Inventory Management Application Documentation

Contents

Introduction

The Slab app is scans, recognizes, and manages slab inventory data using an iPhone. The app uses Apple's Vision Kit library to scan text and manage inventory data efficiently, taking away the need to check manually using paper and pen which is tedious. This file will provide an explanation of the app's functionality so it can be developed or maintained.

Features

The app includes the following features:
- Scanning and recognizing text using the device's camera.
- Managing slab inventory in the database so add, check if a slab is in its right location.
- Edit the location of a slab to further improve management

Screenshots and File Functionality

MVC STRUCTURE

MAIN/

---NucorSlabsApp.Swift

MODELS/

---DataModel.swift

VIEWS/

---ContentView.swift

---CheckedItemView.swift

---ChangeLocationView.swift

---AddSlabView.swift

CONTROLLERS/

---SlabInventoryViewModel

---IDScannerView

SlabApp.swift

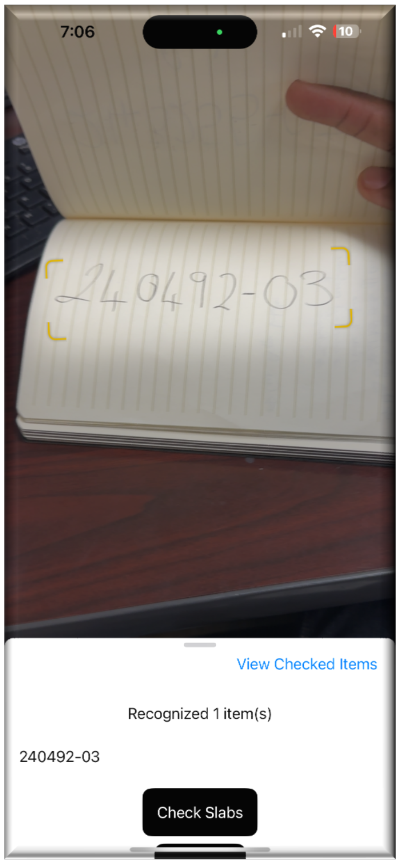
This is the main entry point of the app. It initializes the application and sets up the main environment object, SlabInventoryViewModel, which manages the state of the app.

DataModel.swift

This file contains the data model for the slabs, represented by the Slab struct. It has an ObservableObject called SlabDataManager that handles data management for the app, and functions responsible for loading, saving, and updating slab data both locally and remotely.

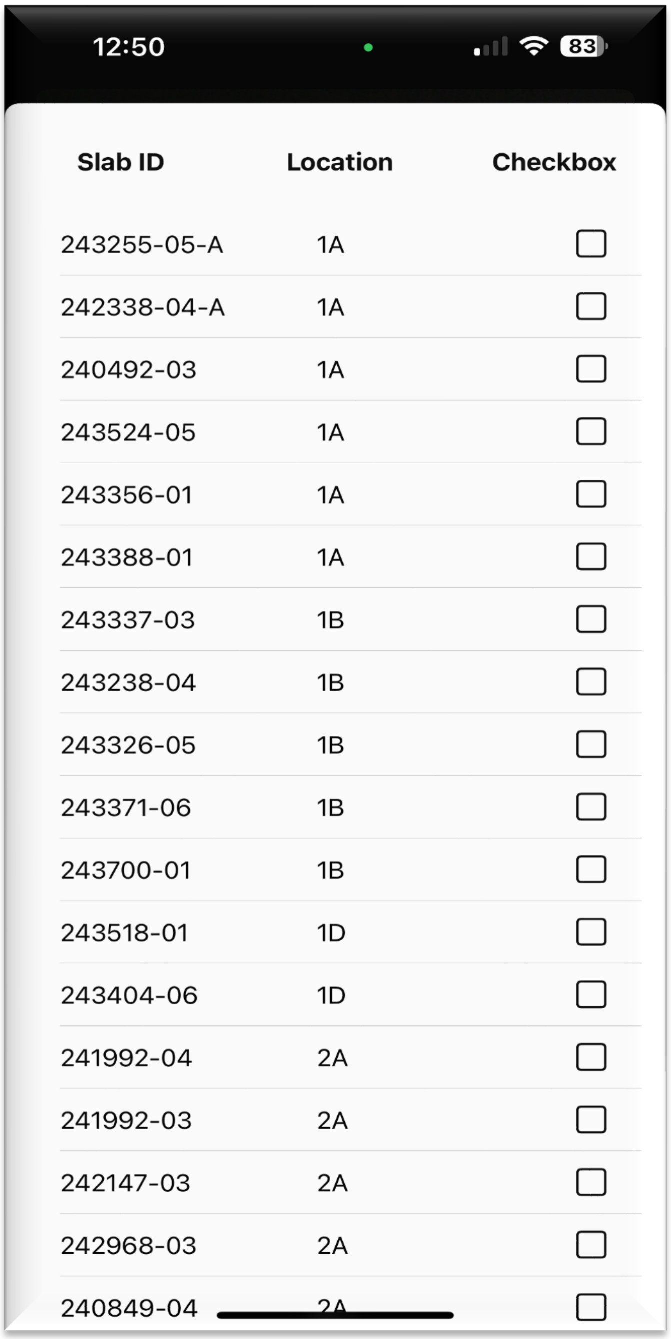
ContentView.swift

ContentView is the main view of the app. It displays different views based on the data scanner access status and allows users to scan slabs, add new slabs, and view checked items. It uses the IDScannerView component to handle the scanning functionality.



Recognized 1 item(s)

240492-03

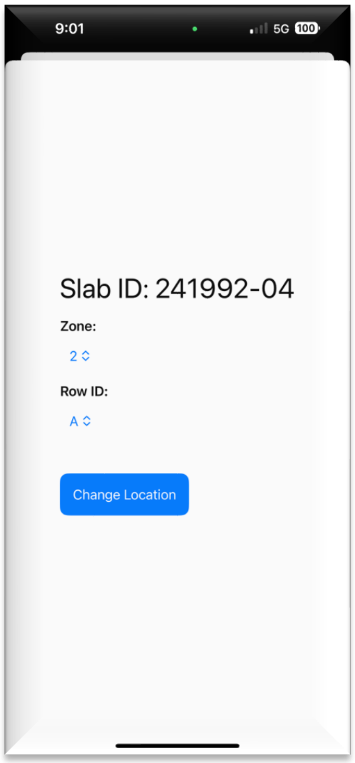


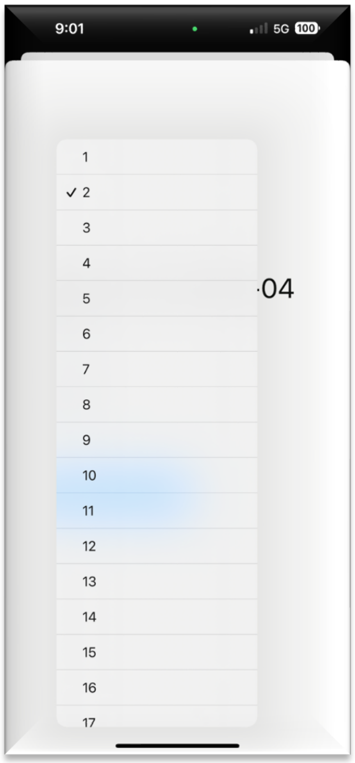
Slab ID	Location	Checkbox
243255-05-A	1A	☐
242338-04-A	1A	☐
240492-03	1A	☐
243524-05	1A	☐
243356-01	1A	☐
243388-01	1A	☐
243337-03	1B	☐
243238-04	1B	☐
243326-05	1B	☐
243371-06	1B	☐
243700-01	1B	☐
243518-01	1D	☐
243404-06	1D	☐
241992-04	2A	☐
241992-03	2A	☐
242147-03	2A	☐
242968-03	2A	☐
240849-04	2A	☐

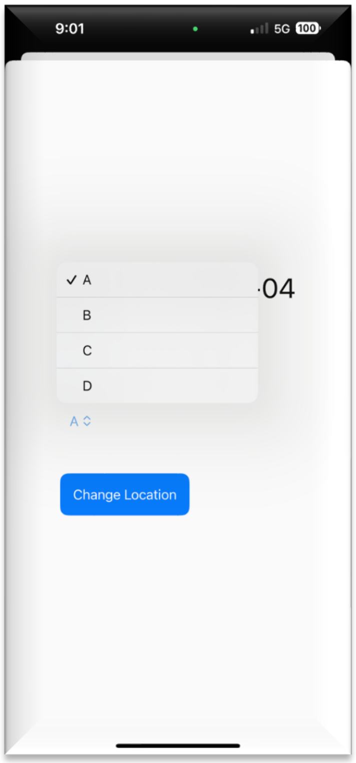
CheckedItemsView.swift
CheckedItemsView displays a list of checked slabs. It provides an interface for viewing slab details, updating slab locations, and toggling the checked status of slabs.

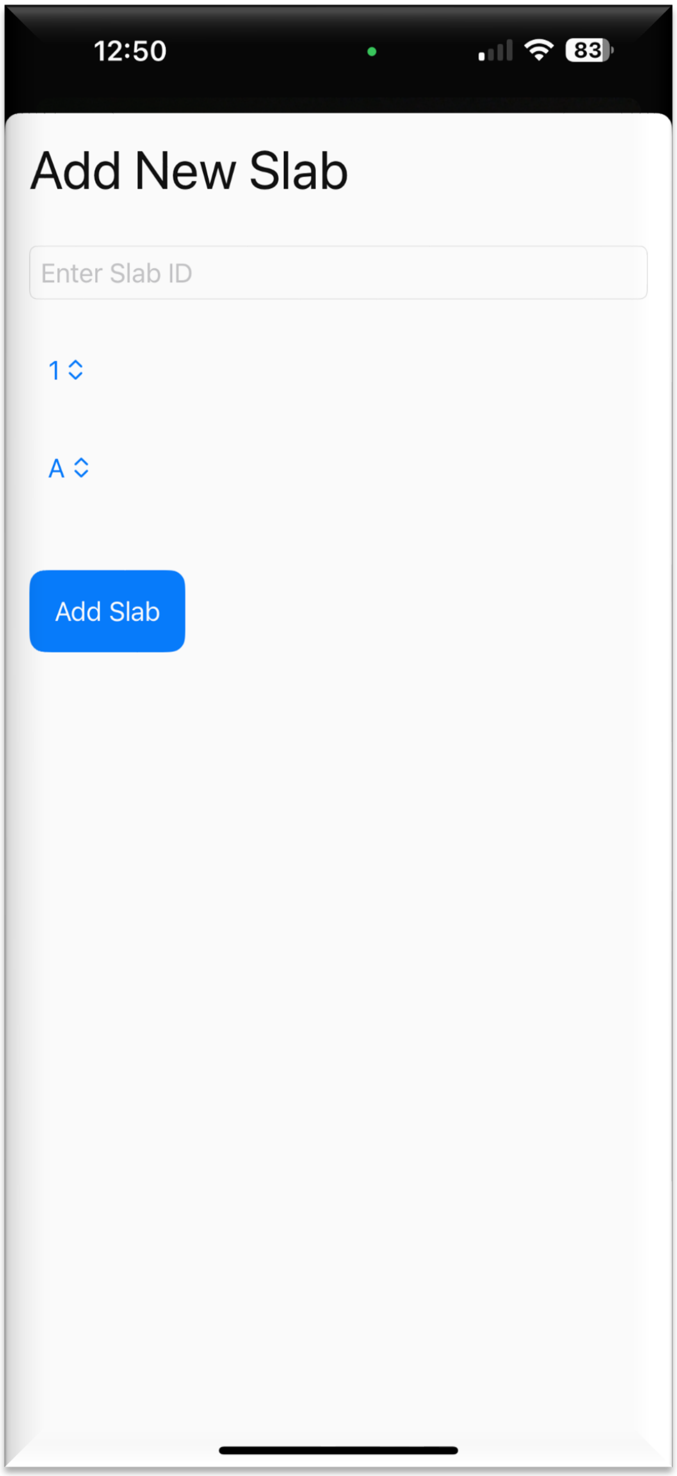
ChangeLocationsView.swift

This view provides an interface for updating a slab's location within the yard. Users can select new values for the zone and row ID, and the app updates these changes on the server.









AddSlabView.swift

AddSlabView provides the interface for adding new slabs to the inventory. Users can enter the slab ID, select the zone and row ID, and add the slab to the inventory. The view also validates the input to ensure that the slab ID is not empty. Because slabs can be in inventory but not reflected on the slab locations page.

SlabInventoryViewModel.swift

The SlabInventoryViewModel class manages the scanner's state, including camera access, recognized items, and user permissions. It handles the requests for camera access and updates the scanner status accordingly.

IDScannerView.swift

A UIViewControllerRepresentable that integrates Apple's VisionKit for scanning text and barcodes. It manages the setup and control of the DataScannerViewController, including the start/stop scanning process and updating recognized items.

Deployment

Using a personal developer account, it distributes the app to those that work in Slab Inventory through Ad hoc distribution.

1.) Assign the application a certificate, and an app identifier which is obtained by going to the certificates, Identifiers & profiles, make sure these items are active in your macs keychain.

2.) A.) Create an ad hoc provisioning profile for the application in the same section,
B.) Which you also need to add people's devices that will use the app by putting down their UDID. You can get their UDID number by using Xcode Connect the Device to your laptop. On Xcode go to Window > Devices and Simulators, and then select the connected device from the list on the left. The UDID will be shown
C.) Now Open your project in Xcode, select the target for your app, go to the Signing & Capabilities tab, choose your Ad Hoc provisioning profile under Provisioning Profile and Ensure that your build configuration is set to Release

3.) Archive Your App in Xcode, go to Product > Archive to create an archive of your app, After the archiving process completes, the Xcode Organizer window will open, displaying your archived builds, select your archive in the Organizer window click on Distribute App, choose Ad Hoc as the distribution method

4.) Export the App, after validation, choose Export to generate an IPA file (the file format for iOS apps), Save the IPA file to a location on your computer.

Installation

1.) To install the app, you can distribute the IPA file using Apple Configurator. Download and open Apple Configurator ‘2’. Connect the device to your Mac. Drag the IPA file into Apple Configurator 2 to install the app or you can use Drag the IPA file into iTunes and sync it to the device.
2.) To Register Additional Devices, Collect UDIDs. Update the Provisioning Profile. Go back to the Apple Developer portal and update the Ad Hoc provisioning profile to include the new devices. Download the updated provisioning profile and apply it in Xcode. Rebuild the app with the updated provisioning profile, archive it again, and export the IPA file for distribution.

Make sure the phone is compatible to do OCR

Security

Can only be deployed to specific devices by getting their UDID.

Conclusion

The Nucor Slab app provides an efficient way to manage slab inventory using text recognition. This app has potential to save time and fuel as there is better management of the slabs. This document covers the key components and features of the app and how the code accomplishes that.