```
//
//  AdminStudentDetailViewController.swift
//  AttendanceApplication
//


import Foundation
import UIKit
import Alamofire

class AdminBeaconDetailViewController: UIViewController {

    // MARK: IBOutlets

    @IBOutlet weak var roomNumberField: UITextField!
    @IBOutlet weak var zoneNameField: UITextField!
    @IBOutlet weak var descriptionField: UITextField!
    @IBOutlet weak var statusLabel: UILabel!
    @IBOutlet weak var updateBeaconButton: UIButton!
    @IBOutlet weak var deleteBeaconButton: UIButton!

    // MARK: Private properties

    private var zonesNames: [String] = []
    private var selectedZone: String?

    // MARK: Public properties

    var beacon: Beacon!

    // MARK: UIViewController Methods

    override func viewDidLoad() {
        super.viewDidLoad()

        updateBeaconButton.layer.cornerRadius = 4
        deleteBeaconButton.layer.cornerRadius = 4

        getZones()
        createZonePicker()

        roomNumberField.text = beacon.roomNumber
        zoneNameField.text = beacon.zoneName
        descriptionField.text = beacon.description
    }

    // MARK: IBAction methods

    @IBAction func updateBeaconButton(_ sender: Any) {
        updateBeacon()
    }

    @IBAction func deleteBeaconButton(_ sender: Any) {
        deleteBeacon()
```

```
}

// MARK: Internal methods

internal func createZonePicker() {
    let zonePicker = UIPickerView()
    zonePicker.delegate = self

    zoneNameField.inputView = zonePicker
}

internal func updateBeacon(){
    let parameters: Parameters = [
        "type": "admin.edit_beacon",
        "args": [
            "beacon_id": beacon.id,
            "room_number": roomNumberField.text,
            "zone_name": zoneNameField.text,
            "description": descriptionField.text
        ]
    ]

    Alamofire.request(HTTPHelper.url, method: .post, parameters: parameters,
     encoding: JSONEncoding.default).responseJSON {
        response in

        switch response.result {
        case .failure( _):

            return

        case .success(let data):
            // First make sure a dictionary is recieved: Data validation
            guard let json = data as? [String : AnyObject] else {
                // Print statement for debugging purposes, not seen by users.
                print("Failed to get expected dictionary from webserver.")
                return
            }

            // Then make sure that key/value pairs are correct: Data validation
            guard let success = json["successful"] as? Int, let reason =
             json["reason"] as? String else {
                // Print statement for debugging purposes, not seen by users.
                print("Failed to get expected data from webserver")
                return
            }

            if success == 1 {
                self.statusLabel.text = "Successfully edited beacon " +
                 self.roomNumberField.text!
            } else {
                self.statusLabel.text = "Failed to edit beacon " +
                 self.roomNumberField.text! + ": " + reason
            }
```

```
        }
    }
}

internal func deleteBeacon(){
    let parameters: Parameters = [
        "type": "admin.delete_beacon",
        "args": [
            "beacon_id": beacon.id,
        ]
    ]

    Alamofire.request(HTTPHelper.url, method: .post, parameters: parameters,
     encoding: JSONEncoding.default).responseJSON {
        response in

        switch response.result {
        case .failure( _):

            return

        case .success(let data):
            // First make sure a dictionary is recieved: Data validation
            guard let json = data as? [String : AnyObject] else {
                // Print statement for debugging purposes, not seen by users.
                print("Failed to get expected dictionary from webserver.")
                return
            }

            // Then make sure that key/value pairs are correct: Data validation
            guard let success = json["successful"] as? Int, let reason =
             json["reason"] as? String else {
                // Print statement for debugging purposes, not seen by users.
                print("Failed to get expected data from webserver")
                return
            }

            if success == 1 {
                self.statusLabel.text = "Successfully deleted beacon " +
                 self.roomNumberField.text!
            } else {
                self.statusLabel.text = "Failed to delete beacon " +
                 self.roomNumberField.text! + ": " + reason
            }
        }
    }
}

internal func getZones(){
    let parameters: Parameters = [
        "type": "admin.get_zones",
        "args": [
            "query": ""
        ]
```

```
        ]

        Alamofire.request(HTTPHelper.url, method: .post, parameters: parameters,
         encoding: JSONEncoding.default).responseJSON {
            response in

            switch response.result {
            case .failure( _):

                return

            case .success(let data):
                // First make sure a dictionary is recieved: Data validation
                guard let json = data as? [String : AnyObject] else {
                    // Print statement for debugging purposes, not seen by users.
                    print("Failed to get expected dictionary from webserver.")
                    return
                }

                // Then make sure that key/value pairs are correct: Data validation
                guard let success = json["successful"] as? Int, let zones =
                 json["zones"] as? [String] else {
                    // Print statement for debugging purposes, not seen by users.
                    print("Failed to get expected data from webserver")
                    return
                }

                if success == 1 {
                    self.zonesNames = zones
                } else {

                }
            }
        }
    }
}

// MARK: Extentions

extension AdminBeaconDetailViewController: UIPickerViewDelegate,
 UIPickerViewDataSource {
    func numberOfComponents(in pickerView: UIPickerView) -> Int {
        return 1
    }

    func pickerView(_ pickerView: UIPickerView, numberOfRowsInComponent component:
     Int) -> Int {
        return zonesNames.count
    }

    func pickerView(_ pickerView: UIPickerView, titleForRow row: Int, forComponent
     component: Int) -> String? {
        return zonesNames[row]
    }
```

```
    func pickerView(_ pickerView: UIPickerView, didSelectRow row: Int, inComponent
     component: Int) {
        selectedZone = zonesNames[row]
        zoneNameField.text = selectedZone
    }
}
```