**Setup For Windows**

**URL:** <https://reactnative.dev/docs/set-up-your-environment>

## **Installing dependencies**

#### **First, install Choco**

<https://chocolatey.org/install>

Need to install using PowerShell with administrator access

Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))

#### **Then, install Node,JDK**

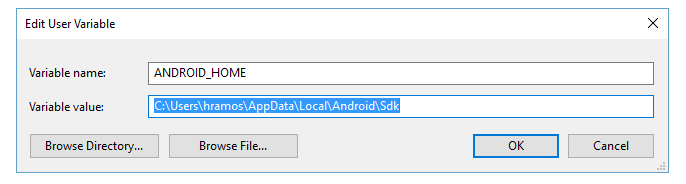
choco install -y nodejs-lts microsoft-openjdk17

## **Install the Android Studio**

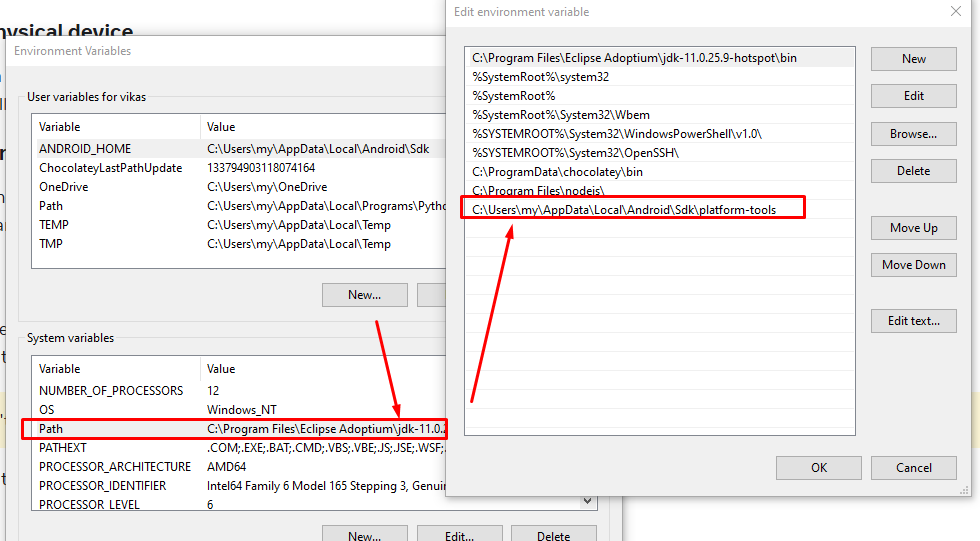
<https://developer.android.com/studio/index.html>

After installing the Android Studio now install the emulator, os, and all from the virtual device manager

#### **Configure the ANDROID\_HOME environment variable**



#### **Add platform-tools to Path**



### **That's it!**

## **Now Need to install react native App**

<https://reactnative.dev/docs/environment-setup>

Install Following Commands

npx create-expo-app@latest

//In cli provide app name “my-app”

cmd my-app

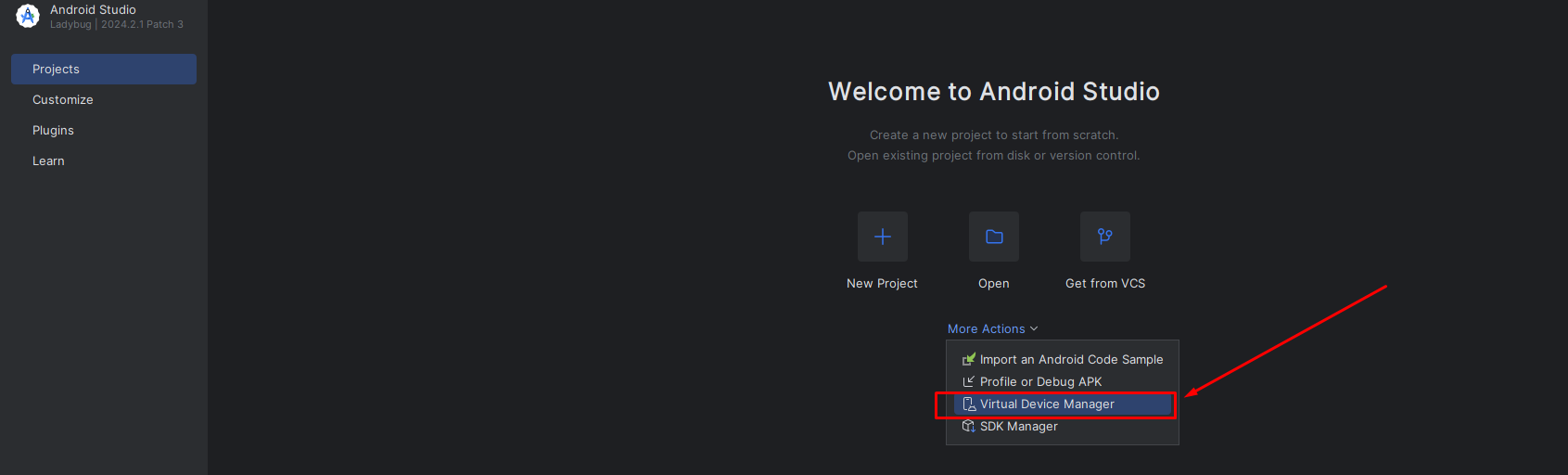
npx expo start

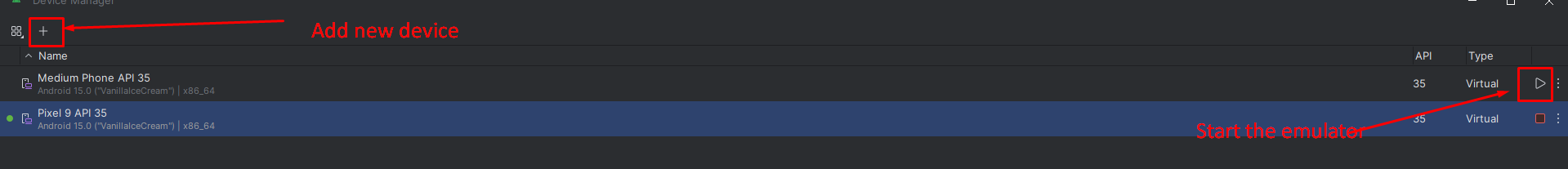
In cli there is option to run app in android

Before run this make sure you emulator is already running



## **Start Emulator**





## **Create a Build**

<https://docs.expo.dev/build/setup/>

npm install -g eas-cli

eas login

#### After above command need to Log in to your Expo account

eas build:configure

Above commands will create the .aab file which is for play store

For creating.apk follow the following commands

<https://docs.expo.dev/build-reference/apk/>

Replace this in eas.json on root

{

"build": {

"preview": {

"android": {

"buildType": "apk"

}

},

"preview2": {

"android": {

"gradleCommand": ":app:assembleRelease"

}

},

"preview3": {

"developmentClient": true

},

"preview4": {

"distribution": "internal"

},

"production": {}

}

}

Then run the following command

npm install -g eas-cli

eas build -p android --profile preview

## 

## 

## 

## 

## **Login in React native**

React Native doesn't have localStorage. Use AsyncStorage from @react-native-async-storage/async-storage to store tokens and other data persistently.

npm install @react-native-async-storage/async-storage

Install Axios

npm install axios

**Basic Login Form in Native using Wordpress API**

**import React, { useState, useEffect } from 'react';**

**import {Text, View, TextInput, Image, Alert, ImageBackground, TouchableOpacity, ActivityIndicator} from 'react-native';**

**import ThemeStyle from '../css/ThemeStyle';**

**import axios from 'axios';**

**import AsyncStorage from '@react-native-async-storage/async-storage';**

**const LoginScreen = ({ navigation }) => {**

**// prevent user to open direct this page if user is login**

**useEffect(() => {**

**const checkToken = async () => {**

**const token = await AsyncStorage.getItem('token');**

**if (token) {**

**// Navigate to Dashboard**

**navigation.navigate('Home');**

**}**

**};**

**checkToken(); // Call the function to check token**

**}, [navigation]); // Re-run the effect if navigation changes**

**// Use States for fetch user details and error handling**

**const [emailOrUsername, setEmailOrUsername] = useState('');**

**const [password, setPassword] = useState('');**

**const [success, setSucess] = useState('');**

**const [loader, setLoader] = useState(false);**

**// when user hit submit button this function will work**

**const handleSubmit = async () => {**

**setSucess(false);**

**if (!emailOrUsername || !password) {**

**Alert.alert('Error', 'Please fill in all fields.');**

**return;**

**}**

**// Perform validation (if needed)**

**if (emailOrUsername.includes('@') && !/\S+@\S+\.\S+/.test(emailOrUsername)) {**

**Alert.alert('Error', 'Please enter a valid email address.');**

**return;**

**}**

**// Alert.alert('Success', `Email/Username: ${emailOrUsername}\nPassword: ${'\*'.repeat(password.length)}`);**

**setLoader(true);**

**try {**

**const response = await axios.post(`https://bookmyevents.tmdemo.in/bme-admin/wp-json/jwt-auth/v1/token`, {**

**username: emailOrUsername,**

**password: password,**

**});**

**// Alert.alert('Success', JSON.stringify(response.data));**

**// Save the JWT token and user info using AsyncStorage**

**await AsyncStorage.setItem('token', response.data.token);**

**await AsyncStorage.setItem('userName', response.data.user\_display\_name);**

**// Reset the navigation stack and navigate to the dashboard**

**setLoader(false);**

**setSucess(false);**

**// Navigate to Dashboard**

**navigation.navigate('Home');**

**} catch (error) {**

**// Alert.alert('Error', error.response?.data?.message || 'Something went wrong');**

**setEmailOrUsername('');**

**setPassword('');**

**setLoader(false);**

**setSucess(true);**

**}**

**};**

**return (**

**<View style={[ThemeStyle.mainWrapper,{flex:1,alignItems:'center'}]}>**

**<View>**

**<Image**

**source={require('../../assets/main-logo.png')}**

**style={ThemeStyle.mainLogo}**

**/>**

**</View>**

**<View>**

**<Text style={ThemeStyle.loginHeader}>Welcome Back !</Text>**

**<Text style={ThemeStyle.loginSubHdng}>Use Credentials to access your account</Text>**

**</View>**

**{/\* Error message display \*/}**

**{success && (**

**<View style={ThemeStyle.inputWrap}>**

**<Text style={ThemeStyle.ErrorMsg}>Invalid username or password.</Text>**

**</View>**

**)}**

**{/\* Username & Email \*/}**

**<View style={ThemeStyle.inputWrap}>**

**<Image**

**source={require('../../assets/user.png')}**

**style={ThemeStyle.fieldIcon}**

**/>**

**<TextInput**

**style={ThemeStyle.globalInput}**

**placeholder="Enter email or username"**

**placeholderTextColor="#d3d0d1"**

**value={emailOrUsername}**

**onChangeText={setEmailOrUsername}**

**editable={!loader} // Disable the field when loader is true**

**/>**

**</View>**

**{/\* Password \*/}**

**<View style={ThemeStyle.inputWrap}>**

**<Image**

**source={require('../../assets/pass.png')}**

**style={ThemeStyle.fieldIcon}**

**/>**

**<TextInput**

**style={ThemeStyle.globalInput}**

**placeholder="Enter your password"**

**placeholderTextColor="#d3d0d1"**

**value={password}**

**onChangeText={setPassword}**

**secureTextEntry // Hides the password input**

**editable={!loader} // Disable the field when loader is true**

**/>**

**</View>**

**{/\* Forgot Password \*/}**

**<View style={[ThemeStyle.inputWrap , ThemeStyle.forgotWrap]}>**

**<TouchableOpacity onPress={() => navigation.navigate("ForgotPassword")} style={ThemeStyle.forgotPassword}>**

**<Text style={ThemeStyle.forgotPasswordText}> Forgot Password ?</Text>**

**</TouchableOpacity>**

**</View>**

**{/\* Submit \*/}**

**<View style={ThemeStyle.inputWrap}>**

**<ImageBackground**

**source={require('../../assets/button.png')} // Adjust the path to your image**

**style={ThemeStyle.submitBtnBG}**

**>**

**<TouchableOpacity onPress={handleSubmit} style={ThemeStyle.submitBtn}>**

**<Text style={ThemeStyle.submitBtnText}>Login</Text>**

**</TouchableOpacity>**

**</ImageBackground>**

**</View>**

**<View style={ThemeStyle.loginUnderWrap}>**

**<Text style={ThemeStyle.loginUnderText}>**

**Don't have an account?**

**</Text>**

**<TouchableOpacity**

**onPress={() => navigation.navigate("Register")}**

**style={ThemeStyle.loginUnderTouchableArea}**

**>**

**<Text style={ThemeStyle.signupUnderLogin}> Signup</Text>**

**</TouchableOpacity>**

**</View>**

**{/\* Loader display on submit click \*/}**

**{loader && (**

**<View style={ThemeStyle.loaderScreen}>**

**<ActivityIndicator size="large" color="#ffffff" />**

**</View>**

**)}**

**</View>**

**);**

**};**

**export default LoginScreen;**

### Dashboard page which opens after login Prevent user from going back to login screen after logging in successfully in React Native

**import React, { useEffect } from "react";**

**import { Text, View, BackHandler } from "react-native";**

**import ThemeStyle from "./css/ThemeStyle";**

**const HomeScreen = ({ navigation }) => {**

**// prevent user to go back agter login in anroid only**

**useEffect(() => {**

**const handleBackButton = () => true;**

**BackHandler.addEventListener("hardwareBackPress", handleBackButton);**

**return () => {**

**BackHandler.removeEventListener(**

**"hardwareBackPress",**

**handleBackButton**

**);**

**};**

**}, []);**

**// prevent user to go back agter login in both android & ios**

**useEffect(() => {**

**const unsubscribe = navigation.addListener('beforeRemove', (e) => {**

**e.preventDefault(); // Prevent navigation**

**// You can also show a confirmation dialog here before navigating away**

**});**

**return unsubscribe; // Cleanup when the component is unmounted**

**}, [navigation]);**

**return (**

**<View style={ThemeStyle.MainScreenWrapper}>**

**<View>**

**<Text>Home</Text>**

**</View>**

**</View>**

**);**

**};**

**export default HomeScreen;**

## Register in React native

import React, { useState } from "react";

import { Text, View, TextInput, Image, Alert, ScrollView, ImageBackground, TouchableOpacity, ActivityIndicator} from "react-native";

import { Picker } from "@react-native-picker/picker";

import ThemeStyle from "../css/ThemeStyle";

import axios from "axios";

const RegisterScreen = ({ navigation }) => {

const [formData, setFormData] = useState({ firstname: "", lastname: "", username: "", email: "", phone: "", password: "", confirmPassword: ""});

const [loader, setLoader] = useState(false);

const handleSubmit = async () => {

const { firstname, lastname, username, email, phone, password, confirmPassword} = formData;

if (!firstname || !lastname || !username || !email || !phone || !password || !confirmPassword) {

Alert.alert("Error", "Please fill in all fields.");

return;

}

// Validate email format

if (!/\S+@\S+\.\S+/.test(email)) {

Alert.alert("Error", "Please enter a valid email address.");

return;

}

// Validate phone number (example: check length)

if (phone.length < 10 || isNaN(phone)) {

Alert.alert("Error", "Please enter a valid phone number.");

return;

}

// Validate passwords match

if (password !== confirmPassword) {

Alert.alert("Error", "Passwords do not match.");

return;

}

setLoader(true);

try {

const auth = btoa("main\_admin:Ze5L lZ7I aOeb kmIc jD3c DZB9"); // Replace with your admin username and application password

await axios.post(

`https://bookmyevents.tmdemo.in/bme-admin/wp-json/wp/v2/users`,

{

first\_name: formData.firstname,

last\_name: formData.lastname,

username: formData.username,

email: formData.email,

password: formData.password,

acf: {

field\_6763c144d8d8a: formData.phone,

},

},

{

headers: {

Authorization: `Basic ${auth}`,

},

}

);

setLoader(false);

navigation.navigate("Login");

} catch (error) {

setLoader(false);

console.log(error);

}

};

const handleInputChange = (field, value) => {

setFormData((prevData) => ({

...prevData,

[field]: value,

}));

};

return (

<ScrollView>

<View style={[ThemeStyle.mainWrapper, { flex: 1, alignItems: "center" }]}>

<View>

<Image

source={require("../../assets/main-logo.png")}

style={ThemeStyle.mainLogo}

/>

</View>

<View>

<Text style={ThemeStyle.loginHeader}>Create an account !</Text>

<Text style={ThemeStyle.loginSubHdng}>

Please fill the following details to create an account

</Text>

</View>

{/\* Firstname \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/user.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Enter firstname"

placeholderTextColor="#d3d0d1"

value={formData.firstname}

onChangeText={(text) => handleInputChange("firstname", text)}

/>

</View>

{/\* Lastname \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/user.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Enter lastname"

placeholderTextColor="#d3d0d1"

value={formData.lastname}

onChangeText={(text) => handleInputChange("lastname", text)}

/>

</View>

{/\* Username \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/user.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Enter username"

placeholderTextColor="#d3d0d1"

value={formData.username}

onChangeText={(text) => handleInputChange("username", text)}

/>

</View>

{/\* Email \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/email.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Enter email"

placeholderTextColor="#d3d0d1"

value={formData.email}

onChangeText={(text) => handleInputChange("email", text)}

/>

</View>

{/\* Phone \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/phone.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Enter phone number"

placeholderTextColor="#d3d0d1"

value={formData.phone}

onChangeText={(text) => handleInputChange("phone", text)}

keyboardType="phone-pad"

/>

</View>

{/\* Password \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/pass.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Enter your password"

placeholderTextColor="#d3d0d1"

value={formData.password}

onChangeText={(text) => handleInputChange("password", text)}

secureTextEntry

/>

</View>

{/\* Confirm Password \*/}

<View style={ThemeStyle.inputWrap}>

<Image

source={require("../../assets/pass.png")}

style={ThemeStyle.fieldIcon}

/>

<TextInput

style={ThemeStyle.globalInput}

placeholder="Confirm password"

placeholderTextColor="#d3d0d1"

value={formData.confirmPassword}

onChangeText={(text) => handleInputChange("confirmPassword", text)}

secureTextEntry

/>

</View>

{/\* Signup \*/}

<View style={ThemeStyle.inputWrap}>

<ImageBackground

source={require("../../assets/button.png")} // Adjust the path to your image

style={ThemeStyle.submitBtnBG}

>

<TouchableOpacity

onPress={handleSubmit}

style={ThemeStyle.submitBtn}

>

<Text style={ThemeStyle.submitBtnText}>Signup</Text>

</TouchableOpacity>

</ImageBackground>

</View>

<View style={ThemeStyle.loginUnderWrap}>

<Text style={ThemeStyle.loginUnderText}>

Already have an account ?

</Text>

<TouchableOpacity

onPress={() => navigation.navigate("Login")}

style={ThemeStyle.loginUnderTouchableArea}

>

<Text style={ThemeStyle.signupUnderLogin}> Log in here</Text>

</TouchableOpacity>

</View>

{loader && (

<View style={ThemeStyle.loaderScreen}>

<ActivityIndicator size="large" color="#ffffff" />

</View>

)}

</View>

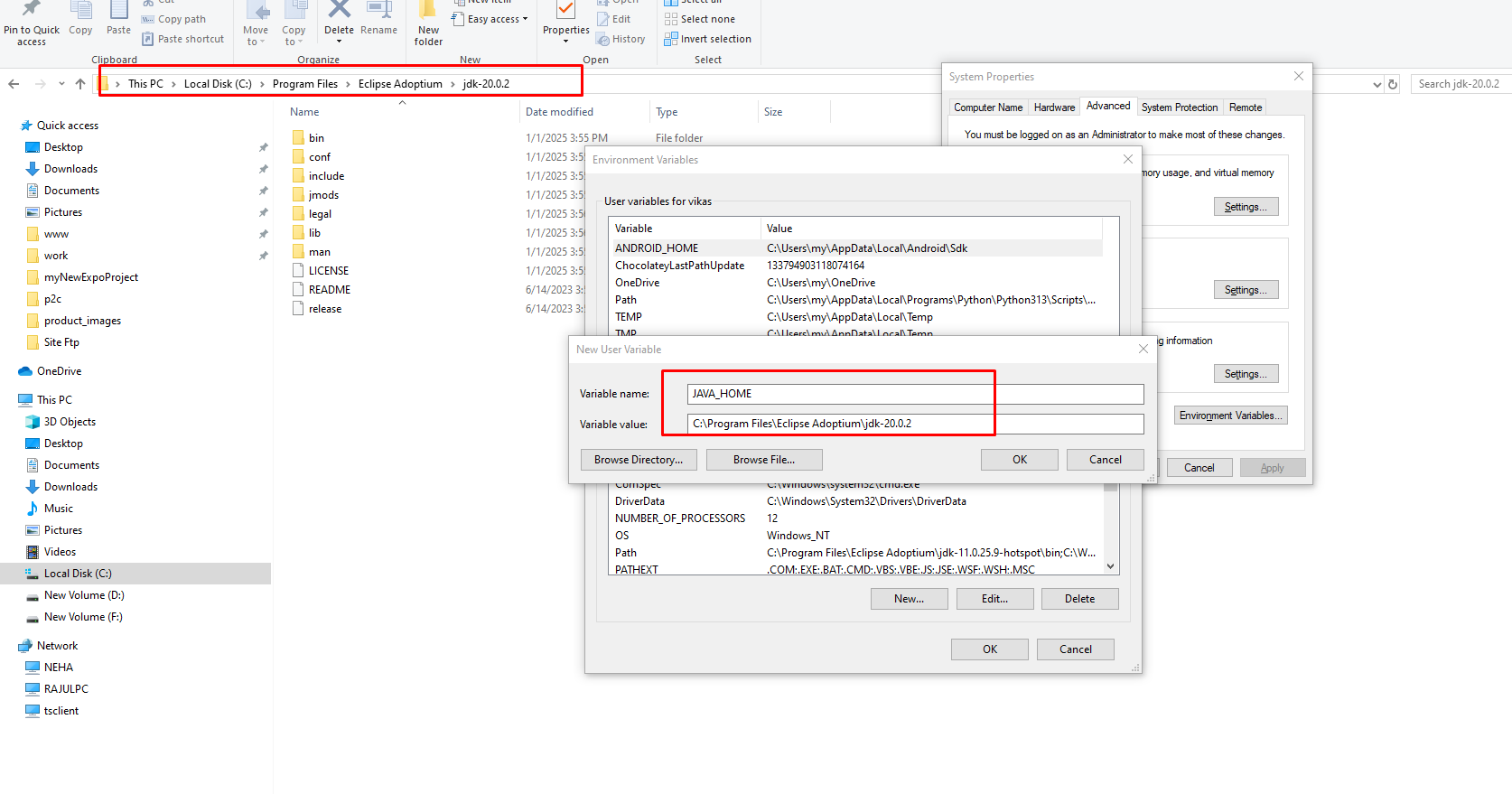
</ScrollView>

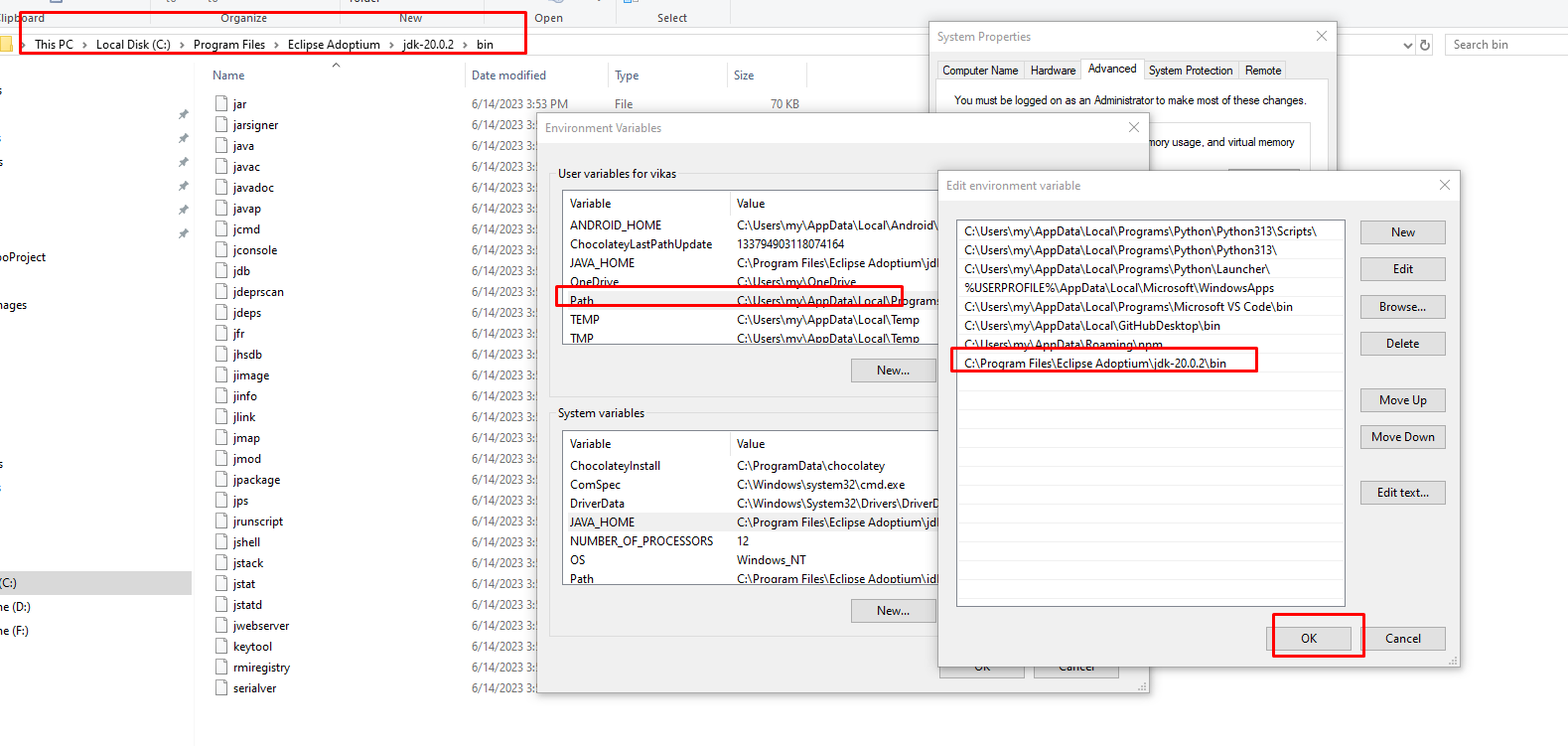
);

};

export default RegisterScreen;

## If need to update JDK First download the latest version of JDK

****

****

**For web view of react native**

**use these commands**

**npm install -g bun**

**npx expo install react-dom react-native-web @expo/metro-runtime**

## Load Custom Fonts

<https://docs.expo.dev/develop/user-interface/fonts/>

Install font module

npx expo install expo-font

**import { useFonts } from "expo-font";**

**function App() {**

**const [fontsLoaded] = useFonts({**

**// link your fonts as show below**

**"Belleza-Regular": require("./assets/fonts/Belleza/Belleza-Regular.ttf"),**

**"DMSans-Black": require("./assets/fonts/DM\_Sans/DMSans-Black.ttf"),**

**});**

**}**

**Then just use the font family in in your style component**

**loginSubHdng: {**

**fontFamily: "DMSans-SemiBold",**

**},**

## Important Node Modules

1. For Viewport units (vh,vw)  
   <https://www.npmjs.com/package/react-native-expo-viewport-units>
2. Rendering html response from API in React native <https://www.npmjs.com/package/react-native-render-html>