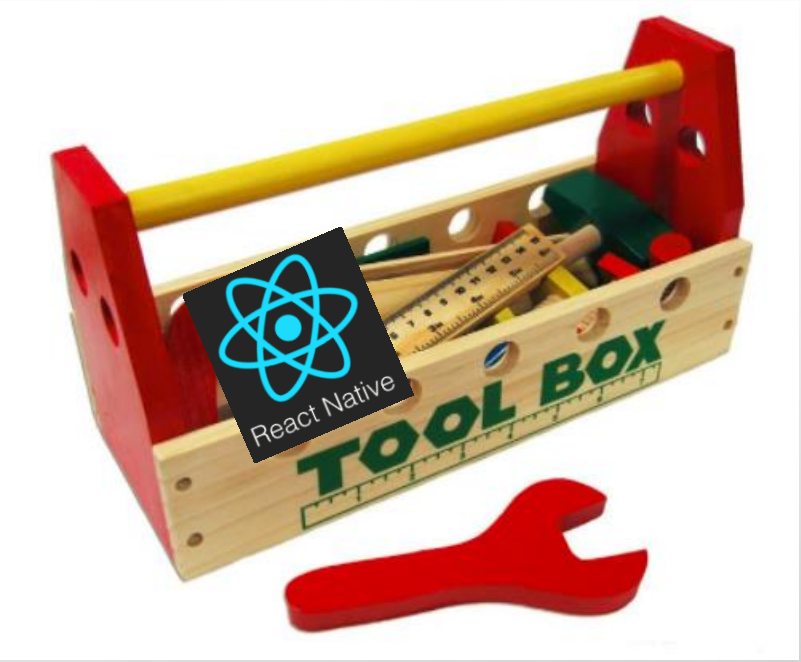


REACT NATIVE TOOLBOX

01 – 17

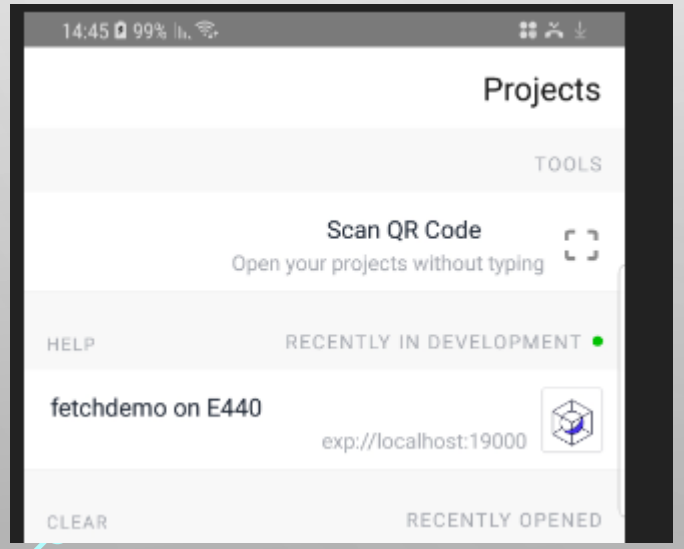
SYLLABUS

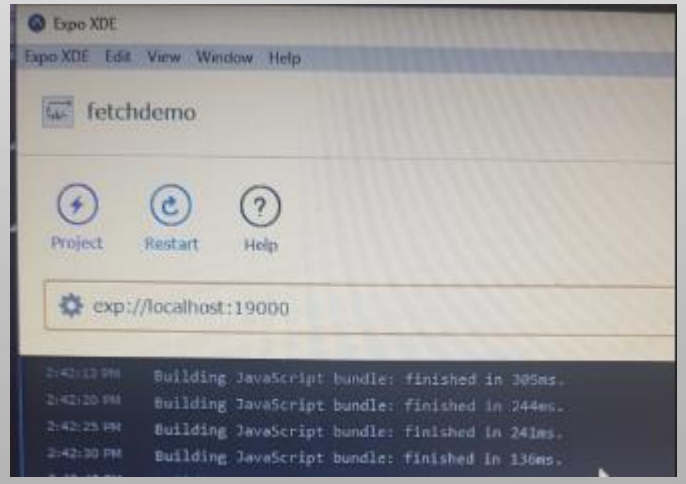
- **-01 Expo**
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

- 09 Compass
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib
- 15 Linking
- 17-1 Maps
- 17 Fingerprint authentication

EXPO

1. הכי מהיר זה לעבוד ב-LOCALHOST כאשר המכשיר וגם המחשב מחוברים על אותה רשת WIFI.
2. ניתן לחבר את המכשיר למחשב עם כבל USB ולאפשר DEBUGGING ("איתור באגים של USB") דרך הUSB ב"אפשרויות למפתחים"





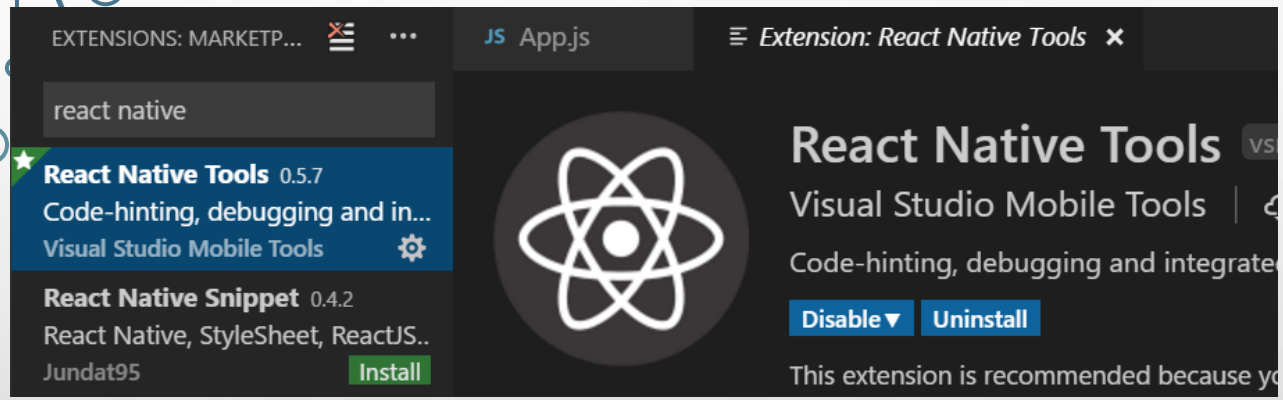
SYLLABUS

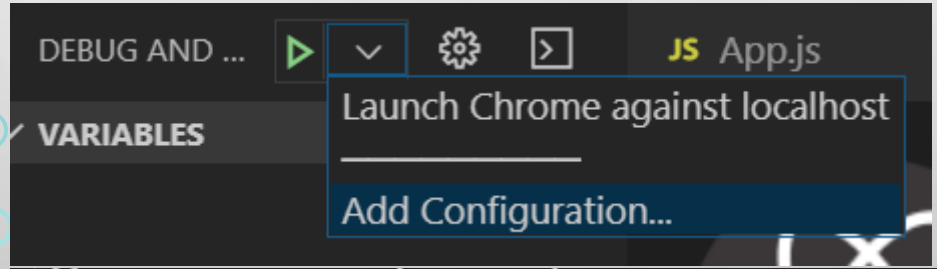
- **00 Debug**
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

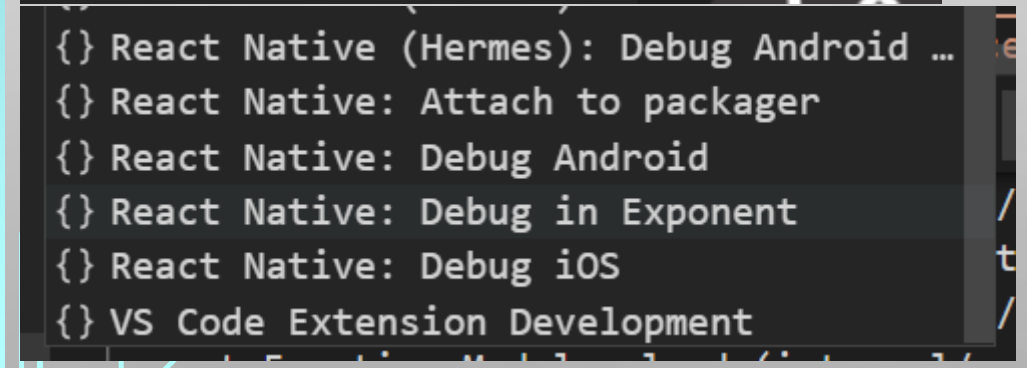
DEBUG

1. לשקשק\לנענע\לנער את המכשיר ואז לבחור בתפריט את "Debug JS Remotely"
2. יפתח הכרום בצורה אטומטית ואז יש ללחוץ F12
3. יש לבחור את הטאב של Sources
4. כאשר נגיע לBREAKPOINT הקוד יעצר בשורה המתאימה ואז אפשר לדאבג רגיל.

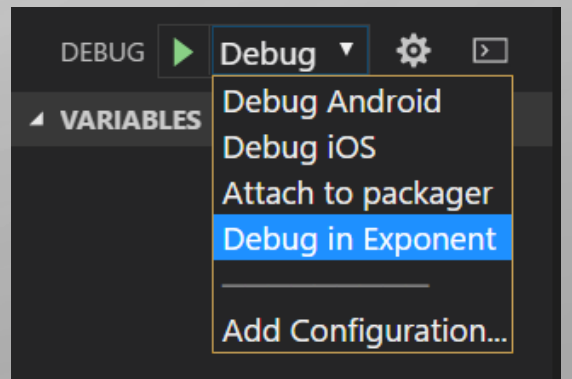
DEBUG IN VS CODE

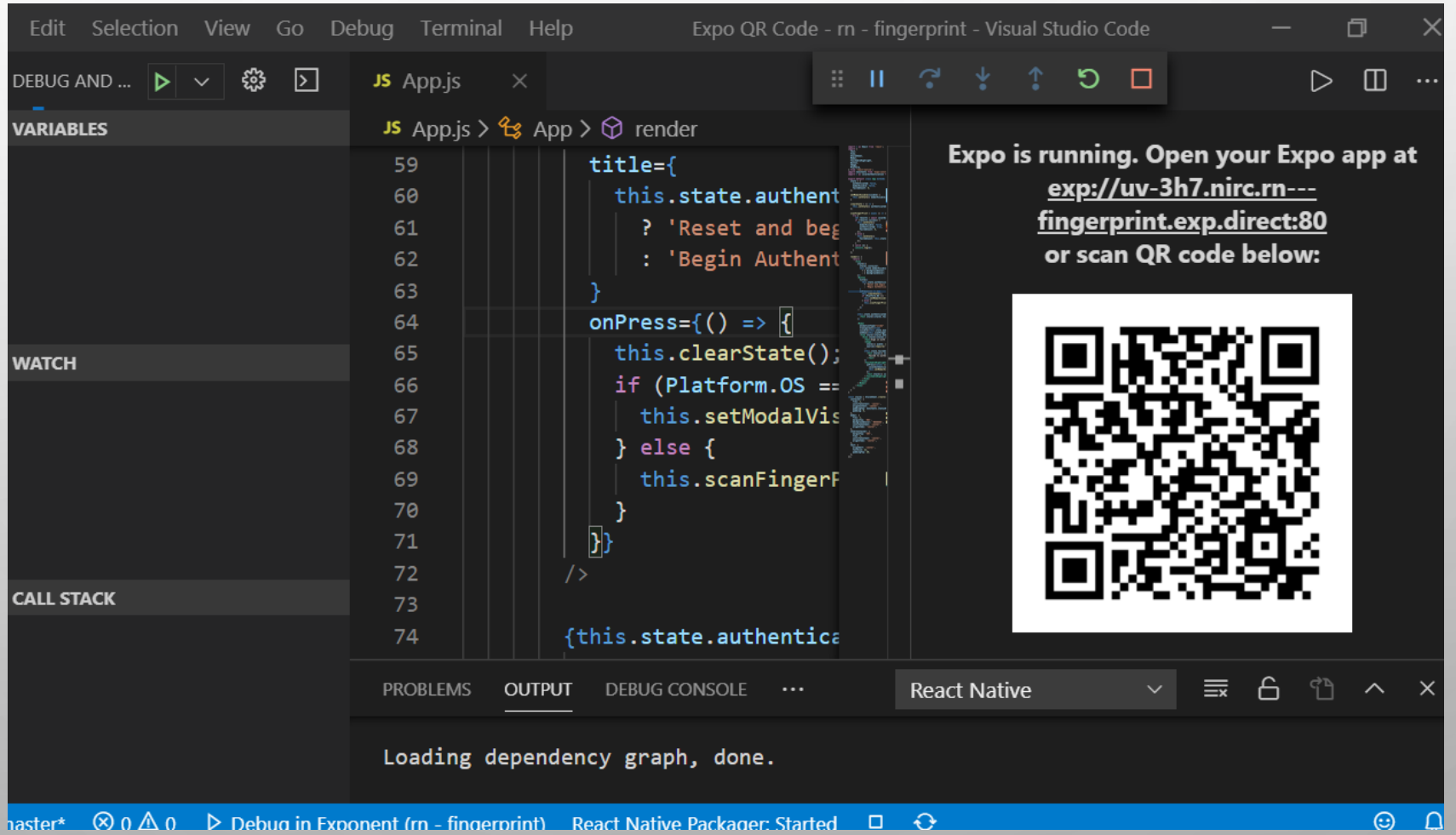






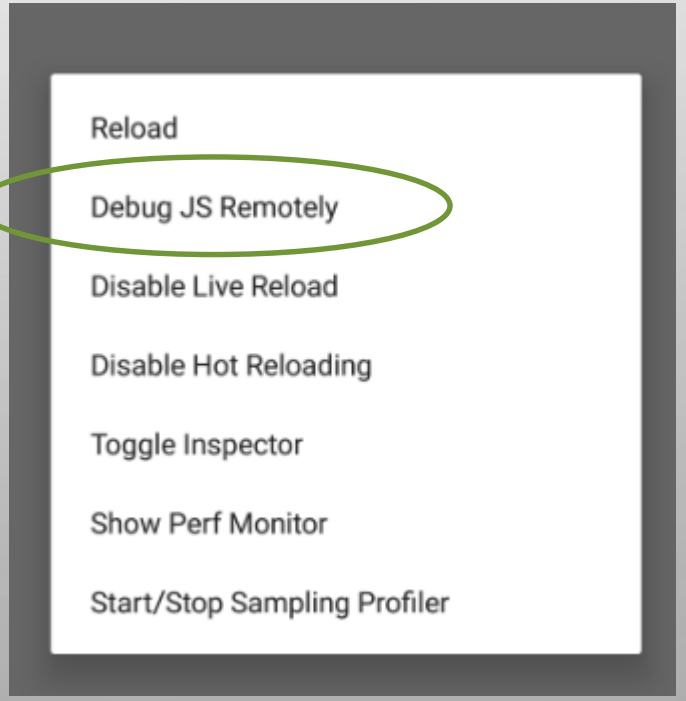
- https://github.com/Microsoft/vscode-react-native/blob/master/doc/expo.md
- ניתן להתקין תוסף שמאפשר לדאבג במקום בכרום בתוך ה- visual code עצמו.
- להתקין את התוסף react native tools
- יתכן ותתבקשו להכניס שם וסיסמא של EXPO
- לבחור




Edit Selection View Go Debug Terminal Help
Expo QR Code - rn - fingerprint - Visual Studio Code
DEBUG AND ...
App.js
VARIABLES
WATCH
CALL STACK
App.js > App > render
title={
this.state.authent
? 'Reset and beg
: 'Begin Authent
}
onPress={() => {
this.clearState();
if (Platform.OS ==
this.setModalVis
} else {
this.scanFingerP
}
}}
/>
{this.state.authentica
Expo is running. Open your Expo app at
exp://uv-3h7.nirc.rn---fingerprint.exp.direct:80
or scan QR code below:
PROBLEMS
OUTPUT
DEBUG CONSOLE
React Native
Loading dependency graph, done.
Debug in Exponent (rn - fingerprint)
React Native Packager: Started

DEBUG IN VS CODE

- לחכות למסך שייציג QR לסריקה – לוקח זמן, סבלנות☺
- ברגע שמסתיים – לשקשק את המכשיר ולבחור את האופציה " Debug JS Remotely"
- לדאבג☺



SYLLABUS

- 00 Debug
- **01 APK Creation**
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

APK CREATION

1. אם רוצים את האפליקציה על המכשיר אבל בתוך האפליקציה של EXPO ניתן רק לעשות PUBLISH .

- https://www.youtube.com/watch?v=JAkO1-F0Cgs&index=10&list=PL06z42zB6YZ_G3sjHluv6uj9bA76c9v7V&t=7s

1. אם רוצים כאפליקציה נפרדת יש לייצר APK
2. להתקין npm install exp-cli
3. להתקין npm install -g exp

APK CREATION CONT'

4. exp login
5. exp start (נתקע אצלי באמצע ועדין עובד בסוף)
6. exp build:android , בבחירה נא לבחור אופציה 1. לוקח הרבה זמן ~יותר מעשר דקות
7. ניתן לראות את הסטטוס ע"י exp build:status (אצלי לא מראה כלום) וניתן גם לראות את הסטטוס דרך האתר שלהם ע"י הלינק שמקבלים. (כן עובד אצלי)
8. אם הצליח להסתיים לוקאלית תקבלו APK, אם לא ניתן להוריד מהאתר ברגע שהסתיים.
9. נותר להתקין על הטלפון ע"י למשל התוכנה APKINSTALLER ב /http://apkinstaller.com/downloads
10. install apk on device

https://www.youtube.com/watch?v=N2qCAF0LBMY

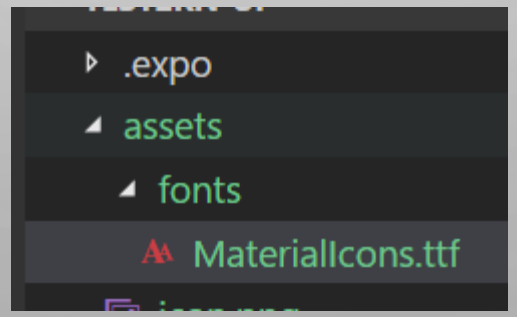
SYLLABUS

- 00 Debug
- 01 APK Creation
- **02 React Native Material UI**
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

REACT NATIVE MATERIAL UI

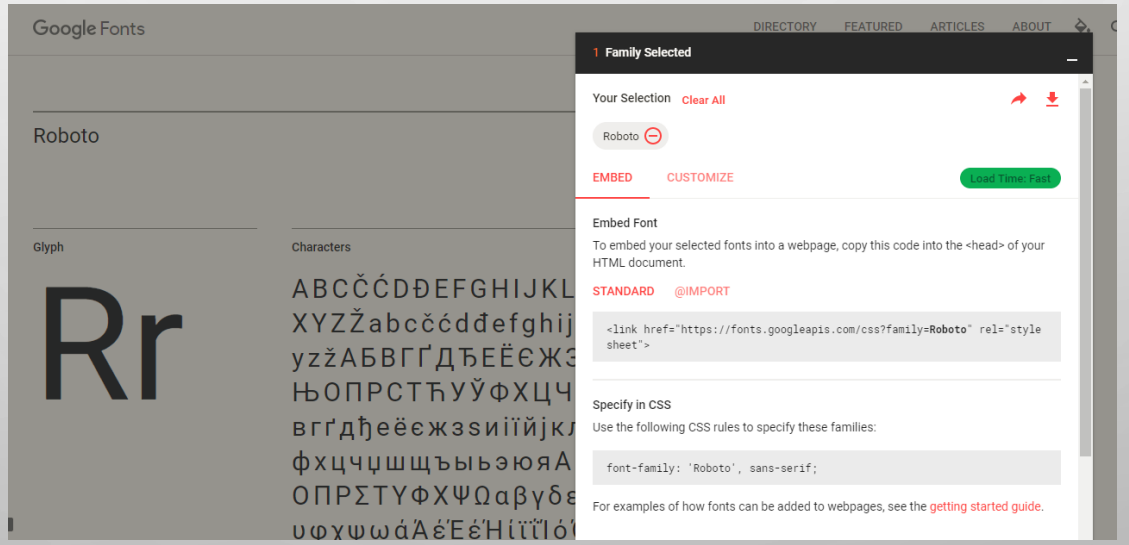
- https://www.npmjs.com/package/react-native-material-ui
- npm **install** react-native-material-ui –save
- npm i react-native-cli
- react-native link react-native-vector-icons
- copy from : ./node_modules/react-native-vector-icons/Fonts/MaterialIcons.ttf

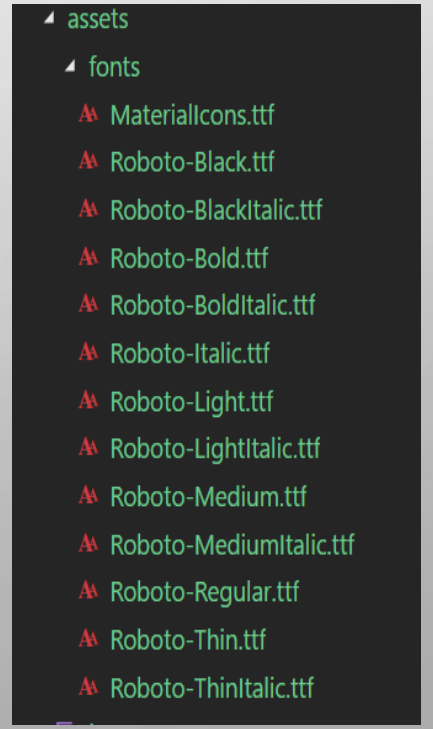
  to : assets/fonts



REACT NATIVE MATERIAL UI

- This project uses Roboto as the main font for text. Make sure to add Roboto to your project





REACT NATIVE MATERIAL UI

- Warp every thing in <ThemeProvider>
- import { Button, ThemeProvider } from 'react-native-material-ui';
- onPress

```
btnPrimaryPress(){
  alert("primary pressed!");
}

render() {
  return (
    <ThemeProvider>
    <View style={styles.container}>
      <Text>Open up App.js to start working on your app!7 {new Date().to
      <Button primary text="Primary" onPress={this.btnPrimaryPress} />
      <Button accent text="Accent" />
```

REACT NATIVE MATERIAL UI

- link to demo site: https://github.com/xotahal/react-native-material-ui-demo-app/tree/master/src

- Icons list: https://blog.foswiki.org/System/MaterialIcons

- **react-native-material-design** is similar BUT tested only for android and not for IOS!!!

SYLLABUS

- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- **03 Fetch**
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

FETCH

- נעשה אותו דבר כמו בREACT רגיל. עובד רגיל בEXPO
- דוגמאות לקריאות ל WEB SERVICE וגם לWEB API

SYLLABUS

- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- **04 Geolocation**
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

GEOLOCATION

- מתוך האתר יש ל-API כל מיני יכולות

```
navigator.geolocation.getCurrentPosition(
      (position) => {
        const output=
        'latitude=' + position.coords.latitude +
        '\nlongitude=' + position.coords.longitude +
        '\naltitude=' + position.coords.altitude +
        '\nheading=' + position.coords.heading +
        '\nspeed=' + position.coords.speed

        alert(output);
      },
      (error) => alert(error.message),
      { enableHighAccuracy: true, timeout: 20000, maximumAge: 1000 }
    );
```

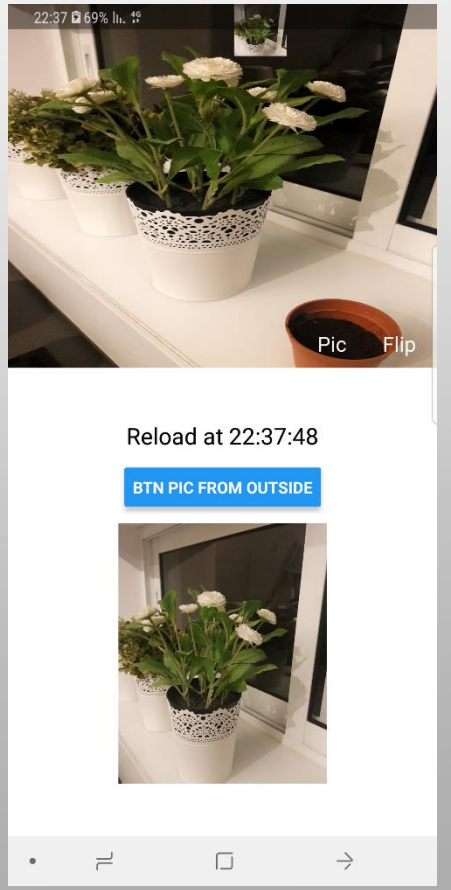
SYLLABUS

- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- **05 Camera**
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

```
import { Camera } from 'expo-camera';
...
 this.state = {
      hasCameraPermission: null,
      type: Camera.Constants.Type.back,
      picUri: 'https://facebook.github.io/react-native/docs/assets/favicon.png'
    };
...
 async componentWillMount() {
    const { status } = await Camera.requestPermissionsAsync();
    this.setState({ hasCameraPermission: status === 'granted' });
  }

  btnPic = async () => {
    debugger;
    let photo2 = await this.camera.takePictureAsync();
    //alert(photo2.uri);
    this.setState({ picUri: photo2.uri });
    Vibration.vibrate();
  }
...
onPress={() => {this.setState({
     type: this.state.type === Camera.Constants.Type.back ? Camera.Constants.Type.front : Camera.Constants.Type.back });
...
<Camera ref={ref => { this.camera = ref; }}
```

expo install expo-camera

CAMERA



SYLLABUS

- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- **06 Navigation**
- 07 Image Upload
- 08 Push Notification

NAVIGATOR

- https://reactnavigation.org/
- V5

- npm install @react-navigation/native
- expo install react-native-gesture-handler react-native-reanimated react-native-screens react-native-safe-area-context @react-native-community/masked-view
- npm install @react-navigation/stack
- npm install @react-navigation/bottom-tabs
- expo install @expo/vector-icons
- npm install @react-navigation/material-bottom-tabs react-native-paper
- npm install @react-navigation/drawer

STACK NAVIGATOR

```
import React from 'react';
import { View, Text } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';

import FirstPage from './Pages/FirstPage';
import SecondPage from './Pages/SecondPage';
import TabbedPageNavigator from './Pages/TabbedPage';
import MaterialTabbedPage  from './Pages/MaterialTabbedPage';

const Stack = createStackNavigator();
function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator initialRouteName="FirstPage">
        <Stack.Screen name="FirstPage" component={FirstPage} />
        <Stack.Screen name="SecondPage" component={SecondPage} />
        <Stack.Screen name="TabbedPageNavigator" component={TabbedPageNavigator} />
        <Stack.Screen name="MaterialTabbedPage" component={
      </Stack.Navigator>
    </NavigationContainer>
  );
}
export default App;
```



```
<TouchableOpacity onPress={() => {
          this.props.navigation.navigate('SecondPage');
        }}>
  <Text style={{ ...}} >
        Goto Second Page!</Text>
</TouchableOpacity>
```

```
import { createMaterialBottomTabNavigator } from '@react-navigation/material-bottom-tabs';
import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';
…
const Tab = createMaterialBottomTabNavigator();

function MaterialTabbedPageNavigator() {
  return (
    <Tab.Navigator
      initialRouteName="TabbedSecondAlternatePage"
      activeColor="#55ff00"
      inactiveColor='black'
      barStyle={{ backgroundColor: '#694fad' }}
    >
      <Tab.Screen
        name="TabbedAlternatePage"
        component={TabbedAlternatePage}
        options={{
          tabBarLabel: 'Alternate',
          tabBarIcon: ({ color }) => (
            <MaterialCommunityIcons name="bell" color={color} size={26} />
          ),
        }}
      />
      <Tab.Screen…

    </Tab.Navigator>
  );
}
```

MATERIAL BOTTOM TABBED NAVIGATOR

There is also Material**Top**TabNavigator

TABBED NAVIGATOR

```
import MaterialCommunityIcons from 'react-native-vector-icons/MaterialCommunityIcons';
import { createBottomTabNavigator } from '@react-navigation/bottom-tabs';
…
const Tab = createBottomTabNavigator();
export default function TabbedPage() {
  return (
    <Tab.Navigator
      initialRouteName="Feed"
      tabBarOptions={{
        //activeTintColor: '#e91e63',
        activeTintColor: 'purple',
      }}
    >
      <Tab.Screen
        name="TabbedAlternatePage"
        component={TabbedAlternatePage}
        options={{
          tabBarLabel: 'Alternate Page',
          tabBarIcon: ({ color, size }) => (
            <MaterialCommunityIcons name="home" color={color} size={size} />
          ),
        }}
      />
      <Tab.Screen…

    </Tab.Navigator>
  );
}
```

DRAWER NAVIGATOR

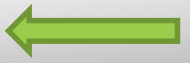
```
import { createDrawerNavigator } from '@react-navigation/drawer';
const Drawer = createDrawerNavigator();

function MyDrawer() {
  return (
    <Drawer.Navigator initialRouteName="FirstPage">
      <Drawer.Screen
        name="FirstPage"
        component={FirstPage}
        options={{ drawerLabel: 'FirstPage' }}
      />
      <Drawer.Screen…
    </Drawer.Navigator>
  );
}

//pay attention that the stack navigator screens must math the drawer screens!
const Stack = createStackNavigator();
function App() {
  return (
      <NavigationContainer>
        <MyDrawer>
          <Stack.Navigator initialRouteName="FirstPage">
            <Stack.Screen name="FirstPage" component={FirstPage} />
…
            <Stack.Screen name="MaterialTabbedPageNavigator" component={MaterialTabbedPageNavigator} />
          </Stack.Navigator>
        </MyDrawer>
      </NavigationContainer>
```

SEND INFO BETWEEN PAGES PARAMS

```
this.props.navigation.navigate('SecondPage', { user: 'Lucy' + new Date().getSeconds() });
```

- Sending

```
this.props.route.params != undefined ?
        this.props.route.params.user : '...'
```



- recieving

NAVIGATION EVENTS – REFRESH PAGE THOUGH NAVIGATION

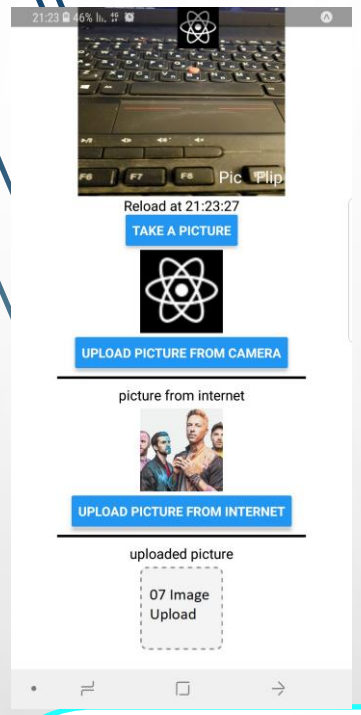
```
componentDidMount() {
  this._unsubscribeFocus = this.props.navigation.addListener('focus', (payload) => {
    console.log('will focus', payload);
    this.setState({stam:'stam'});
  });
  this._unsubscribeBlur = this.props.navigation.addListener('blur', (payload) => {
    console.log('will blur', payload)
  });
}

componentWillUnmount() {
  this._unsubscribeFocus();
  this._unsubscribeBlur();
}
```

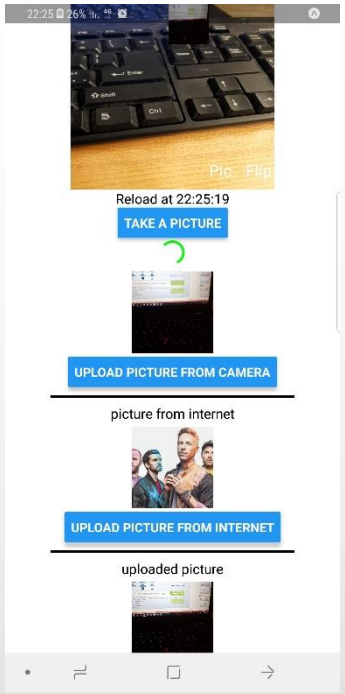
בגלל ש האירועים הללו רצים פעם אחת אין בעיה לקרוא ל setstate בפנים. הוא יקרא רק פעם אחת.

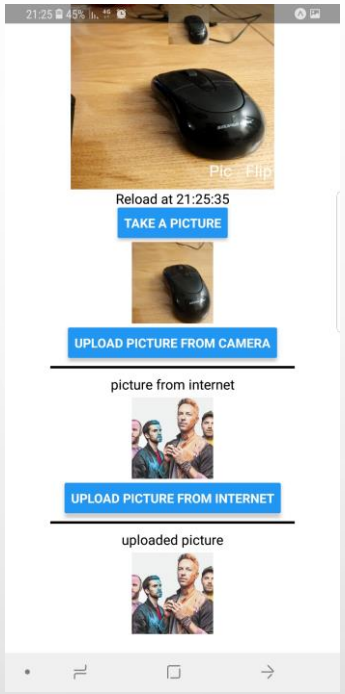
- בכדי לאפשר הרצת קוד במעבר בין עמודים.
- נוכל לדעת מאיפה הגענו ועם איזה מידע ולהריץ למשל setState

SYLLABUS

- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- **07 Image Upload**
- 08 Push Notification







WEB SERVICE

IMAGE UPLOAD – FETCH, CLIENT SIDE

- תמונה מהמצלמה: נצלם תמונה כרצף של ביטים בבסיס 64 ונשלח אותו לשרת כמחרוזת ארוכה בליווי השם של התמונה.
- תמונה מהאינטרנט: נשלח לשרת את הURL ואת השם של התמונה והשרת ב C# יוריד את התמונה וישמור אותה אצלו.

```
let photo = await this.camera.takePictureAsync({
    quality: 0.1,
    base64: true,
  });
  this.setState({
    pic64base: photo.base64,
    picName64base: 'image1_' + new Date().getTime() + '.jpg',
    picUri: `${photo.base64}`,
  });
```

להקטין את גודל התמונה שצולמה. 1 מקסימום 0 מינימום

חשוב!!! בRN כאשר מזמנים תמונה היא נשמרת תחת השם שלה ב CACHE ולכן אם מזמנים תמונה עם אותו שם תופיע התמונה הראשונה שוב פעם. לכן כאשר נעלה את התמונה לשרת נצטרך ליצר לה שם חדש בכל פעם. פה אני מייצר את השם החדש. השם החדש מורכב מתחילית אשר ביקשנו ועוד מספר שנוצר מהתיקים של המחשב.

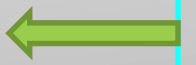
WEB SERVICE

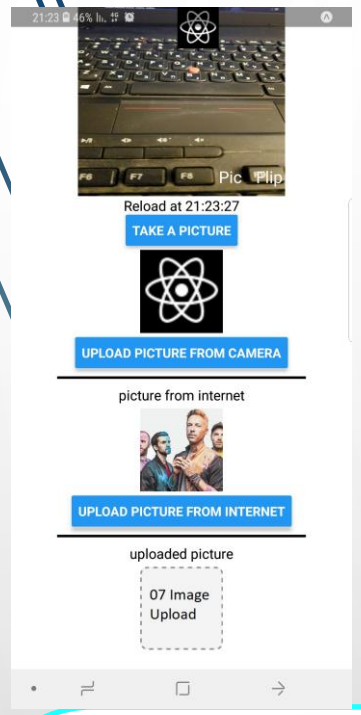
IMAGE UPLOAD – FETCH, CLIENT SIDE

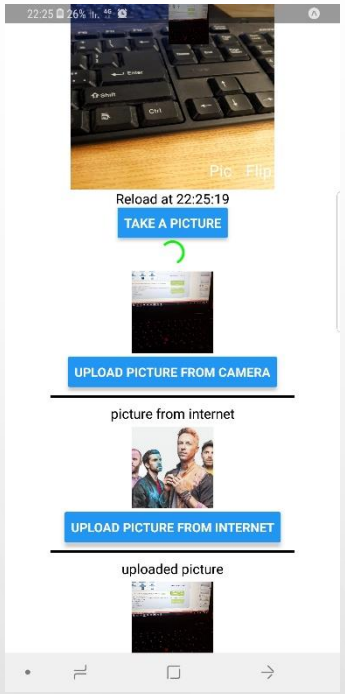
```
uploadBase64ToASMX = () => {
   this.setState({ animate: true });
   let urlAPI =
     'http://ruppinmobile.tempdomain.co.il/site01/webservice.asmx/ImgUpload';

   fetch(urlAPI, {
     method: 'POST',
     body: JSON.stringify({
       base64img: this.state.pic64base,
       base64imgName: this.state.picName64base,
     }),
```

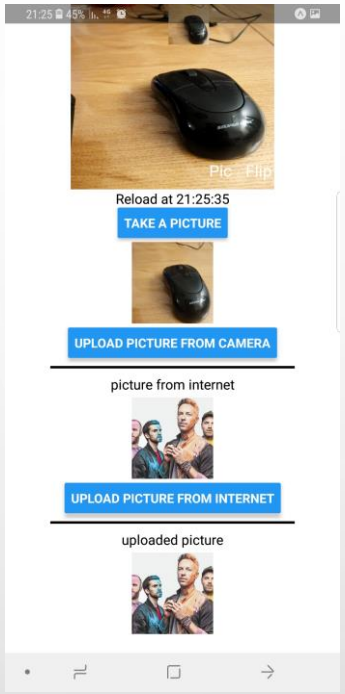
WEB SERVICE
IMAGE UPLOAD – C#, SERVER SIDE

```
[WebMethod]
    public string ImgUpload(string base64img, string base64imgName)
    {
        //for example - pay attention the first '/' is part of the image!
        //
        //File.AppendAllText(Server.MapPath("images/file1.txt"), base64imgName + "\r\n");
        File.WriteAllBytes(Server.MapPath("images/"+ base64imgName), Convert.FromBase64String(base64img));

        return new JavaScriptSerializer().Serialize(new { res = "OK" });
    }
```









WEB API

IMAGE UPLOAD – FETCH, CLIENT SIDE

```
...
let photo = await this.camera.takePictureAsync({quality : 0.7});
...
 imageUpload = (imgUri, picName) => {
    let urlAPI = "http://.../site01/uploadpicture";
    let data1 = new FormData();
    data1.append('picture', {
      uri: imgUri,
      name: picName,
      type: 'image/jpg'
    });
```

להקטין את גודל התמונה שצולמה. 1 מקסימום 0 מינימום

מיקום הקוד של צד השרת

IMAGE UPLOAD – FETCH, CLIENT SIDE

```
...
    const config = {
      method: 'POST',
      body: datal,
    }

    fetch(urlAPI, config)
      .then((res) => {
        if (res.status == 201) {return res.json(); }
        else {return "err"; }
      })
      .then((responseData) => {
        if (responseData != "err") {
          let picNameWOExt = picName.substring(0, picName.indexOf("."));
          let imageNameWithGUID = responseData.substring(responseData.indexOf(picNameWOExt),
responseData.indexOf(".jpg") + 4);
          this.setState({
            uplodedPicUri: { uri: this.uplodedPicPath + imageNameWithGUID },
          });
          console.log("img uploaded successfully!");
        }
        else {alert('error uploding ...');  }
      })
      .catch(err => {alert('err upload= ' + err); });
```

חשוב!!! בRN כאשר מזמנים תמונה היא נשמרת תחת השם שלה ב CACHE ולכן אם מזמנים תמונה עם אותו שם תופיע התמונה הראשונה שוב פעם. לכן כאשר נעלה את התמונה לשרת נצטרך ליצר לה שם חדש בכל פעם. זאת ניתן לעשות בצד השרת. ולקבל את השם החדש לצד הלקוח. פה אני מחלץ את השם החדש. השם החדש מורכב מתחילית אשר ביקשנו כאשר שלחנו את התמונה לשרת ועוד GUID שנוצר בשרת

IMAGE UPLOAD – C# WEB API, SERVER SIDE

```
[EnableCors(origins: "*", headers: "*", methods: "*")]
public class ValuesController : ApiController
{
...
[Route("uploadpicture")]
public Task<HttpResponseMessage> Post()
{
    string outputForNir="start---";
    List<string> savedFilePath = new List<string>();
    if (!Request.Content.IsMimeMultipartContent())
    {
        throw new HttpResponseException(HttpStatusCode.UnsupportedMediaType);
    }
    string rootPath = HttpContext.Current.Server.MapPath("~/uploadFiles");
    var provider = new MultipartFileStreamProvider(rootPath);
    var task = Request.Content.ReadAsMultipartAsync(provider).
        ContinueWith<HttpResponseMessage>(t =>
        {
            if (t.IsCanceled || t.IsFaulted)
            {
                Request.CreateErrorResponse(HttpStatusCode.InternalServerError, t.Exception);
            }
```

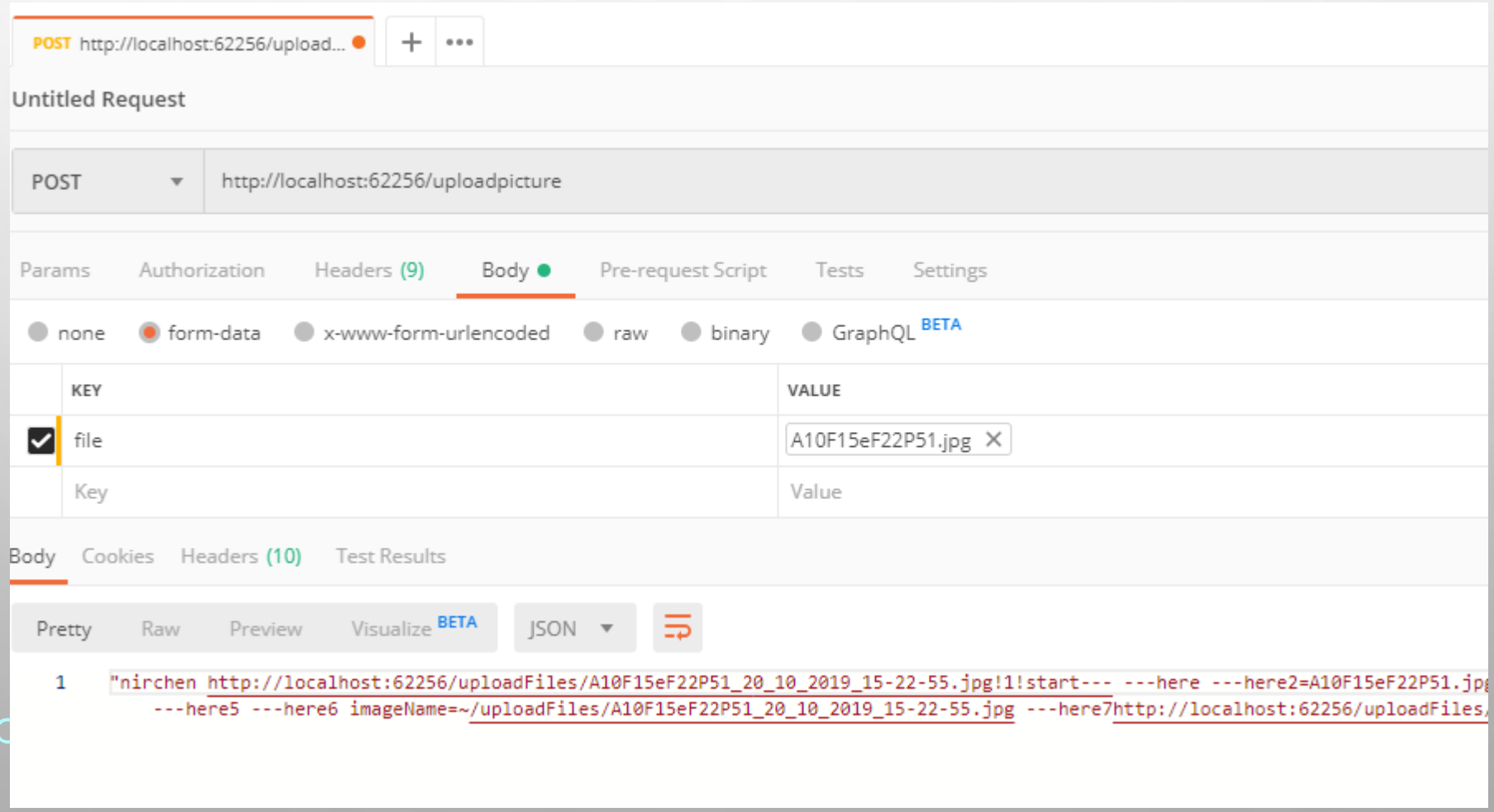
IMAGE UPLOAD – C# WEB API, SERVER SIDE

```
foreach (MultipartFileData item in provider.FileData)
{
    try
    {
        outputForNir += " ---here";
        string name = item.Headers.ContentDisposition.FileName.Replace("\"", "");
        outputForNir += " ---here2=" + name;

        //need the guid because in react native in order to refresh an inamge it has to have a new name
        string newFileName = Path.GetFileNameWithoutExtension(name) +"_"+  Guid.NewGuid() +
Path.GetExtension(name);
        //string newFileName = name + "" + Guid.NewGuid();
        outputForNir += " ---here3" + newFileName;

        //delete all files begining with the same name
        string[] names = Directory.GetFiles(rootPath);
        foreach (var fileName in names)
        {
            if (Path.GetFileNameWithoutExtension(fileName).IndexOf(Path.GetFileNameWithoutExtension(name)) != -
1)
            {
                File.Delete(fileName);
            }
        }

        //File.Move(item.LocalFileName, Path.Combine(rootPath, newFileName));
        File.Copy(item.LocalFileName, Path.Combine(rootPath, newFileName), true);
        File.Delete(item.LocalFileName);
        outputForNir += " ---here4";
```

IMAGE UPLOAD – C# WEB API, SERVER SIDE

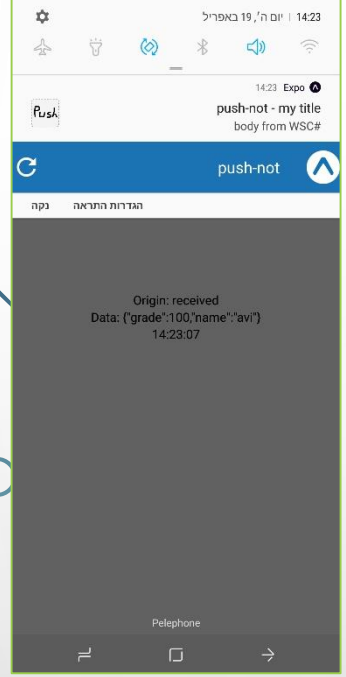
```
 Uri baseuri = new Uri(Request.RequestUri.AbsoluteUri.Replace(Request.RequestUri.PathAndQuery, string.Empty));
                    outputForNir += " ---here5" ;
                    string fileRelativePath = "~/uploadFiles/" + newFileName;
                    outputForNir += " ---here6 imageName=" + fileRelativePath;
                    Uri fileFullPath = new Uri(baseuri, VirtualPathUtility.ToAbsolute(fileRelativePath));
                    outputForNir += " ---here7" + fileFullPath.ToString();
                    savedFilePath.Add(fileFullPath.ToString());
                }
                catch (Exception ex)
                {
                    outputForNir += " ---excption="+ ex.Message;
                    string message = ex.Message;
                }
            }

            return Request.CreateResponse(HttpStatusCode.Created, "nirchen " + savedFilePath[0] + "!" +
provider.FileData.Count + "!" + outputForNir + ":)");
        });
    return task;
}
```

IMAGE UPLOAD – C# WEB API, SERVER SIDE TESTING



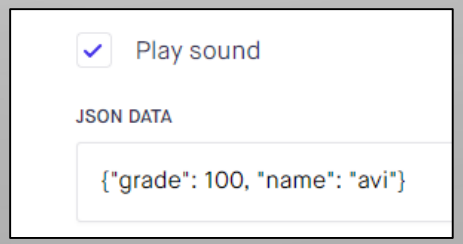
SYLLABUS

- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- **08 Push Notification**



PUSH NOTIFICATION

- נעשה שימוש בספריה של EXPO בכדי לקבל יכולות של PN עבור ANDROID ו-IOS באותו קוד. כמו כן לא צריך לייצר משתמש של APPLE.
- https://docs.expo.io/versions/latest/guides/push-notifications
- שימו לב שצריך לאפשר את ההתראות במכשיר. בחלק ממכשירי ה- ANDROID לא הצלחתי לאפשר זאת ויכול להיות שצריך לחפש\"לשחק" הכן מאפשרים את ההתראות.
- אין צורך לפתוח חשבון בשירות ענן כלשהוא. EXPO כבר מבצעים הכל עבורינו.
- בכדי לשלוח הודעה כל מה שצריך הוא לשלוח הודעת POST לשרת של EXPO שמעביר את ההודעה לשירות ענן בכדי לשלוח PN.
- יש מערכת ששולחת הודעות ישירות מהאתר של EXPO בכדי לבחון את הקוד בצד לקוח https://expo.io/dashboard/notifications



חשוב!!!
חייבים להיות מחוברים לEXPO גם בטלפון עצמו!!!!
וגם בvisual code ע"י expo login והכנסת שם וסיסמה!!!

PN – CLIENT SIDE REGISTRATION

```
import { Notifications } from 'expo';
import * as Permissions from 'expo-permissions';
export default async function registerForPushNotificationsAsync() {
    const { status: existingStatus } = await Permissions.getAsync(
        Permissions.NOTIFICATIONS
    );
    let finalStatus = existingStatus;

    // only ask if permissions have not already been determined, because iOS won't necessarily prompt the user a second time.
    if (existingStatus !== 'granted') {
        // Android remote notification permissions are granted during the app install, so this will only ask on iOS
        const { status } = await Permissions.askAsync(Permissions.NOTIFICATIONS);
        finalStatus = status;
    }

    // Stop here if the user did not grant permissions
    if (finalStatus !== 'granted') {
        return;
    }

    // Get the token that uniquely identifies this device
    let token = await Notifications.getExpoPushTokenAsync();
    //alert(token);
    // POST the token to your backend server from where you can retrieve it to send push notifications.
    return (
        token
    );
}
```

- פה צריך לדאוג ל:
  - קבלת הרשאה ליכולת קבלת התראות
  - קבלת מספר ייחודי מזהה של מכשיר הטלפון

PN – CLIENT SIDE RECEIVING MSG

```
...
import { Notifications } from 'expo';
import registerForPushNotificationsAsync from './registerForPushNotificationsAsync';

export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      notification: {},
    };
  }

  componentDidMount() {
      registerForPushNotificationsAsync()
        .then((token) => {
            this.setState({ token });
      this._notificationSubscription = Notifications.addListener(this._handleNotification);
  }

  _handleNotification = (notification) => {
    this.setState({ notification: notification });
  };

  render() {
    return (
      <Text>Origin: {this.state.notification.origin}</Text>
      <Text>Data: {JSON.stringify(this.state.notification.data)}</Text>
...
```


- פה נעשה רישום של פונקציה לקבלת התראה
- יש אפשרות לראות בהתראה עצמה בין היתר TITLE, BODY,
- באייקון - BADGE
- באפליקציה - DATA

PN – CLIENT SIDE SENDING MSG

```
btnPushFromClient = () => {
  let per = {
    to: this.state.token,
    title: 'title from client',
    body: "body from client side",
    badge: 3,
    data: { name: "nir", seconds: new Date().getSeconds() }
  };

  // POST adds a random id to the object sent
  fetch('https://exp.host/--/api/v2/push/send', {
    method: 'POST',
    body: JSON.stringify(per),
    headers: {
      "Content-type": "application/json; charset=UTF-8"
    }
  })
    .then(response => response.json())
    .then(json != null) {
        console.log(`
            returned from server\n
            json.data= ${JSON.stringify(json.data)}`);
      } else {
        alert('err json => {
      if (json');
      }
    });
}
```

PN – SERVER SIDE SENDING MSG C# WEB API

```
[Route("sendpushnotification")]
public string Post([FromBody]PushNotData pnd)
{
    // Create a request using a URL that can receive a post.
    WebRequest request = WebRequest.Create("https://exp.host/--/api/v2/push/send");
    // Set the Method property of the request to POST.
    request.Method = "POST";
    // Create POST data and convert it to a byte array.
    var objectToSend = new
    {
        to = pnd.to,
        title = pnd.title,
        body = pnd.body,
        badge = pnd.badge,
        data = pnd.data//new { name = "nir", grade = 100 }
    };

    string postData = new JavaScriptSerializer().Serialize(objectToSend);

    byte[] byteArray = Encoding.UTF8.GetBytes(postData);
    // Set the ContentType property of the WebRequest.
    request.ContentType = "application/json";
    // Set the ContentLength property of the WebRequest.
    request.ContentLength = byteArray.Length;
    // Get the request stream.
    Stream dataStream = request.GetRequestStream();
```


- כל מה שנותר זה רק לקרוא לפונקציה הזו או מהשרת או מהלקוח ע"י fetch

PN – SERVER SIDE SENDING MSG C# WEB API – CONT'

```
// Write the data to the request stream.
    dataStream.Write(byteArray, 0, byteArray.Length);
    // Close the Stream object.
    dataStream.Close();
    // Get the response.
    WebResponse response = request.GetResponse();
    // Display the status.
    string returnStatus = ((HttpWebResponse)response).StatusDescription;
    //Console.WriteLine(((HttpWebResponse)response).StatusDescription);
    // Get the stream containing content returned by the server.
    dataStream = response.GetResponseStream();
    // Open the stream using a StreamReader for easy access.
    StreamReader reader = new StreamReader(dataStream);
    // Read the content.
    string responseFromServer = reader.ReadToEnd();
    // Display the content.
    //Console.WriteLine(responseFromServer);
    // Clean up the streams.
    reader.Close();
    dataStream.Close();
    response.Close();

    return "success:) --- " + responseFromServer + ", " + returnStatus;
}
```

SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

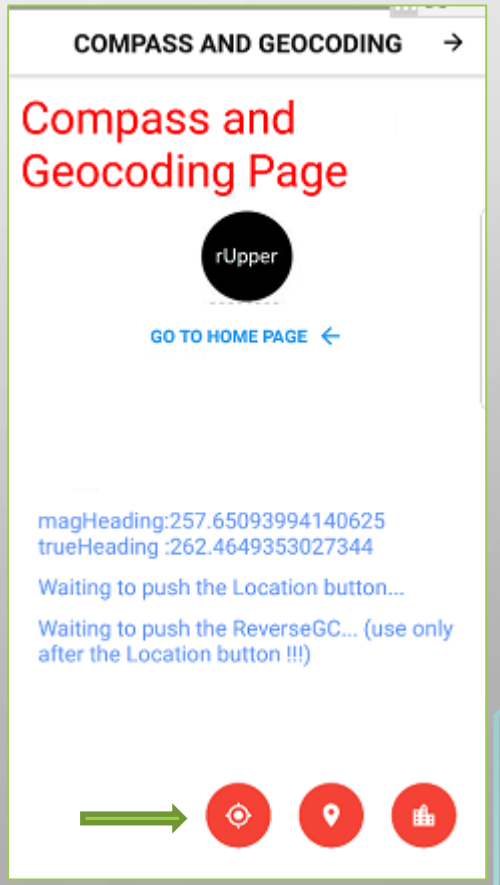
- **09 Compass**
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib

COMPASS

- expo install expo-location
- ניתן לקבל מידע לקבל הכיוון שהטלפון מופנה אליו.
- כיוון אמיתי וכיוון מגנטי

```
import * as Location from 'expo-location';
...
btnHeading = async () => {

    let { status } = await Location.requestPermissionsAsync();
    if (status !== 'granted') {
      alert('Permission to access location was denied');
    }

    let heading = await Location.getHeadingAsync({});
    this.setState({ heading });
  };
```

magHeading:275.7325439453125
trueHeading :280.5465393066406



SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

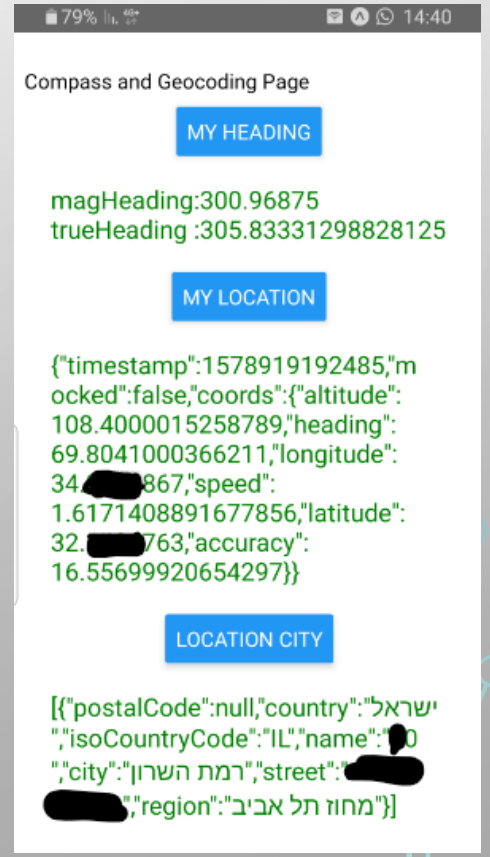
- 09 Compass
- **10 Geocoding**
- 11 Facebook
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib

GEOCODING

- expo install expo-location
- ניתן לקבל מידע אודות כתובת עבור נ.צ. קיים.
- ניתן לקבל גם הפוך - נ.צ. עבור כתובת (לא בדוגמה הזו)

```
import * as Location from 'expo-location';
...
btnLocation = async () => {
    let { status } = await Location.requestPermissionsAsync();
    if (status !== 'granted') {
      this.setState({errorMessage: 'Permission to access location was denied',  });
    }
    let location = await Location.getCurrentPositionAsync({});
    this.setState({ location });
};
btnReverseGC = async () => {
    let { status } = await Location.requestPermissionsAsync();
    if (status !== 'granted') {
      this.setState({ errorMessage: 'Permission to access location was denied', });
    }
    if (this.state.location) {
      let reverseGC = await Location.reverseGeocodeAsync(this.state.location.coords);
      this.setState({ reverseGC });
    }else{
      alert('You must push the Location button first in order to get the loc
can get the reverse geocode for the latitude and longitude!');
    }
};
```


מחקתי מהדוגמה את הפרטים של הנ.צ. והכתובת



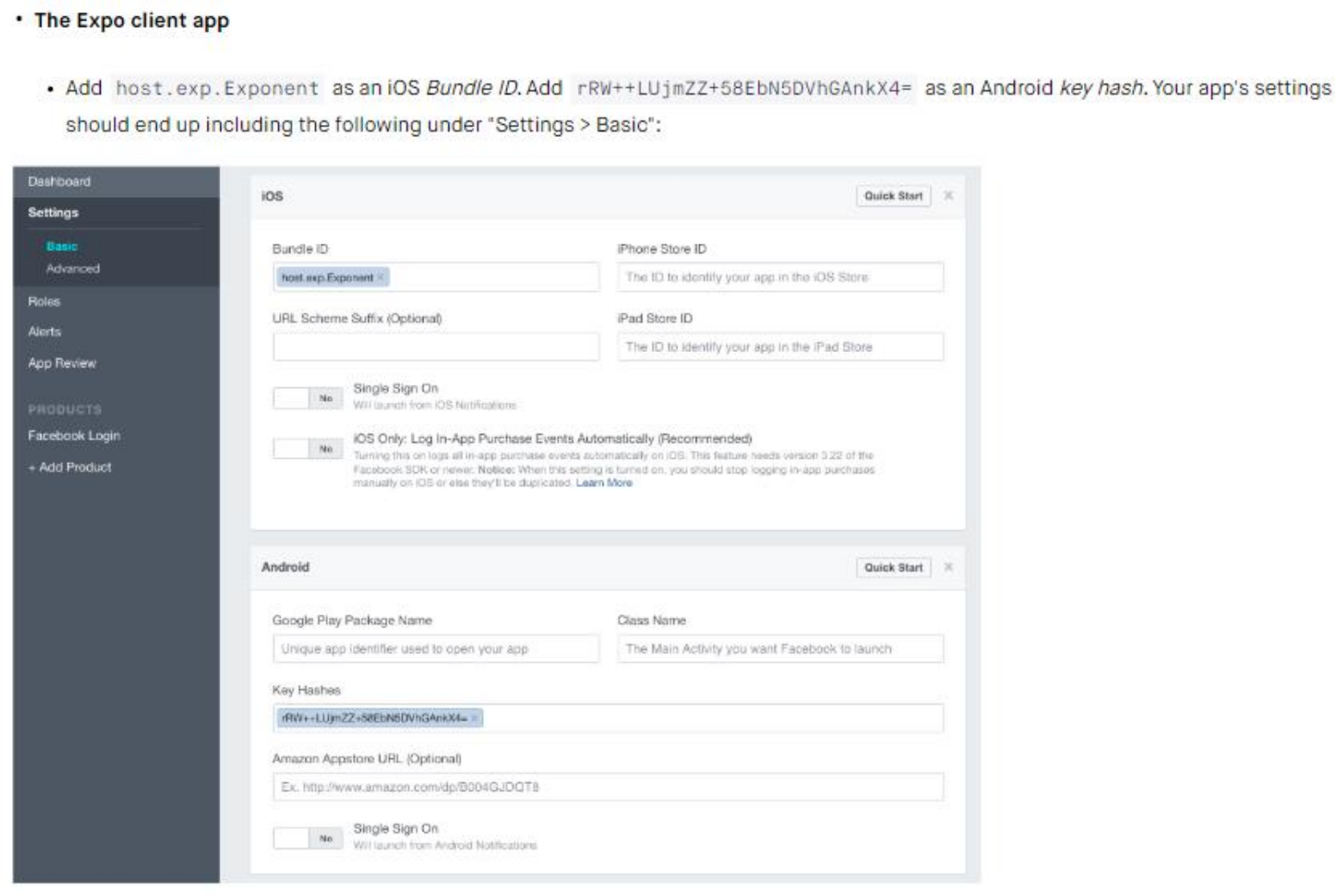
SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload

- 08 Push Notification
- 09 Compass
- 10 Geocoding
- **11 Facebook**
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib

- צריך ליצור פרוייקט ב FB developer API בכדי לקבל appId

- https://docs.expo.io/versions/latest/sdk/facebook/
- rRW++LUjmZZ+58EbN5DVhGAnkX4=
- host.exp.Exponent

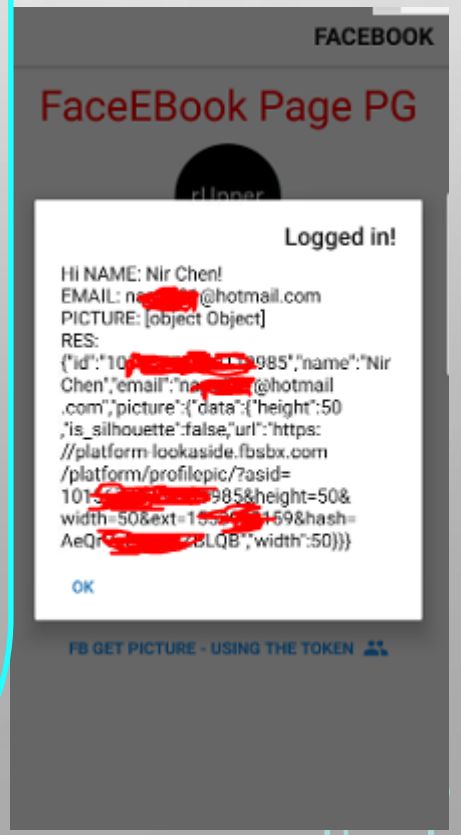


FB

- צריך לקבל token FB יחודי בכדי לעבוד עם ה API שלהם.
- פה אני מקבל שם, מייל, , מספר ID...

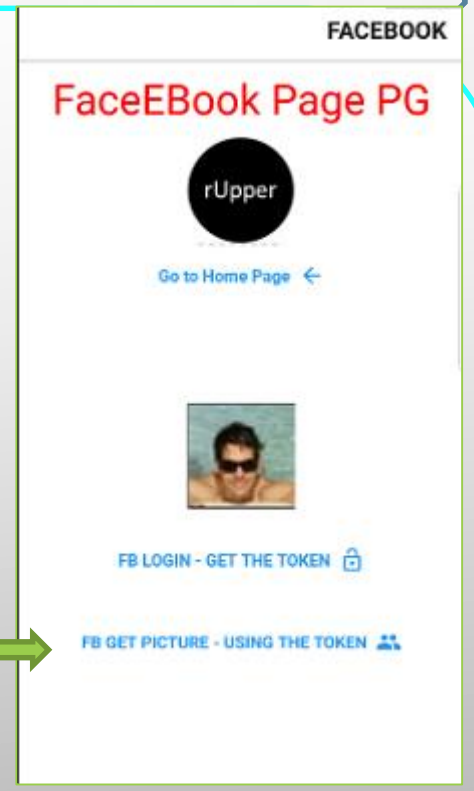
```
import * as Facebook from 'expo-facebook';
...
async function btnFBLogin() {
    try {
      await Facebook.initializeAsync('[YOUR APPID]');
      const { type, token, expires, permissions, declinedPermissions, }
        = await Facebook.logInWithReadPermissionsAsync({
          permissions: ['public_profile'],
        });
      if (type === 'success') {
        // Get the user's name using Facebook's Graph API
        const response = await fetch(`https://graph.facebook.com/me?fields=id,name,email,picture&
access_token=${token}`);
        let res = await response.json();
        Alert.alert('Logged in!', `Hi NAME: ${res.name}!\nEMAIL: ${res.email}\nPICTURE: ${res.picture.
data.url}`);
        //Alert.alert('Logged in!', `Hi NAME: ${res.name}!\nEMAIL: ${res.email}\nPICTURE: ${res.pictu
re}\nRES:${JSON.stringify(res)} `);
      } else {          type === 'cancel'       }
    } catch ({ message }) {
      alert(`Facebook Login Error: ${message}`);
    }
  }
```


מחקתי מהדוגמה את הפרטים האישיים



FB

```
btnFetch_PersonPicture = () => {
        // POST adds a random id to the object sent
        fetch(`https://graph.facebook.com/me?fields=picture&access_token=${this.state.token}`, {
            method: 'POST',
            body: '',
            headers: {
                "Content-type": "application/json; charset=UTF-8"
            }
        })
            .then(response => response.json())
            .then(json => {
                if (json != null) {
                    this.setState({ photoUrl: json.picture.data.url });
                alert(`picture= ${json.picture}\npicture.data.url= ${(json.picture.data.url)}\nRES=${JSON.stringify(json)}`);
                } else {
                    this.setState({ lblErr: true });
                }
            });
    }
```



- פה אני מקבל תמונה

SYLLABUS

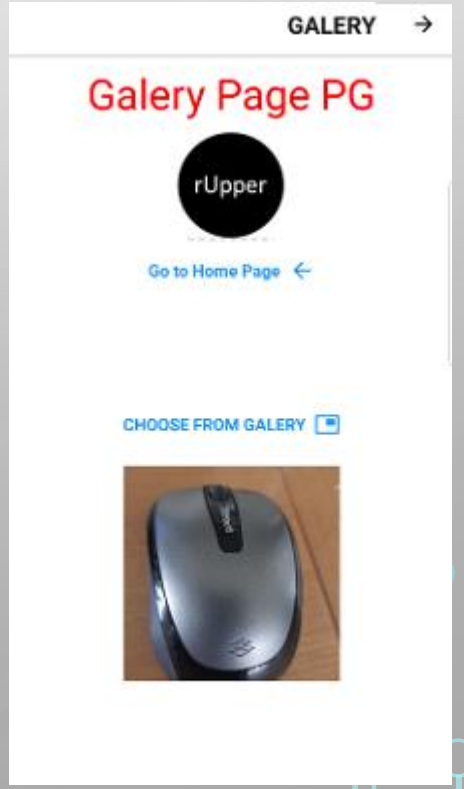
- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload

- 08 Push Notification
- 09 Compass
- 10 Geocoding
- 11 Facebook
- **12 Image gallery**
- 13 Sms
- 14 React Elements UI Lib

IMAGE GALLERY

- ניתן לבחור תמונה מהגלריה
- expo install expo-image-picker

```
import * as ImagePicker from 'expo-image-picker';
...
btnOpenGalery = async () => {
        let result = await ImagePicker.launchImageLibraryAsync({
            //allowsEditing: true,
            //aspect: [4, 3],
          });

          if (!result.cancelled) {
            this.setState({ image: result.uri });
          }
    };
```



SYLLABUS

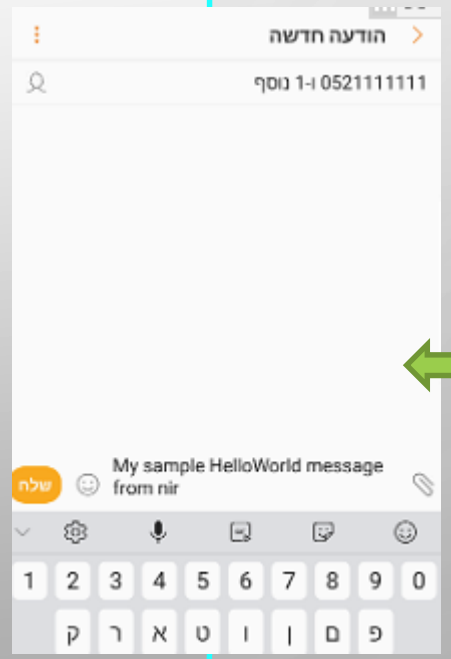
- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

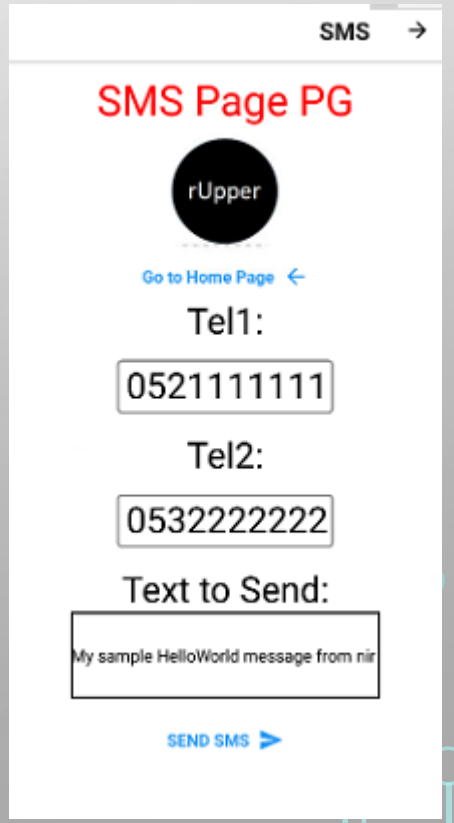
- 09 Compass
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- **13 Sms**
- 14 React Elements UI Lib

SMS

- ניתן לפתוח את מסך הוסעת ה SMS עם רשימת טלפונים והודעה מוכנה מהאפליקציה שלנו.
- זה לא שולח מהאפליקציה שלנו אלה רק פותח את אפליקציית ה SMS-ים הקיימת

```
import { SMS } from 'expo';
...
btnSendSms = async () => {
        const isAvailable = await SMS.isAvailableAsync();
        if (isAvailable) {
            //alert('send');
            const { result } =
            await SMS.sendSMSAsync(
                [this.state.txtTel1, this.state.txtTel2],
                 this.state.txtTextValue);
            // alert(result);
            this.setState({ txtTextValue: result });
        } else {
            alert('misfortune... there\'s no SMS available on this device');
        }
    };
```



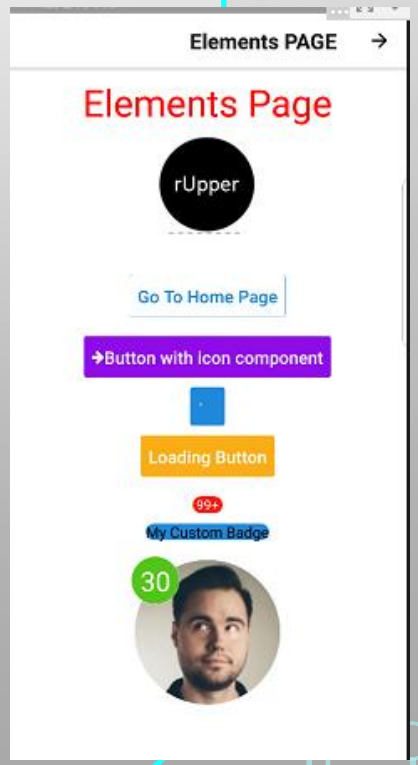
SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification
- 09 Compass
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- 13 Sms
- **14 React Elements UI Lib**

REACT NATIVE ELEMENTS UI

```
import { Button, ThemeProvider } from 'react-native-elements';
import Icon from 'react-native-vector-icons/FontAwesome';

const MyApp = () => {
  return (
    <ThemeProvider>
    <Button
          icon={
            <Icon
              name="arrow-right"
              size={15}
              color="white"
            />
          }
          iconRight
          title="Button with right icon"
        />
    </ThemeProvider>
  );
};
```

- https://react-native-training.github.io/react-native-elements/
- npm install --save react-**native**-elements



SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification
- 09 Compass
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib
- **15 Linking**

LINKING

```
Linking.openURL('https://expo.io');
WebBrowser.openBrowserAsync('https://expo.io');
…
Linking.openURL('mailto:support@expo.io?subject=Congrats Snoopy&body=Enjoy your stay,%0ARegards');
…
Linking.openURL('tel:+123456789');
….
Linking.openURL('sms:+123456789');
…
    //this will open the app through whatsapp
    //1. need to publish the app first!
    //2. when clicked in whatsapp this will go to the published page on expo.io
    //   then you can open the app in the expo if installed.
    btnWhatsapp = () => {
        let text = "hello ";
        text = 'https://expo.io/@nirc/get-a-ride-demo';
        let phoneNumber = '+972523333333';
        Linking.openURL(`whatsapp://send?text=${text}&phone=${phoneNumber}`);
    }
```

Open a url

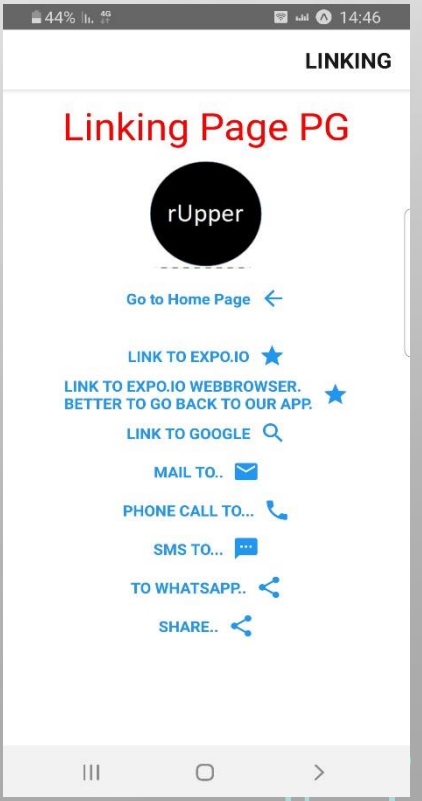
Send mail

Open dialer

Send sms

Send whatsapp

- כל מה שקשור ליצירת קשר עם אפליקציות חיצוניות

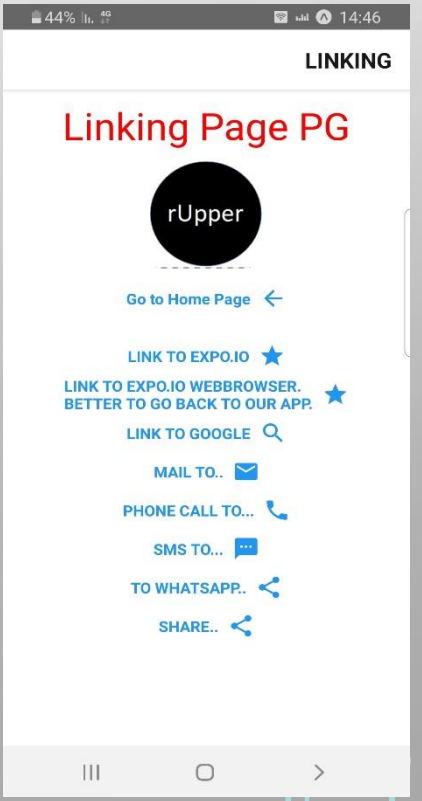


LINKING

```
btnShare = () => {
        //let text =  Expo.Linking.makeUrl();
        let text = 'https://expo.io/@nirc/get-a-ride-demo';

        Share.share({
            message: "Click Here to View More! " + text,
            url: text,
            title: 'nir has invited you to join this activity',
        })
            .then((result) => {
                console.log(result)
                if (result === 'dismissedAction') {
                    return
                }
            })
            .catch((error) => console.log(error))
    }
```

Open all sharable apps

- כל מה שקשור ליצירת קשר עם אפליקציות חיצוניות



SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

- 09 Compass
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib
- 15 Linking
- **17-1 Maps**

```
import { StyleSheet, Text, View, Dimensions } from 'react-native';
import MapView from 'react-native-maps';
import { Marker } from 'react-native-maps';
...
<View style={styles.container}>
      <MapView
          style={{flex: 0.7, width: Dimensions.get('window').width - 30,}}
          region={{
            latitude: 32.157154,
            longitude: 34.843893,
            latitudeDelta: 0.0122,
            longitudeDelta: 0.0121,
          }}  >
          <Marker
            coordinate={{
              latitude: 32.15715,
              longitude: 34.843893
            }}
            title='my place:)'
            description='here i am'
            //image={require('../assets/icon.png')}
          />
      </MapView>
</View>
```


MAPS

- expo install react-native-maps
- https://github.com/react-native-community/react-native-maps

SYLLABUS

- -01 Expo
- 00 Debug
- 01 APK Creation
- 02 React Native Material UI
- 03 Fetch
- 04 Geolocation
- 05 Camera
- 06 Navigation
- 07 Image Upload
- 08 Push Notification

- 09 Compass
- 10 Geocoding
- 11 Facebook
- 12 Image gallery
- 13 Sms
- 14 React Elements UI Lib
- 15 Linking
- 17-1 Maps
- **17 Fingerprint authentication**

FINGERPRINT AUTHENTICATION

```
import * as LocalAuthentication from 'expo-local-authentication';

scanFingerPrint = async () => {
  try {
    let results = await LocalAuthentication.authenticateAsync();
    if (results.success) {
      this.setState({
        modalVisible: false,
        authenticated: true,
        failedCount: 0,
      });
    } else {
      this.setState({
        failedCount: this.state.failedCount + 1,
      });
    }
  } catch (e) {
    console.log(e);
  }
};
```


- expo install expo-local-authentication

MORE EXPO CAPABILITIES!!!

SDK API REFERENCE

- Introduction
- Accelerometer
- Admob
- Amplitude
- AppLoading
- ART
- Asset
- Audio
- AuthSession
- AV
- BarCodeScanner
- BlurView
- Branch
- Brightness
- Calendar
- Camera
- Constants
- Contacts
- DeviceMotion
- DocumentPicker
- ErrorRecovery
- FacebookAds
- Facebook
- FaceDetector
- FileSystem
- Fingerprint
- Font
- GestureHandler
- GLView
- Google
- Gyroscope
- ImageManipulator
- ImagePicker
- IntentLauncherAndroid
- KeepAwake
- LinearGradient
- Localization
- Localization
- Location
- Lottie
- Magnetometer
- MailComposer
- MapView
- Notifications
- Payments
- Pedometer
- Permissions
- Print
- registerRootComponent
- ScreenOrientation
- SecureStore
- Segment
- Speech
- SQLite
- Svg
- takeSnapshotAsync
- Updates
- Video
- WebBrowser