DANIEL GRAHAM

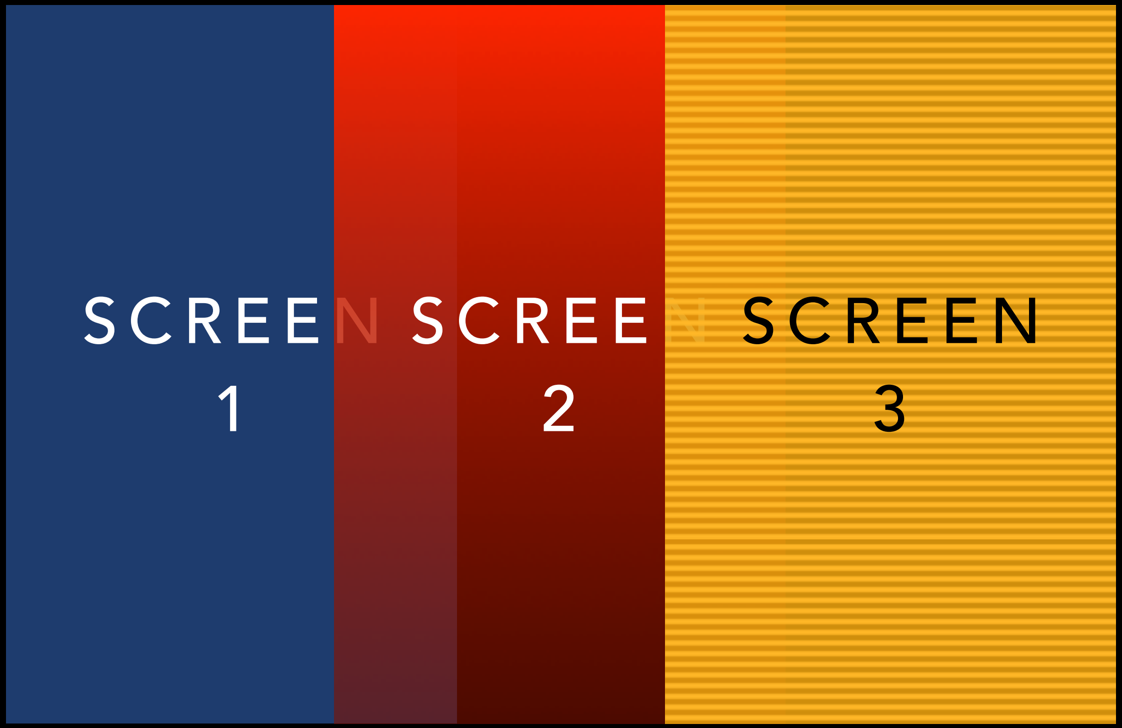
NAVIGATION REACT NATIVE

GETTING STARTED WITH REACT NAVIGATION

- npm install --save react-navigation
- https://reactnavigation.org/docs/en/getting-started.html

STACK NAVIGATOR

THINK BACK BUTTON IN YOUR BROWSER



By default the stack navigator is configured to have the familiar iOS and Android look & feel: new screens slide in from the right on iOS, fade in from the bottom on Android. On iOS the stack navigator can also be configured to a modal style where screens slide in from the bottom.

NOMENCLATURE

- Screens
  - JSX component that gets rendered
- Routes
  - Name JSX Component screen pairs

- https://snack.expo.io/@react-navigation/hello-react-navigation-v3

LETS CREATE A NEW SCREEN TO NAVIGATE TO

```
import React from 'react';
import {View, Text} from 'react-native'
export default class PlayerScreen extends React.Component {
    render() {
      return (
        <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
          <Text>PlayerScreen</Text>
        </View>
      );
    }
  }
```

THIS WILL EVENTUALLY BECOME THE SCREEN THAT PLAYS THE PODCAST

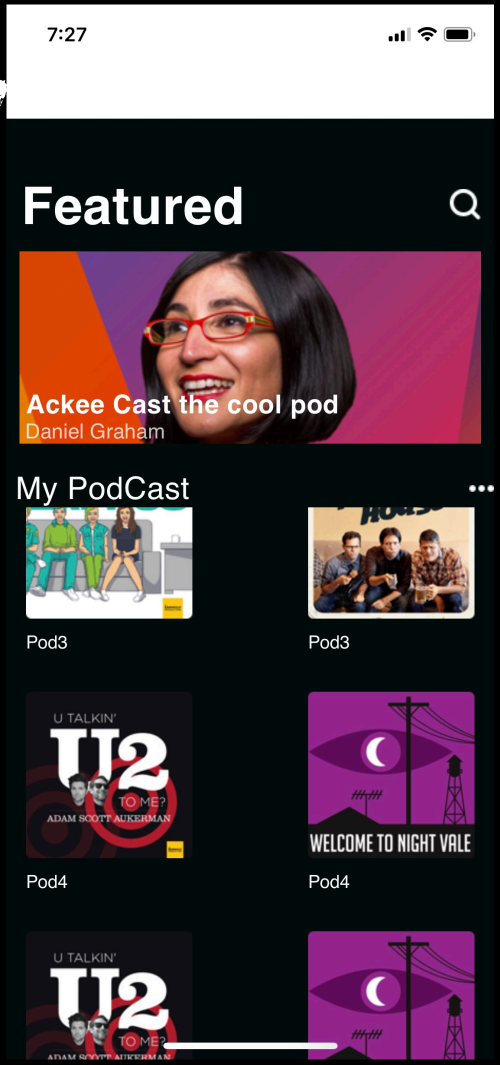
STACK NAVIGATOR ENTRIES

```
createStackNavigator({
  // For each screen that you can navigate to, create a new entry like this:
  Profile: {
    // `ProfileScreen` is a React component that will be the
    // main content of the screen.
    screen: ProfileScreen,
  },
  ...MyOtherRoutes,
});
```

MODIFYING THE ENTRY POINT (APP.JS)

```
import React from 'react'
import { createStackNavigator, createAppContainer } from 'react-navigation'
import HomeScreen from './components/HomeScreen'
import PlayerScreen from './components/PlayerScreen'

const RootStack = createStackNavigator(
  {
    Home: HomeScreen,
    Player: PlayerScreen,
  },
  {
    initialRouteName: 'Home',
  }
);

const AppContainer = createAppContainer(RootStack);

export default class App extends React.Component {
  render() {
    return (<AppContainer />)
  }
}
```

NOTICE BAR



MOVING BETWEEN SCREENS

PARADIGM ON THE WEB

```
<a href="details.html">Go to Details</a>
```

```
<a onClick={() => { document.location.href = "details.html"; }}>Go to Details</a>
```

PARADIGM ON IN REACT NAVIGATION

```
<Button
     title="Go to Details"
     onPress={() => this.props.navigation.navigate('Details')}
      />
```

```
<Button
     title="Go to Details"
     onPress={() => this.props.navigation.navigate('Details')}
      />
```

- this.props.navigation: the navigation prop is passed in to every screen component (definition) in stack navigator

- navigate('Details'): we call the navigate function with the name of the route that we'd like to move the user to.

```
<Button
     title="Go to Details"
     onPress={() => this.props.navigation.navigate('Details')}
      />
```

- If you already on the details calling navigation will not do anything

```
<Button
  title="Go to Details... again"
  onPress={() => this.props.navigation.push('Details')}
/>
```

- However you might want another details screen that shows details about another item.
(Many with some unique data: params or redux.

PROGRAMMATICALLY GOING BACK

```
<View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
        <Text>Details Screen</Text>
        <Button
          title="Go to Details... again"
          onPress={() => this.props.navigation.push('Details')}
        />
        <Button
          title="Go to Home"
          onPress={() => this.props.navigation.navigate('Home')}
        />
        <Button
          title="Go back"
          onPress={() => this.props.navigation.goBack()}
        />
      </View>
```

SNACK ON REACT NAVIGATION

- https://snack.expo.io/@react-navigation/going-back-v3

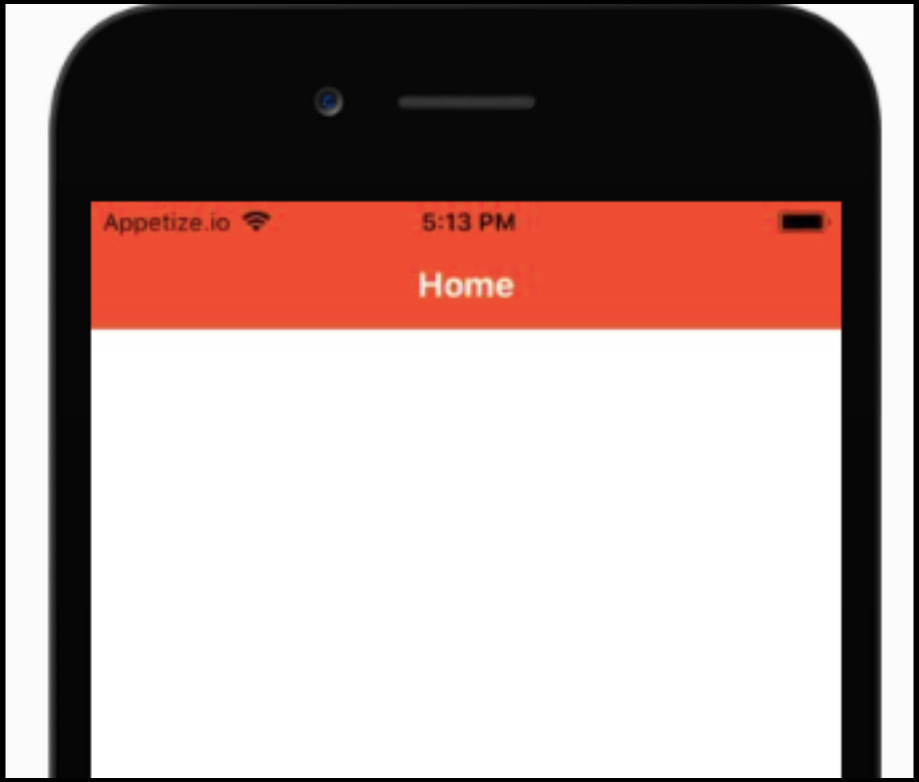
ADDING NAVIGATION TO PODCAST

```
   <TouchableOpacity onPress={()=>{this.props.navigation.navigate('Player')}}>
            <Featured/>
        </TouchableOpacity>
```

STYLING SUPPORT

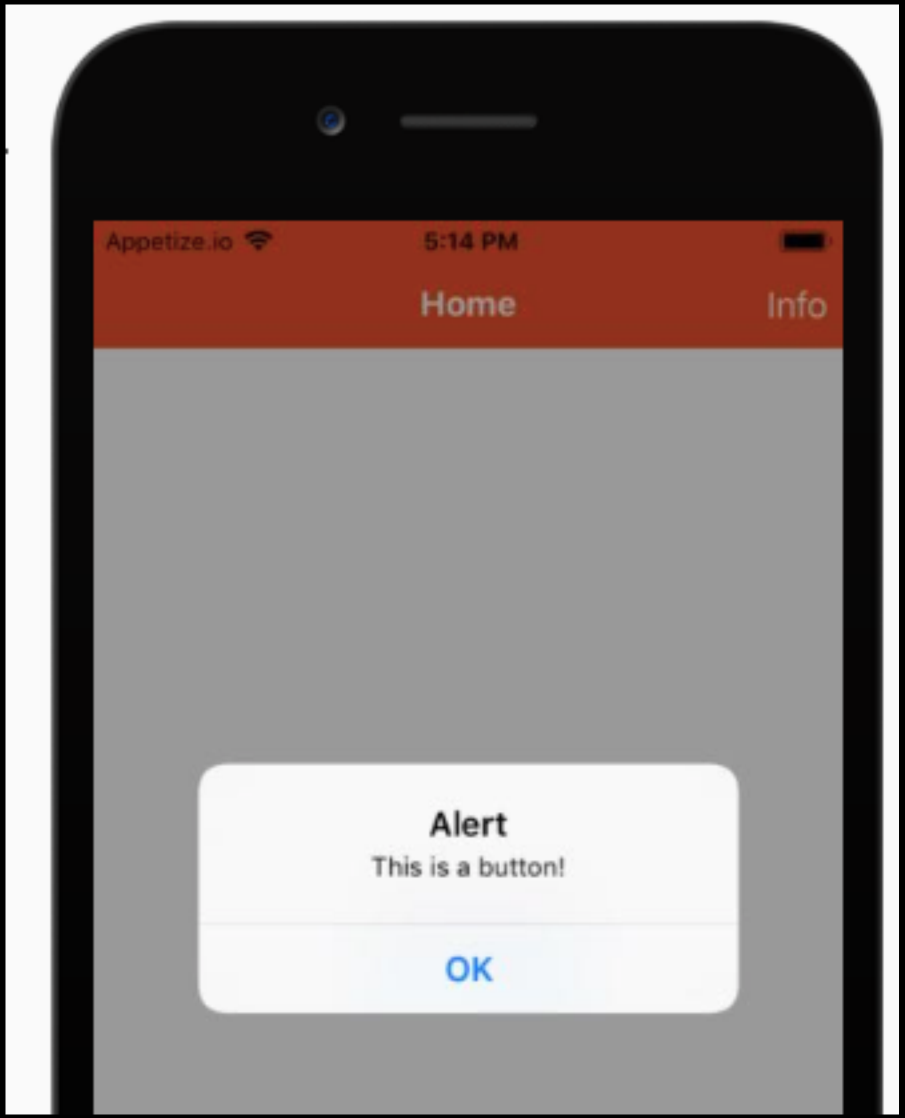
Configuring the Header

```
class HomeScreen extends React.Component {
  static navigationOptions = {
    title: 'Home',
    headerStyle: {
      backgroundColor: '#f4511e',
    },
    headerTintColor: '#fff',
    headerTitleStyle: {
      fontWeight: 'bold',
    },
  };
```



```
static navigationOptions = {
  title: 'Home',
  headerStyle: {
    backgroundColor: '#f4511e',
  },
  headerRight: (
    <Button
      onPress={() => alert('This is a button!')}
      title="Info"
      color="#fff"
    />
  ),
  headerTintColor: '#fff',
  headerTitleStyle: {
    fontWeight: 'bold',
  },
}
```

ADDED BUTTON RIGHT

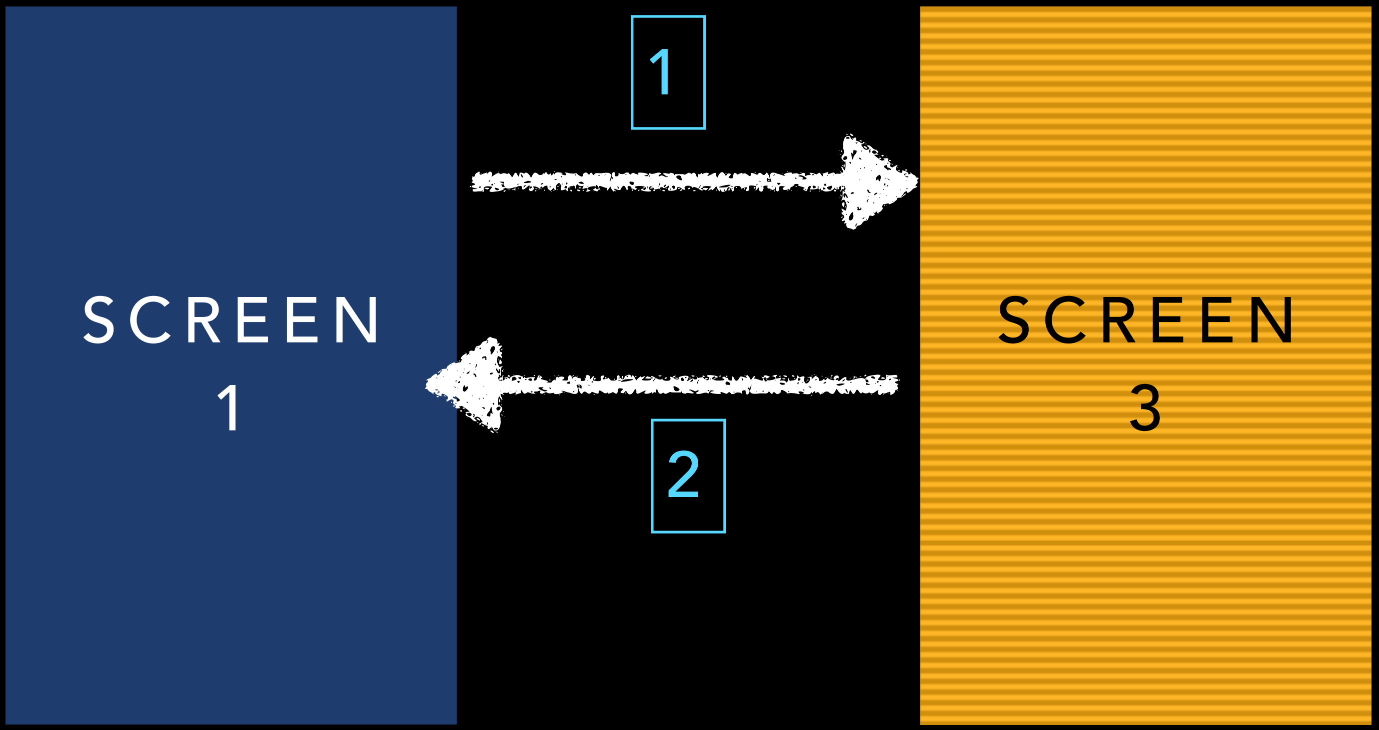


Alert
This is a button!

https://snack.expo.io/@professorxii/aGVhZG

NAVIGATION LIFE CYCLE

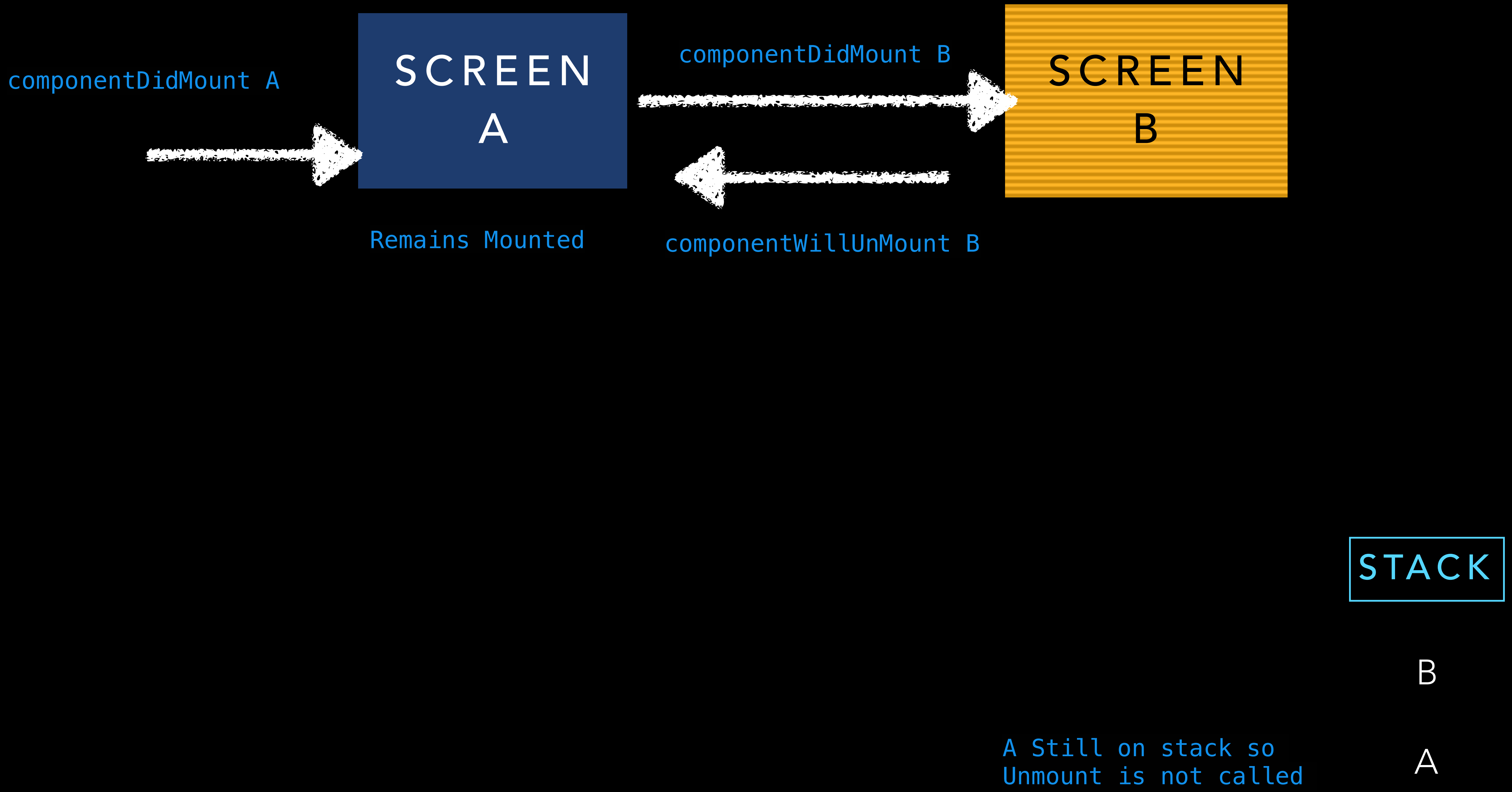
- What happens to the Home Screen when you navigate away from it?
  - Excepted behavior



1 COMPONENTWILLUNMOUNT

2 COMPONENTDIDMOUNT

HOWEVER THE BEHAVIOR IS MORE COMPLICATED


componentDidMount A
SCREEN A
Remains Mounted
componentDidMount B
SCREEN B
componentWillUnMount B
STACK
B
A
A Still on stack so Unmount is not called

- willFocus - the screen will focus
- didFocus - the screen focused (if there was a transition, the transition completed)
- willBlur - the screen will be unfocused
- didBlur - the screen unfocused (if there was a transition, the transition completed)

```javascript
const didBlurSubscription = this.props.navigation.addListener(
  'didBlur',
  payload => {
    console.debug('didBlur', payload);
  }
);

// Remove the listener when you are done
didBlurSubscription.remove();
```

PASSING PARAMS

PASS PARAMETER TO SCREEN

```
this.props.navigation.navigate('RouteName', { /* params go here */ })
```

READ PARAMETER

```
this.props.navigation.getParam(paramName, defaultValue)
```

```
class HomeScreen extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
        <Text>Home Screen</Text>
        <Button
          title="Go to Details"
          onPress={() => {
            /* 1. Navigate to the Details route with params */
            this.props.navigation.navigate('Details', {
              itemId: 86,
              otherParam: 'anything you want here',
            });
          }}
        />
      </View>
    );
  }
}
```

READING THE PROPERTIES IN THE SUBVIEW

```
/* 2. Get the param, provide a fallback value if not available */
const { navigation } = this.props;
const itemId = navigation.getParam('itemId', 'NO-ID');
const otherParam = navigation.getParam('otherParam', 'some default value');
```

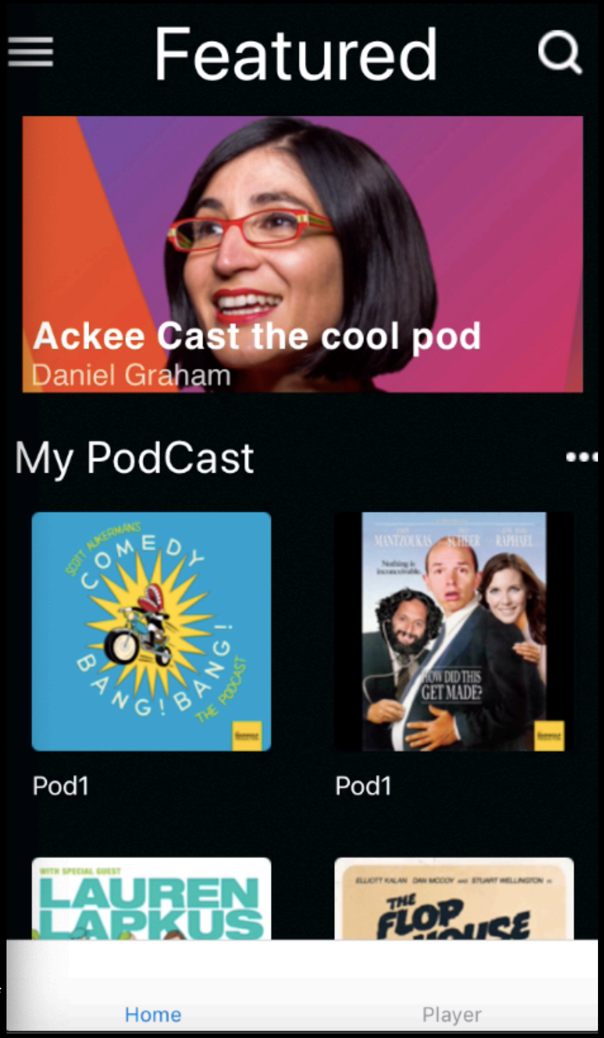
TAB NAVIGATION

HTTPS://SNACK.EXPO.IO/@REACT-NAVIGATION/BASIC-TABS-V3

UPDATE IMPORT

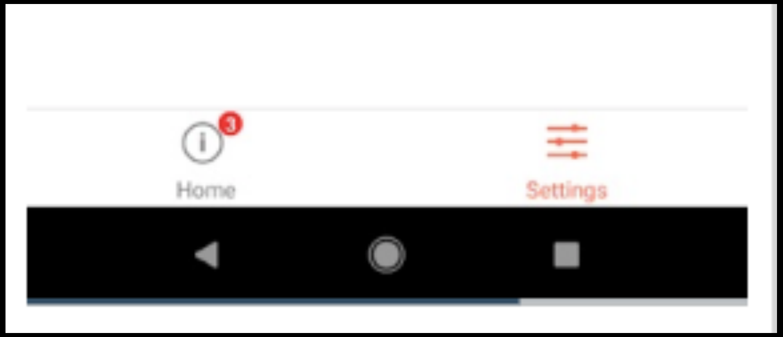
```
import React from 'react';
import {createBottomTabNavigator, createAppContainer} from 'react-navigation'
import HomeScreen from './components/HomeScreen';
import PlayerScreen from './components/PlayerScreen'



const rootStack = createBottomTabNavigator({
    Home: HomeScreen,
    Player: PlayerScreen
  },{
    initalRouteName: 'Home'
  }
)

const AppContainer = createAppContainer(rootStack)


export default class App extends React.Component {
  render() {
    return (
      <AppContainer/>
    );
  }
}
```

UPDATE TO TAB NAVIGATOR



Adding Icons to the tab navigator

```
export default createAppContainer(
  createBottomTabNavigator(
    {
      Home: { screen: HomeScreen },
      Settings: { screen: SettingsScreen },
    },
    {
      defaultNavigationOptions: ({ navigation }) => ({
        tabBarIcon: ({ focused, tintColor }) =>
          getTabBarIcon(navigation, focused, tintColor),
      }),
      tabBarOptions: {
        activeTintColor: 'tomato',
        inactiveTintColor: 'gray',
      },
    }
  )
);
```



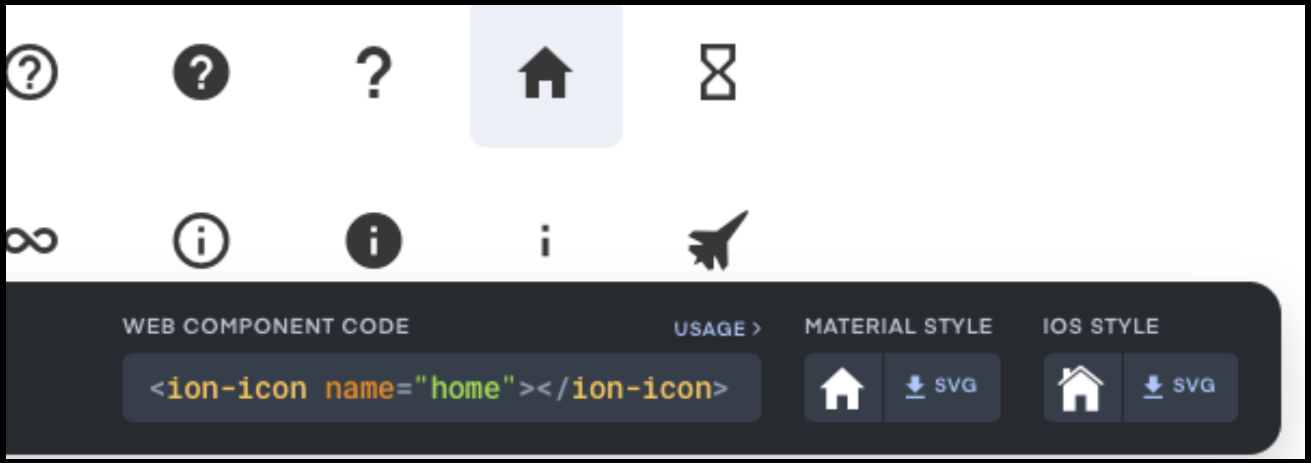
LETS EXPLORE GETTABBARICON

```
import { Ionicons } from '@expo/vector-icons'; // 6.2.2
```

https://ionicons.com

```
const getTabBarIcon = (navigation, focused, tintColor) => {
  const { routeName } = navigation.state;
  let IconComponent = Ionicons;
  let iconName;
  if (routeName === 'Home') {
    iconName = `ios-information-circle${focused ? '' : '-outline'}`;
    // We want to add badges to home tab icon
    IconComponent = HomeIconWithBadge;
  } else if (routeName === 'Settings') {
    iconName = `ios-options${focused ? '' : '-outline'}`;
  }

  // You can return any component that you like here!
  return <IconComponent name={iconName} size={25} color={tintColor} />;
};
```

$ SYNTAX IS NEW
IT JUST DOES
CONCATENATION



```
<ion-icon name="home"></ion-icon>
```

```
const HomeIconWithBadge = props => {
  // You should pass down the badgeCount in some other ways like context, redux, mobx or event emitters.
  console.log(props)
  return <IconWithBadge {...props} badgeCount={3} />;
};
```

↑

SPREAD ATTRIBUTES

ALLOWS YOU PASS A VARIABLE NUMBER OF VARIABLES

Think of the example of summing variable number of numbers for lecture 2

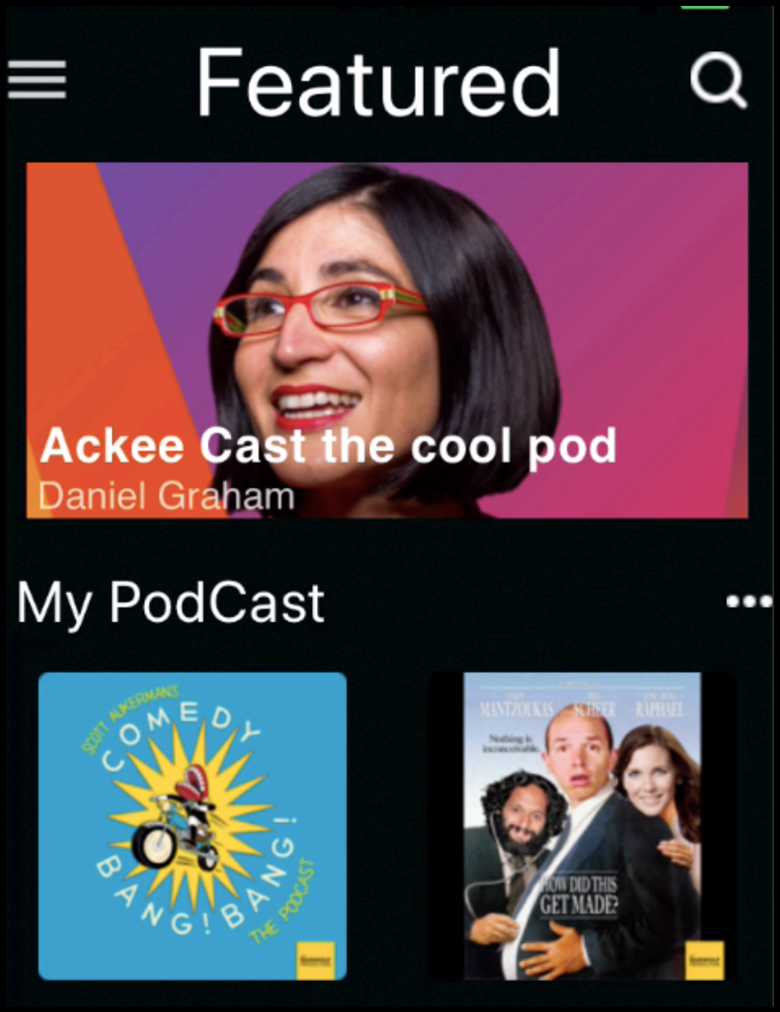
DRAW NAVIGATION

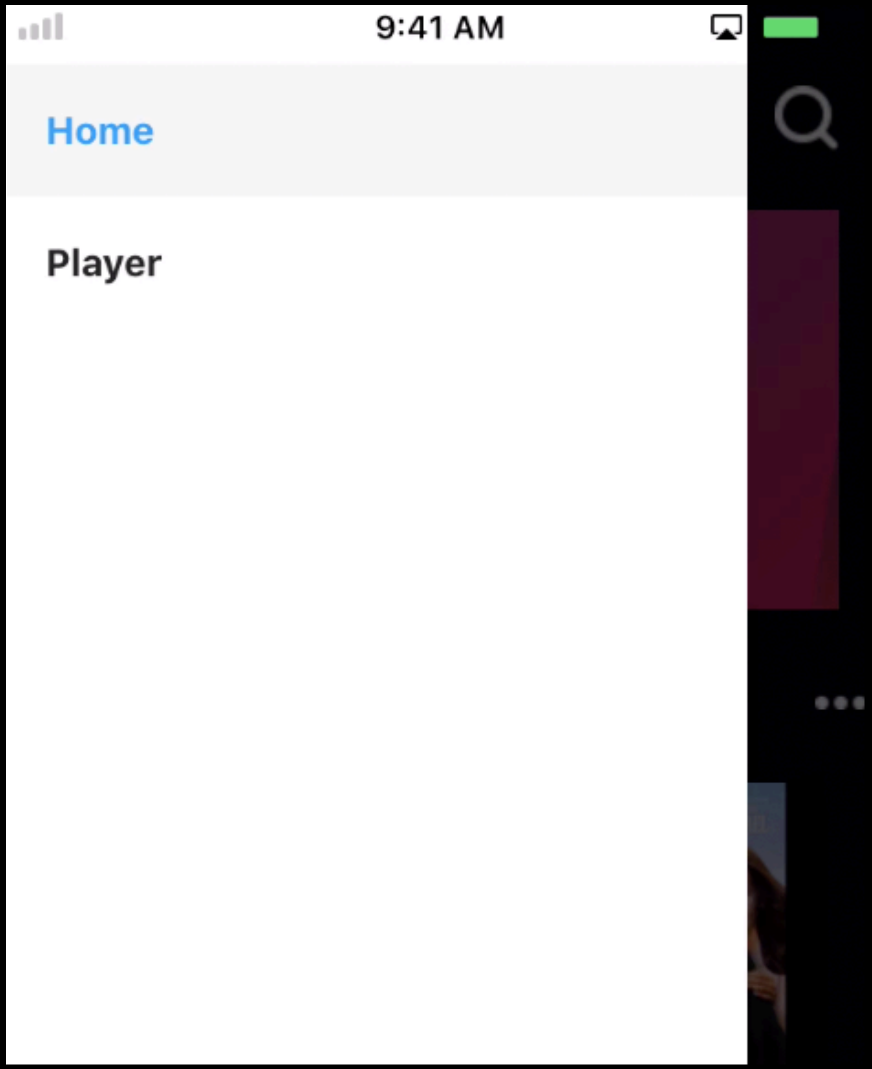
UPDATE TO DRAW NAVIGATOR

```
import React from 'react';
import {createDrawerNavigator, createAppContainer} from 'react-navigation'
import HomeScreen from './components/HomeScreen';
import PlayerScreen from './components/PlayerScreen'



const rootStack = createDrawerNavigator({
    Home: HomeScreen,
    Player: PlayerScreen
  },{
    initalRouteName: 'Home'
  }
)

const AppContainer = createAppContainer(rootStack)


export default class App extends React.Component {
  render() {
    return (
      <AppContainer/>
    );
  }
}
```

UPDATE TO DRAW NAVIGATOR

NAVIGATION DRAW





HELPER FUNCTIONS NAVIGATION DRAW

```
this.props.navigation.openDrawer();
this.props.navigation.closeDrawer();
this.props.navigation.toggleDrawer();
```

```
import * as React from 'react'
import {View, Text, StyleSheet, Image, TouchableOpacity} from 'react-native'

export default class Header extends React.Component{
    constructor(props){
        super(props)
    }

    render(){
        return(
            <View style={styles.container}>
           <TouchableOpacity onPress={()=>{this.props.navigation.toggleDrawer()}}>
                <Image style={styles.menuIcon} source={require('../assets/
menuIcon.png')}/>
            </TouchableOpacity>
                <Text style={styles.title}> Featured</Text>
                <Image style={styles.searchIcon} source={require('../assets/
searchIcon.png')} />
            </View>
        )
    }

}
```

ADDED MENU ICON
CALL TO TOGGLE DRAW

↓

NEED TO PASS NAVIGATION PROPERTY TO CHILD ELEMENTS

```
        <Header navigation={this.props.navigation}/>
```

OTHER NAVIGATION FEATURE

Persistent state. We then user open the APP
Start back where they left off

```
<AppContainer persistenceKey={"NavigationState"} />
```

Screen Can be other AppContainers:
You can nest navigators in this way

MODAL VIEW NAVIGATION

FOR MORE DETAIL ON STYLING THE NAVIGATION

https://reactnavigation.org/docs/en/headers.html