LAB 4

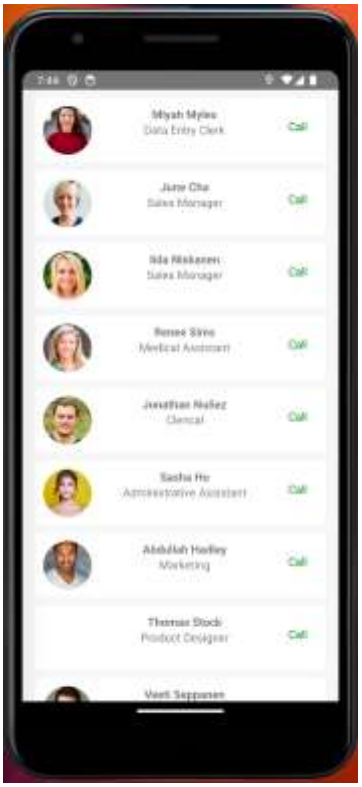
MỤC TIÊU

Kết thúc bài thực hành sinh viên có khả năng:

- ✓ Làm quen với FlatList, ScrollView, KeyboardAvoidingView và StatusBar component
- ✓ Kết hợp sử dụng các component lại với nhau

NỘI DUNG

BÀI 1: SỬ DỤNG FLATLIST TẠO DANH BẠ



Hướng dẫn:

- Tạo danh sách dữ liệu danh bạ mẫu, mỗi object trong danh sách có cấu trúc như sau:

```
type ContactType = {
  name: string;
  email: string;
  position: string;
  photo: string;
};
```

- Hiển thị danh sách danh bạ với **FlatList**, **data** là dữ liệu mẫu đã được bạn tạo sẵn:

```
return (
  <View style={styles.container}>
    <FlatList
      data={data}
      renderItem={({item}) => <ContactItem contact={item} />}
      keyExtractor={item => item.email}
    />
  </View>
);
```

- Tạo item cho **FlatList**:

```
const ContactItem = ({contact}: {contact: ContactType}) => (
  <View style={styles.listItem}>
    <Image source={{uri: contact.photo}} style={styles.avatar} />
    <View style={styles.bodyItem}>
      <Text style={styles.nameText}>{contact.name}</Text>
      <Text>{contact.position}</Text>
    </View>
    <TouchableOpacity style={styles.btnCall}>
      <Text style={styles.callText}>Call</Text>
    </TouchableOpacity>
  </View>
);
```

BÀI 2: THAY ĐỔI MÀU STATUSBAR KẾT HỢP VỚI REFRESHCONTROL

Yêu cầu:

- Bạn phải sử dụng **ScrollView** để chứa nội dung, sau đó gắn thêm **RefreshControl** vào **ScrollView**, nếu người dùng muốn thay đổi màu status bar thì kéo xuống để tải lại màu.

Hướng dẫn:

- Tạo **ScrollView** để bọc lại ứng dụng, sau đó thêm **StatusBar** vào bên trong để có thể thay đổi màu. Để bắt được sự kiện kéo xuống đầu trang, bạn thêm **RefreshControl** vào **ScrollView.**

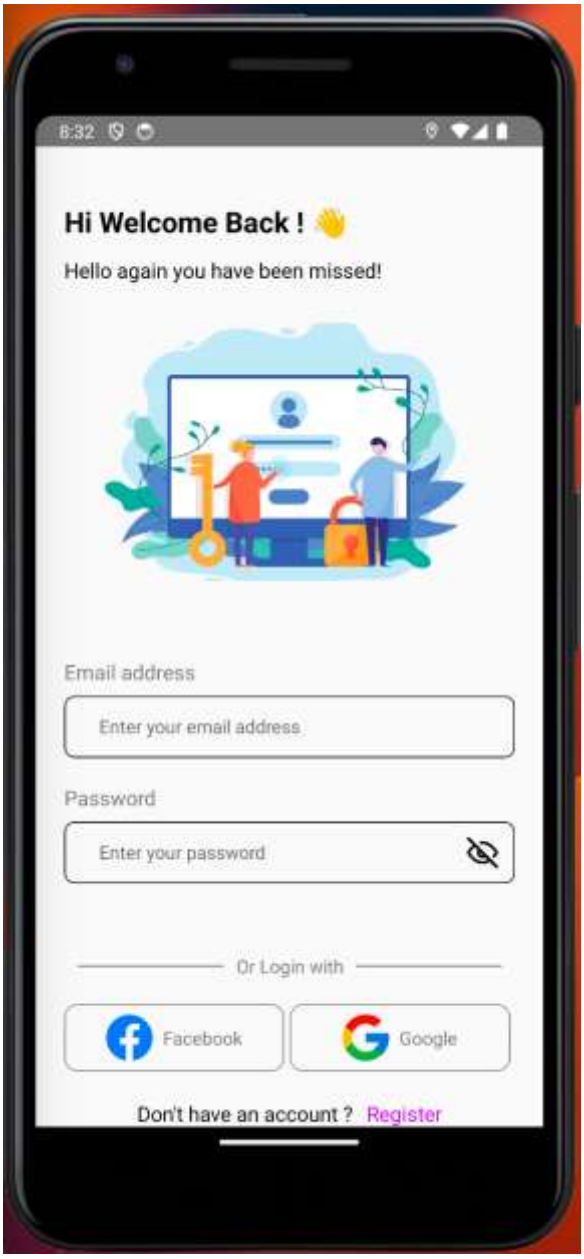
```
return (
  <ScrollView
    style={styles.container}
    refreshControl={
      <RefreshControl refreshing={refreshing} onRefresh={onRefresh} />
    }>
    <StatusBar
      barStyle={barStyle}
      translucent
      backgroundColor={'transparent'}
    />

    <Text style={styles.text}> Kéo xuống để đổi màu Statusbar</Text>
  </ScrollView>
);
```

- Bạn viết thêm hàm **onRefresh** để thay đổi state **barStyle**

```
const [barStyle, setBarStyle] = useState<StatusBarStyle>('light-content');
const [refreshing, setRefreshing] = React.useState(false);
```

BÀI 3: XÂY DỰNG MÀN HÌNH ĐĂNG NHẬP, KẾT HỢP KEYBOARDAVOIDINGVIEW ĐỂ TRÁNH KEYBOARD CHE INPUT KHI NHẬP.



Yêu cầu:

- Phần input password có thể bật tắt che hoặc không che mật khẩu.

- Khi nhập nội dung vào input, phần keyboard không được che phần input nhập, phải sử dụng **KeyboardAvoidingView.**

Hướng dẫn:

- Bọc **KeyboardAvoidingView** ngoài cùng ứng dụng của bạn, bọc thêm **ScrollView** để layout ứng dụng có thể được đẩy lên.

```
return (
  <KeyboardAvoidingView
    behavior={Platform.OS === 'ios' ? 'padding' : 'height'}>
    <ScrollView style={styles.container}>…
    </ScrollView>
  </KeyboardAvoidingView>
);
```

- Các phần UI khác các bạn code theo hình đề bài, riêng phần TextInput nhập mật khẩu các bạn cần thêm prop **secureTextEntry** dạng boolean để bật hoặc tắt che mật khẩu, ngoài ra bạn cũng cần chèn thêm icon hình con mắt để người dùng nhấn vào.

```
<View style={styles.wrapInput}>
  <TextInput
    placeholder="Enter your password"
    secureTextEntry={isPasswordShown}
    style={styles.styleInput}
  />

  <TouchableOpacity…
  </TouchableOpacity>
</View>
```

BÀI 4: GV CHO THÊM

***** YÊU CẦU NỘP BÀI:**

Sv nén file bao gồm các yêu cầu đã thực hiện trên, nộp lms đúng thời gian quy định của giảng viên. Không nộp bài coi như không có điểm.

--- Hết