### **Lab 12-** **Create Button Label etc in Swift**

Working with tints, colors, and basic styles in iOS using Swift allows you to customize the appearance of your views to fit your app's design. Below, I'll explain how to set tints, colors, and basic styles for various UIKit views, and show how to create a custom class to apply these styles consistently across your app.

**Example of Setting Basic Styles for Various Views**

Here’s an example of how to apply these properties to different views in a view controller:

import UIKit

class ViewController: UIViewController {

override func viewDidLoad() {

super.viewDidLoad()

// View background color

self.view.backgroundColor = UIColor.lightGray

// Create a label

let label = UILabel(frame: CGRect(x: 20, y: 50, width: 300, height: 50))

label.text = "Hello, World!"

label.textColor = UIColor.darkGray

label.font = UIFont.systemFont(ofSize: 24)

// Create a button

let button = UIButton(type: .system)

button.frame = CGRect(x: 20, y: 120, width: 200, height: 50)

button.setTitle("Tap me", for: .normal)

button.tintColor = UIColor.white

button.backgroundColor = UIColor.systemBlue

button.layer.cornerRadius = 10

// Add shadow to the button

button.layer.shadowColor = UIColor.black.cgColor

button.layer.shadowOpacity = 0.5

button.layer.shadowOffset = CGSize(width: 2, height: 2)

button.layer.shadowRadius = 5

// Add views to the main view

self.view.addSubview(label)

self.view.addSubview(button)

// Add target-action for the button

button.addTarget(self, action: #selector(buttonTapped), for: .touchUpInside)

}

@objc func buttonTapped() {

print("Button was tapped!")

}

}