Mobile App Lab 1

1. **System Requirements:** The minimum recommended CPU in order to run React Native on Windows is an intel i5. Minimum RAM for spinning up a mobile application on react native would be 8GB, though it 16GB or higher is recommended for more advanced projects. Windows 10 seems to be recommended for spinning up a mobile application using React Native, though I was also able to do it just fine using my system that runs on Windows 11.
2. **Installation Instructions:** First, you’ll need to make sure that you have Android Studio installed on your system. Next, one can install Node via Chocolatey, which is a package manager for Windows. I did this by typing the following command into Windows PowerShell:

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'))

Next, I had to run the following command in order to install the JDK for the project:

choco install -y nodejs-lts microsoft-openjdk11

1. **Configuration Steps:** It is important to make sure you have the proper Android SDKs installed by going into settings and finding the Android SDK section under the Languages and Frameworks tab. Here, it is vital to make sure that Android SDK Platform 33 and Intel x86 Atom\_64 System Image are both downloaded and installed on the system. Under the SDK Tools tab, it is also important to make sure that you have the 33.0.0 version of Android SDK Build-Tools selected.
2. **Project Creation:** Following the instructions provided by the instructor, the next step was to set up the environment required to build an app with native code. This was done in the Control Panel, where I changed the environmental variables found in the User Accounts section. I made a variable name called ANDROID\_HOME and applied it a variable value that exists within the file system of my computer.

Next, I opened a new command prompt in Windows Powershell and used the following command to verify that the new variable had been added to the list:

**Get-ChildItem -Path Env:\**

Next, it was finally time to run the following command and create a brand new React Native project:

npx react-native@latest init AwesomeProject

What this did was create all of the files and folders necessary for the project.

1. **Running The Project:** From this point, the process is fairly simple as all there is left to do is open the code in Android Studio, choose a virtual device via the device manager in the IDE, and then run Metro with the following command:

npm start

It is vital that this command be run in the proper directory (that being the home of the desired project), otherwise it won’t do anything and will probably generate an error. We can now open a separate command terminal and run the following command in order to finally spin up the mobile app in its emulator:

npm run android

This will begin the generation of a virtual display that hosts the application created earlier in the process.

1. **Troubleshooting:** An important step in troubleshooting potential errors in this project is to make sure you have the most recent version of Node installed. Install node using Chocolatey, as discussed earlier in this document. You can also remove the global react-native-cli, if you have it installed already, as this can lead to errors. It is recommended that you run the following command before running your program, just in case:

npm uninstall -g react-native-cli @react-native-community/cli

1. **Resources:**

* Step-by-step installation guide for React Native: <https://reactnative.dev/docs/environment-setup?guide=native>
* Open-source Chocolatey installation guide:

<https://chocolatey.org/install>

* More information on React Native:

<https://www.netguru.com/glossary/react-native>