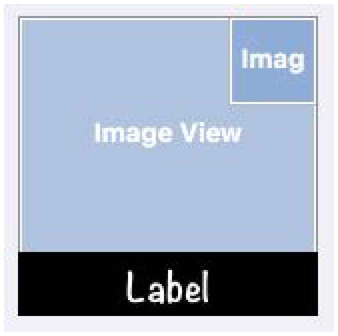
At the bottom, it shows the user's current location in terms of key locations that have been set for the user. Such as home or work. It makes use of the name and picture chosen to represent the key location in the authoring tool. Selecting "Change your location" button at the bottom is currently only for testing purposes, but will allow the user to override their location if an error occurs.

The rounded corners of the location image that appear when the app is ran are done programmatically in the viewDidLoad() function of the ViewController with the following lines of code:



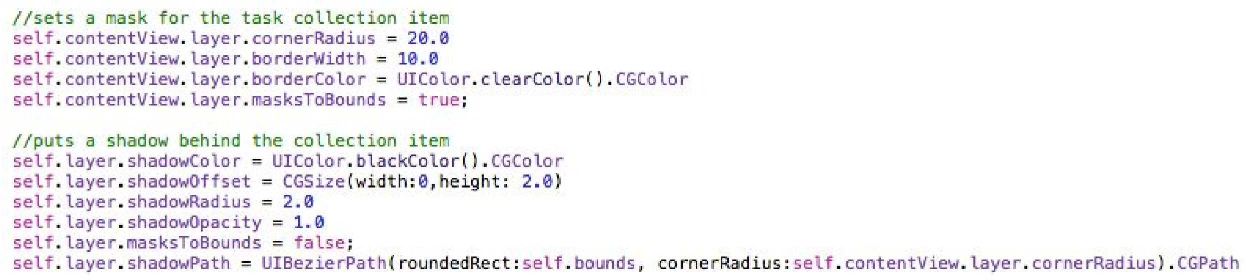
```
//puts a mask on the location image
self.locationImage.layer.cornerRadius = 20.0
self.locationImage.layer.borderWidth = 10.0
self.locationImage.layer.borderColor = UIColor.clearColor().CGColor
self.locationImage.layer.masksToBounds = true;
```

CustomCollectionViewCell



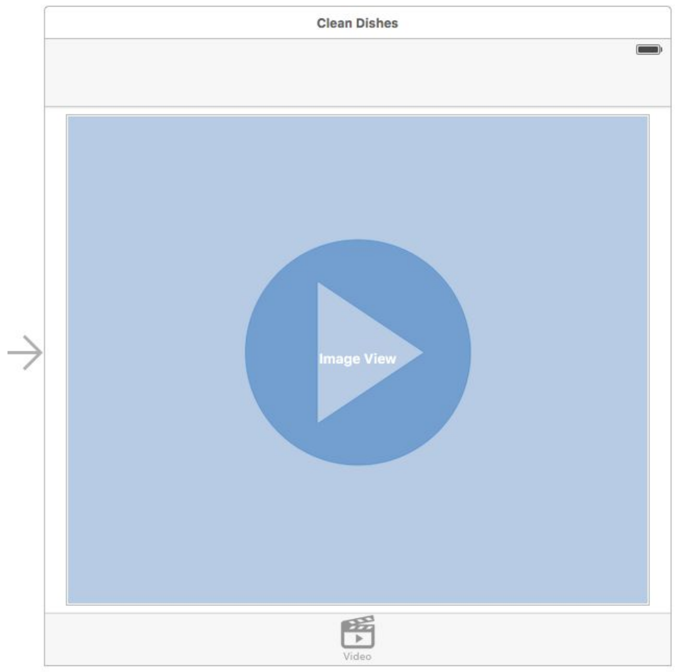
The *CustomCollectionViewCell* is how the user is presented with the tasks on the home screen. It contains a label at the bottom for the task name, an outlet known as *taskName. As well as* an image view that encompasses the majority of the cell. This outlet is known as *taskImage*. And finally another image view outlet in the top right corner known as *comepletionImage* used to place a check mark icon on the task to indicate the task has already been completed.

At runtime, the cells have rounded edges with a black background. This is accomplished with the following lines of code:



```
//sets a mask for the task collection item
self.contentView.layer.cornerRadius = 20.0
self.contentView.layer.borderWidth = 10.0
self.contentView.layer.borderColor = UIColor.clearColor().CGColor
self.contentView.layer.masksToBounds = true;

//puts a shadow behind the collection item
self.layer.shadowColor = UIColor.blackColor().CGColor
self.layer.shadowOffset = CGSize(width:0,height: 2.0)
self.layer.shadowRadius = 2.0
self.layer.shadowOpacity = 1.0
self.layer.masksToBounds = false;
self.layer.shadowPath = UIBezierPath(roundedRect:self.bounds, cornerRadius:self.contentView.layer.cornerRadius).CGPath
```

Video Tab

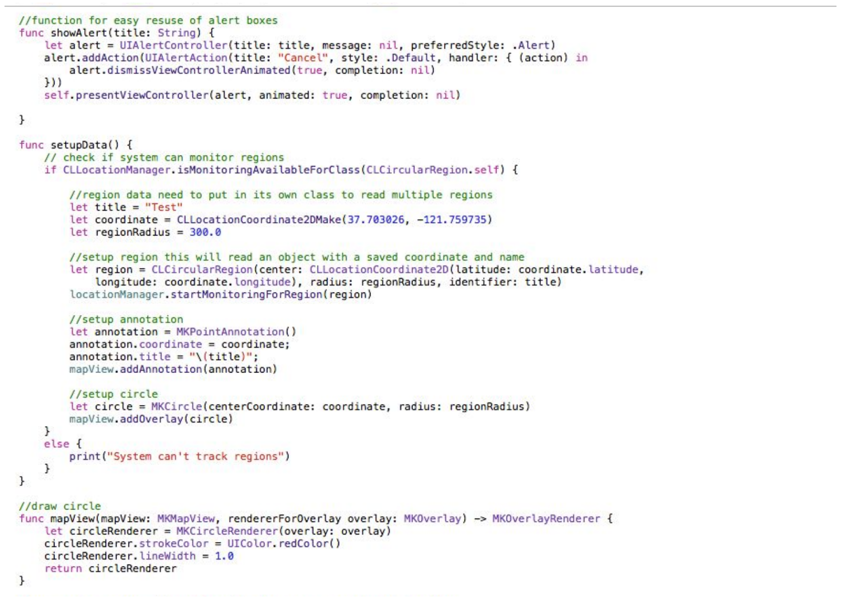


The *VideoTab* view controller is the gateway to access the video guide for the task. It contains an image view outlet called the *imageView*. The *imageView* is used to display a thumbnail image generated from the video file using the following lines of code:

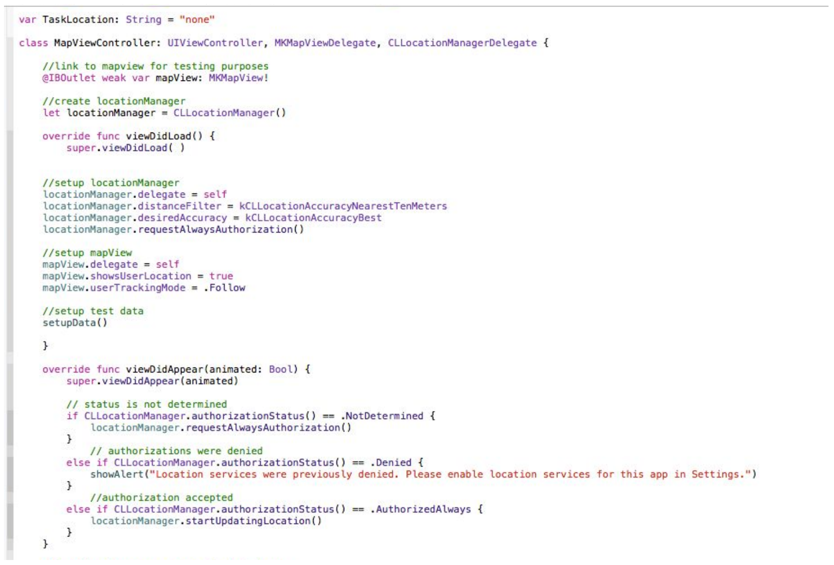
```swift
//generates the background thumbnail image
do {
    let asset = AVURLAsset(URL: fileURL, options: nil)
    let imgGenerator = AVAssetImageGenerator(asset: asset)
    imgGenerator.appliesPreferredTrackTransform = true
    let cgImage = try imgGenerator.copyCGImageAtTime(CMTimeMake(0, 1), actualTime: nil)
    let uiImage = UIImage(CGImage: cgImage)
    imageView.image = uiImage
```

The *VideoTab* also contains a play button known as *button* that presents the *playerViewController,* an AVPlayerViewController created in the *VideoTab* file, when pressed.

MapViewController

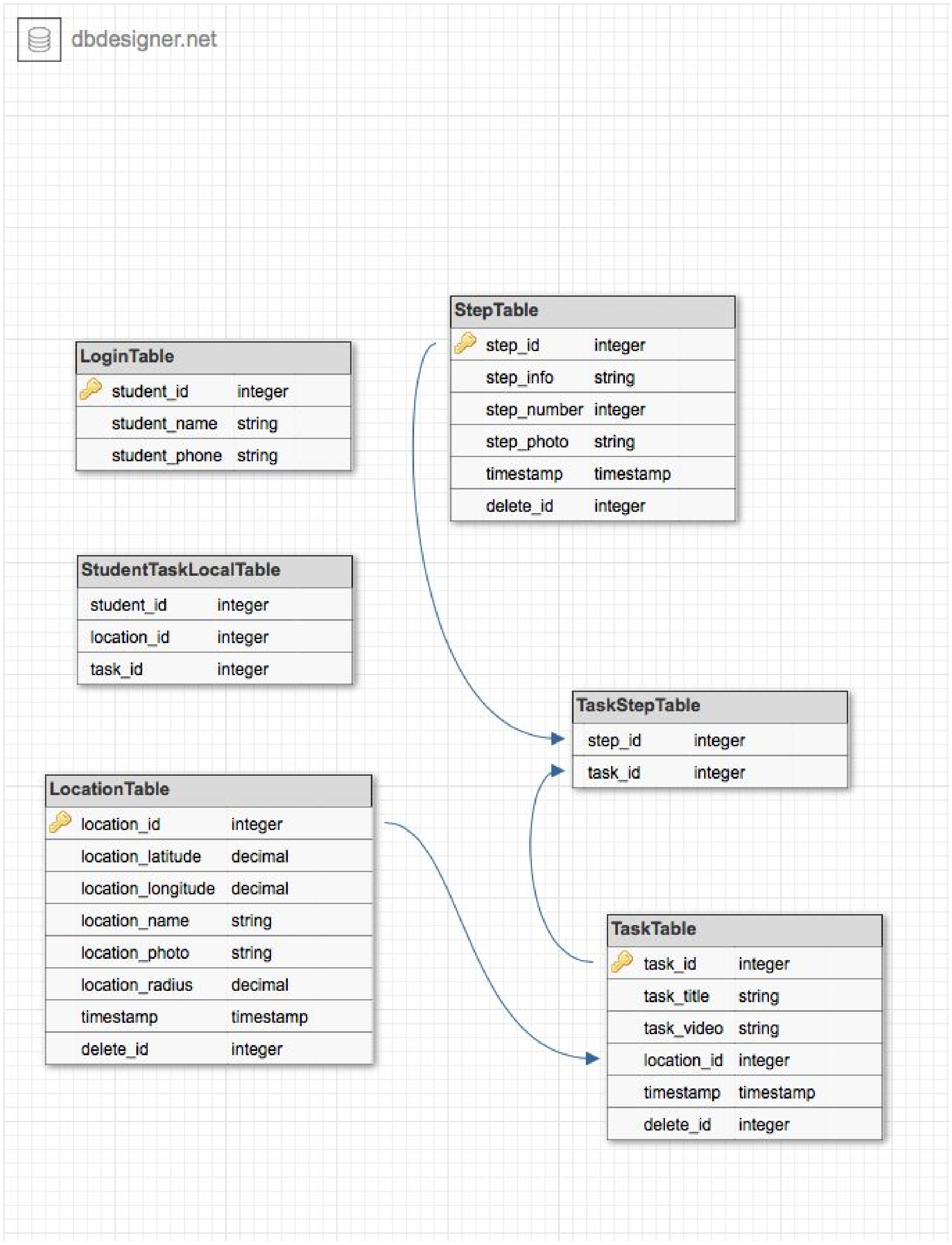


```swift
//function for easy resuse of alert boxes
func showAlert(title: String) {
    let alert = UIAlertController(title: title, message: nil, preferredStyle: .Alert)
    alert.addAction(UIAlertAction(title: "Cancel", style: .Default, handler: { (action) in
        alert.dismissViewControllerAnimated(true, completion: nil)
    }))
    self.presentViewController(alert, animated: true, completion: nil)

}

func setupData() {
    // check if system can monitor regions
    if CLLocationManager.isMonitoringAvailableForClass(CLCircularRegion.self) {

        //region data need to put in its own class to read multiple regions
        let title = "Test"
        let coordinate = CLLocationCoordinate2DMake(37.703026, -121.759735)
        let regionRadius = 300.0

        //setup region this will read an object with a saved coordinate and name
        let region = CLCircularRegion(center: CLLocationCoordinate2D(latitude: coordinate.latitude,
            longitude: coordinate.longitude), radius: regionRadius, identifier: title)
        locationManager.startMonitoringForRegion(region)

        //setup annotation
        let annotation = MKPointAnnotation()
        annotation.coordinate = coordinate;
        annotation.title = "\(title)";
        mapView.addAnnotation(annotation)

        //setup circle
        let circle = MKCircle(centerCoordinate: coordinate, radius: regionRadius)
        mapView.addOverlay(circle)
    }
    else {
        print("System can't track regions")
    }
}

//draw circle
func mapView(mapView: MKMapView, rendererForOverlay overlay: MKOverlay) -> MKOverlayRenderer {
    let circleRenderer = MKCircleRenderer(overlay: overlay)
    circleRenderer.strokeColor = UIColor.redColor()
    circleRenderer.lineWidth = 1.0
    return circleRenderer
}
```

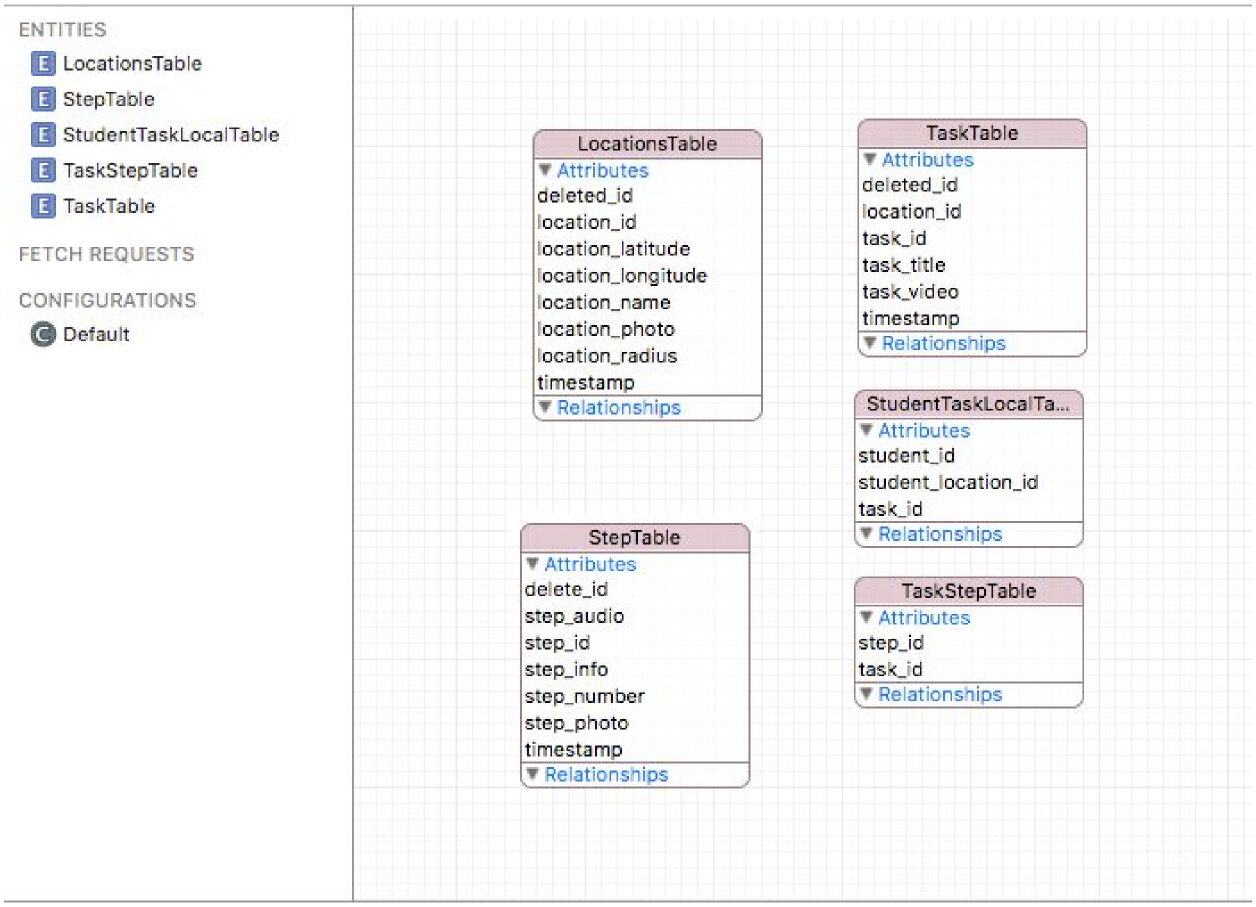


```swift
var TaskLocation: String = "none"

class MapViewController: UIViewController, MKMapViewDelegate, CLLocationManagerDelegate {

    //link to mapview for testing purposes
    @IBOutlet weak var mapView: MKMapView!

    //create locationManager
    let locationManager = CLLocationManager()

    override func viewDidLoad() {
        super.viewDidLoad( )


    //setup locationManager
    locationManager.delegate = self
    locationManager.distanceFilter = kCLLocationAccuracyNearestTenMeters
    locationManager.desiredAccuracy = kCLLocationAccuracyBest
    locationManager.requestAlwaysAuthorization()

    //setup mapView
    mapView.delegate = self
    mapView.showsUserLocation = true
    mapView.userTrackingMode = .Follow

    //setup test data
    setupData()

    }

    override func viewDidAppear(animated: Bool) {
        super.viewDidAppear(animated)

        // status is not determined
        if CLLocationManager.authorizationStatus() == .NotDetermined {
            locationManager.requestAlwaysAuthorization()
        }
            // authorizations were denied
        else if CLLocationManager.authorizationStatus() == .Denied {
            showAlert("Location services were previously denied. Please enable location services for this app in Settings.")
        }
            //authorization accepted
        else if CLLocationManager.authorizationStatus() == .AuthorizedAlways {
            locationManager.startUpdatingLocation()
        }
    }
```

External Database Schema



Internal Database Schema



LocationsTable
▼ Attributes
deleted_id
location_id
location_latitude
location_longitude
location_name
location_photo
location_radius
timestamp
▼ Relationships

TaskTable
▼ Attributes
deleted_id
location_id
task_id
task_title
task_video
timestamp
▼ Relationships

StudentTaskLocalTa...
▼ Attributes
student_id
student_location_id
task_id
▼ Relationships

StepTable
▼ Attributes
delete_id
step_audio
step_id
step_info
step_number
step_photo
timestamp
▼ Relationships

TaskStepTable
▼ Attributes
step_id
task_id
▼ Relationships

PHP Script for the Step Table

```
<?php

// Create connection
$con=mysqli_connect("mysql1.cs.clemson.edu","Team","TeamProj1","TeamProject");

// Check connection
if (mysqli_connect_errno())
{
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
}

// This SQL statement selects ALL from the table 'Locations'
$sql = "SELECT * FROM StepTable";

// Check if there are results
if ($result = mysqli_query($con, $sql))
{
	// If so, then create a results array and a temporary one
	// to hold the data
	$resultArray = array();
	$tempArray = array();

	// Loop through each row in the result set
	while($row = $result->fetch_object())
	{
		// Add each row into our results array
		$tempArray = $row;
	    array_push($resultArray, $tempArray);
	}

	// Finally, encode the array to JSON and output the results
	echo json_encode($resultArray);
}

// Close connections
mysqli_close($con);
?>
```

Example Output from the script:

```
[{"step_id":"1","step_info":"First insert Detergent and Fabric Softener","step_number":"1","step_photo":"picture.png","step_audio":"stepaudio.mp3","delete_id":"0","timestamp":"11"},{"step_id":"2","step_info":"this is actually step 1","step_number":"2","step_photo":"https:\/\/people.cs.clemson.edu\/~jtmarro\/TeamProject\/TeamFiles\/stepExample.jpg","step_audio":"placeholder.mp3","delete_id":"0","timestamp":"11"}]
```