



| | | |
|--|--|---|
| Topic | Capstone class: App Publishing and Local Environment Setup | |
| Class Description | Students learn to set up expo on their local environment. They also learn to generate aab or ipa files which can be published on playstore or appstore. Students build a native Weather app in the local expo environment to forecast weather. | |
| Class | C62 | |
| Class time | 45 mins | |
| Goal | <ul style="list-style-type: none"> Set up expo on the local machine. Generate aab or ipa files for apps to be published on playstore or appstore. Build an app to keep track of certain trading stocks. | |
| Resources Required | <ul style="list-style-type: none"> Teacher Resources <ul style="list-style-type: none"> Laptop with internet connectivity Earphones with mic Notebook and pen Android/iOS Smartphone with Expo App installed Student Resources <ul style="list-style-type: none"> Laptop with internet connectivity Earphones with mic Notebook and pen Android/iOS Smartphone with Expo App installed | |
| Class structure | Warm Up Teacher-led Activity Student-led Activity Wrap up | 5 mins 15 min 15 min 5 min |
| WARM-UP SESSION - 5 mins | | |
| <div>  </div> <p>Teacher starts slideshow from slides 1 to 16</p> <p>Refer to speaker notes and follow the instructions on each slide.</p> | | |
| Activity details | | Solution/Guidelines |

| | |
|--|--|
| <p>Hi, how have you been? Are you excited to learn something new?</p> <p>Run the presentation from slide 1 to slide 4.</p> <p>The following are the warm-up session deliverables:</p> <ul style="list-style-type: none"> • Reconnect with previous class topics. • Warm-Up quiz session. | <p>ESR: Varied Response.</p> <p>Click on the slide show tab and present the slides.</p> |
| <p align="center">QnA Session</p> | |
| <p align="center">Question</p> | <p align="center">Answer</p> |
| <p>How do we resolve the error which comes when pressing the button repeatedly wherein the team name comes on the quiz master app repeatedly?</p> <p>A. By emptying the teams[] array</p> <p>B. By resetting the database</p> <p>C. By using enabled field in the database</p> <p>D. By using the status state in the app</p> | <p>A</p> |
| <p>getTime() function which is used to get the time from the client machine can be replaced by _____ ?</p> <p>A. New Date()</p> <p>B. firebase.database.ServerValue.TIMESTAMP</p> <p>C. firebase.ServerValue.TIMESTAMP</p> <p>D. firebase.database.TIMESTAMP</p> | <p>B</p> |
| <p align="center">Continue the warm-up session</p> | |
| <p align="center">Activity details</p> | <p align="center">Solution/Guidelines</p> |
| <p>Run the presentation from slide 4 to slide 16 to set the problem statement.</p> <p>The following are the warm-up session deliverables:</p> <ul style="list-style-type: none"> • Review code from the previous class. • Setting up expo on the local environment. | |

| | | |
|---|---|--|
| <div>Teacher ends slideshow</div>  | | |
| TEACHER-LED ACTIVITY - 15 mins | | |
| Teacher Initiates Screen Share | | |
| <p align="center"><u>CHALLENGE</u></p> <ul style="list-style-type: none"> Build the weather app | | |
| Step 2: Teacher-led Activity (15 min) | <p>We'll first build a simple weather forecasting app which gets weather and temperature data from a weather API and displays it on a home screen.</p> <p>Do you remember what an API is?</p> <p>Do you remember how to get data from an API in javascript?</p> | <p>ESR:</p> <p>API is a service which gives us some data based on our query.</p> <p>We use 'fetch()' to get data from the API in javascript.</p> |
| | <p>Correct! Now open a new expo-snack.</p> <p>You can open your App.js file to start writing your code.</p> | <p>Student opens a new expo snack.</p> |
| | <p>First we'll write a getWeather() function which will get the json data from the api.</p> <p><i><Teacher helps the student write the getWeather() function></i></p> | <p>Student code to write the getWeather() function which will get the json data from the api and set it to the state.</p> <p>-Student uses fetch() to get the json data from the api</p> |

| | | |
|--|---|--|
| | | and sets it to the weather in the state |
| | <pre> export default class WeatherScreen extends Component { constructor() { super(); this.state = { weather: '', }; } getWeather = async () => { //change latitude and longitude var url = 'https://fcc-weather-api.glitch.me/api/current?lat=35&lon=139'; return fetch(url) .then(response => response.json()) .then(responseJson => { this.setState({ weather: responseJson, }); }) .catch(error => { console.error(error); }); }; </pre> | |
| | <p>Now we need to show the data on the interface.</p> <p>We'll show a loading message while our function is running and get the data from the API and once we have the data we'll show the forecast. To do that we'll write a if -else condition that</p> <p>if this.state.weather === "" then show the "Loading..." message else show the weather forecast. You can also add a cloud image to make the UI look better.</p> | <p>Student codes to show the data on the interface. In the render function student uses if else condition to return the loading message or show the forecast</p> |

```
render() {
  if (this.state.weather === '') {
    return (
      <View style={styles.container}>
        <Text>Loading...</Text>
      </View>
    );
  } else {
    return (
      <View style={styles.container}>
        <View style={styles.subContainer}>
          <Text style={styles.title}>
            Weather Forecast
          </Text>
          <Image
            style={styles.cloudImage}
            source={require('./clouds.png')}
          />
          <View style={styles.textContainer}>
            <Text style={{ fontSize: 18}}>
              {this.state.weather.main.temp}&deg;C
            </Text>
            <Text style={{ fontSize: 20, margin: 10}}>
              humidity : {this.state.weather.main.humidity}
            </Text>
            <Text style={{fontSize: 20}}>
              {this.state.weather.weather[0].description}
            </Text>
          </View>
        </View>
      </View>
    );
  }
}
```

Now let's check the output.

Student runs the code on the emulator and checks the output



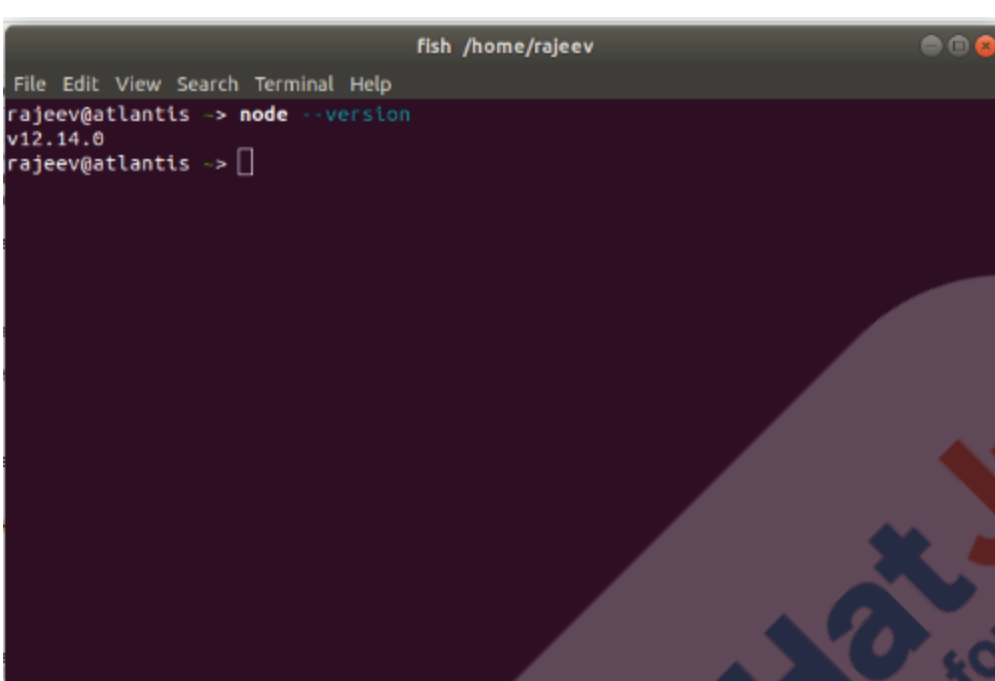
Good job. Now we'll write the code on our local machine. To write and run the code we'll need to install expo and node.js on our machine.

-

| | | |
|--|--|---|
| | <p>We will be following instructions given in Expo documentation on its website to first install expo on our local machine. You can open your activity link to look at the instructions as well.</p> <p>Teacher opens Teacher Activity 1.</p> | <p>The student opens Student Activity 1.</p> |
|  <p>The image shows a screenshot of the Expo documentation website with steps 01 to 06. Step 01: Fall in Love. Step 02: Download NodeJS. Step 03: Get the command line tool. Step 04: Create your first project. Step 05: Preview your project. Step 06: Start coding! The steps include instructions on how to install Node.js, set up Expo, and create a new project.</p> | | |
| | <p>First, we will install node.js on our system.</p> <p>So far, we have only run javascript inside a browser. Node allows us to run javascript outside our browser as well.</p> <p>Let's follow the instructions to install node.</p> | |

| | | |
|--|--|--|
| | <p>For Windows users:</p> <ol style="list-style-type: none"> 1. Download node directly from the given link in Teacher Activity 2 2. Unzip the file. Run the executable inside it (exe) file to install node. 3. To check if node was installed properly, open cmd and type node --version It should show the node version which was installed. <p>For Mac users:</p> <ol style="list-style-type: none"> 1. Install homebrew first. Homebrew is a package manager for your operating system. It helps you in easily installing programs from the terminal. <p>To install homebrew, open your terminal and type: <code>/bin/bash -c "\$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"</code></p> <p>Note: You might have to add "sudo" before the command if you do not have permission to install packages on your OS. "sudo" stands for "do as a super user". You might have to run: <code>sudo /bin/bash -c "\$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"</code></p> | <p>The student installs node from the given link in Student Activity 2 and checks the node --version.</p> |
|--|--|--|

| | | |
|--|--|--|
| | <p>m/Homebrew/install/master/install.sh)”</p> <ol style="list-style-type: none">2. Now install node. On your terminal type: sudo brew install node3. Check if node is installed on your system by typing in the terminal: node --version. <p>For Ubuntu users:</p> <ol style="list-style-type: none">1. Open your terminal and type: sudo apt install node This will install node on your system.2. Check the node installation by typing: node --version | |
|--|--|--|



```

fish /home/rajeev
File Edit View Search Terminal Help
rajeev@atlantis -> node --version
v12.14.0
rajeev@atlantis -> 

```

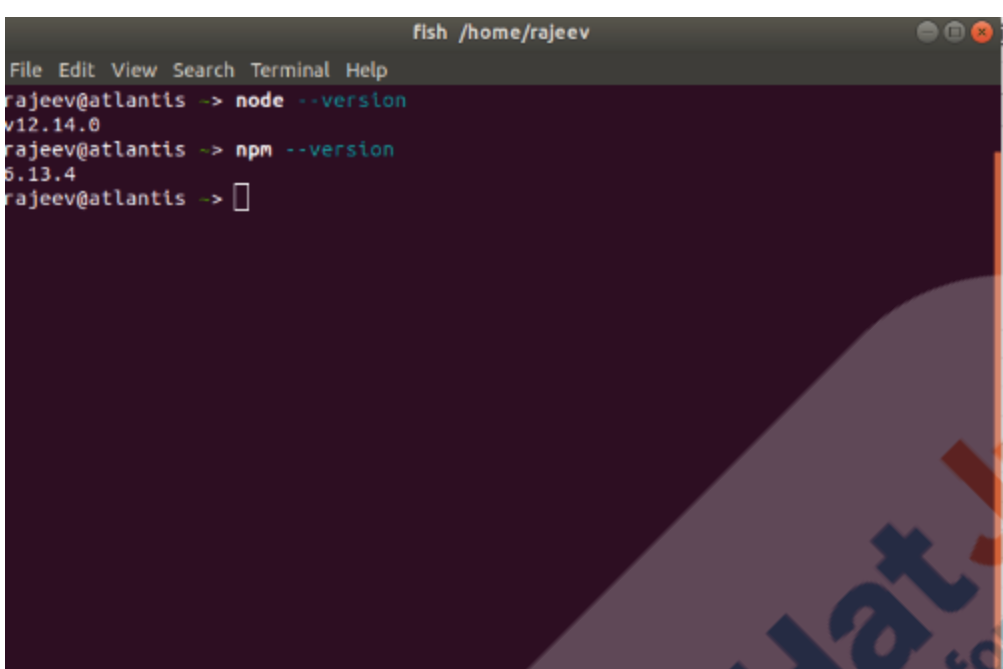
When you install node, npm also gets installed. 'npm' stands for 'node package manager'.

All the libraries that we used in snack including react, react-native, firebase, react-navigation, they all come as node packages. 'npm' helps us in installing and maintaining these packages.

You will learn more about it when we actually use 'npm'.

You can quickly check for 'npm' installation using: `npm --version`

The student checks for 'npm' installation on their system.



```

fish /home/rajeev
File Edit View Search Terminal Help
rajeev@atlantis -> node --version
v12.14.0
rajeev@atlantis -> npm --version
6.13.4
rajeev@atlantis -> 

```

Great! Now we will be using npm to install the expo command line tool.

Expo command line tool or 'expo-cli' comes with many libraries and tools already installed which help us in quickly getting started with building react native apps.

To install 'expo-cli', on your terminal type:

npm install expo-cli --global
or,
npm i -g expo-cli

if you are linux or Mac user add sudo before npm install

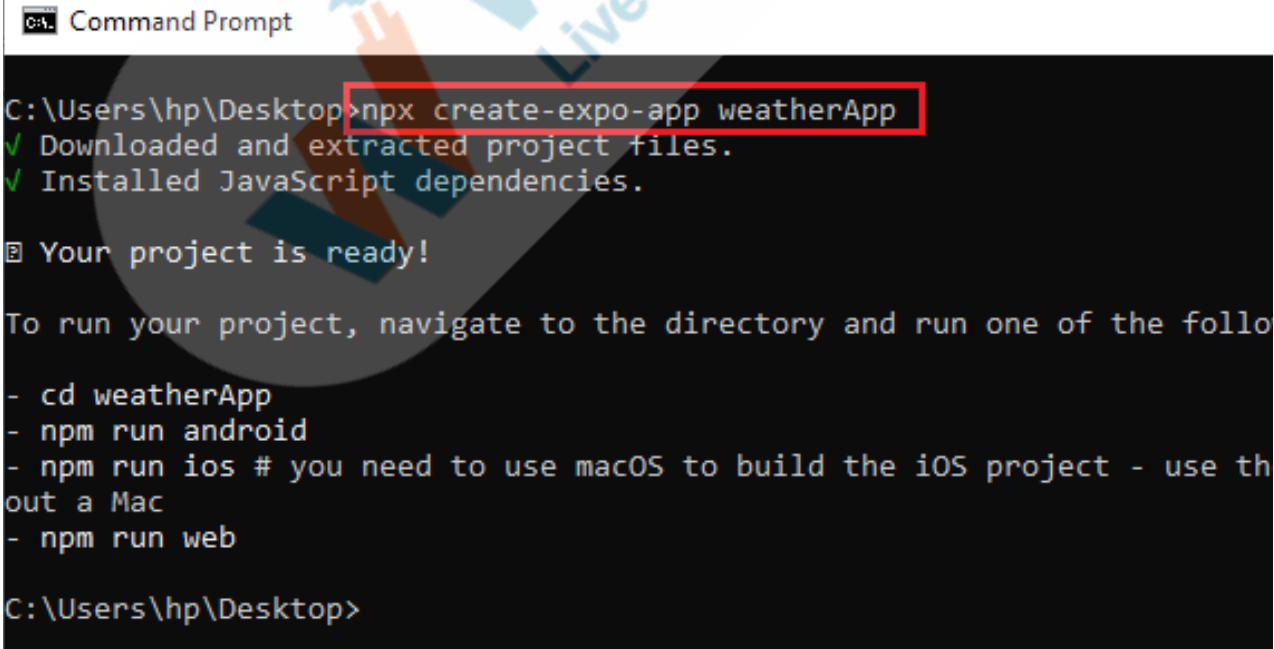
sudo npm install expo-cli --global

The "global" tag installs expo with a global scope. This means you can

The student installs 'expo-cli' on their system.

Note: Installing 'expo-cli' can take some time.

| | | |
|--|--|---|
| | use expo anywhere on your system. Without global tag, expo will be installed only in the folder in which you are running the command. | |
| | <p>Alright ! We have 'expo' installed on our system now.</p> <p>Now let's start coding in the the local environment.</p> <p>First we need to create a new project .To create a new project write</p> <p>npx create-expo-app <project name></p> <p>on your terminal. Choose a blank template , press enter and wait for some time until the process is finished.</p> | <p>Student opens the terminal and writes</p> <p>npx create-expo-app weatherApp</p> <p>-Then selects the blank template , presses enter button and waits till the process is completed.</p> |



```

C:\Users\hp\Desktop>npx create-expo-app weatherApp
✓ Downloaded and extracted project files.
✓ Installed JavaScript dependencies.

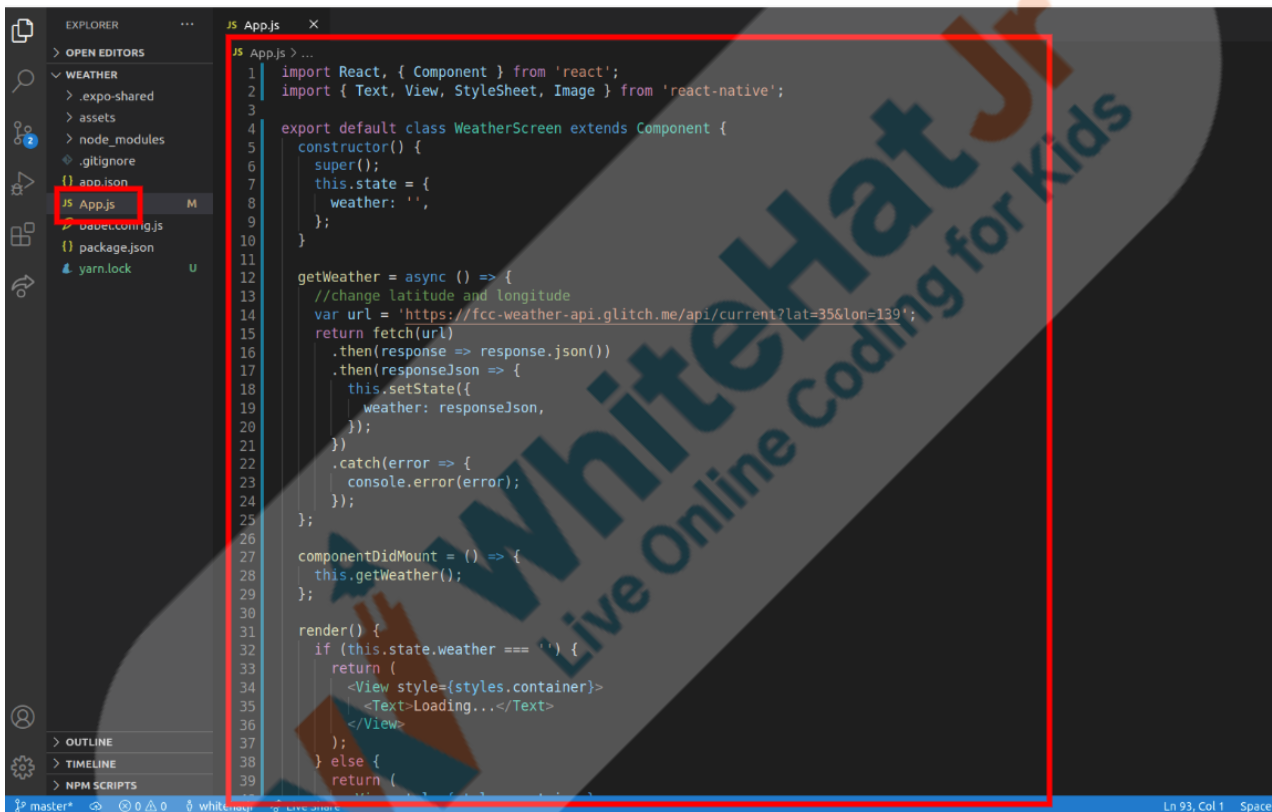
📦 Your project is ready!

To run your project, navigate to the directory and run one of the follo

- cd weatherApp
- npm run android
- npm run ios # you need to use macOS to build the iOS project - use th
out a Mac
- npm run web

C:\Users\hp\Desktop>
  
```

| | | |
|--|---|---|
| | Now your project folder is ready. open the project folder in your editor. | Student opens the project folder in the editor. |
| | We'll write code for the weather app in the editor. We'll write code in our App.js file. | Student opens the App.js file and writes code to create a small weather app which displays projected weather information. |

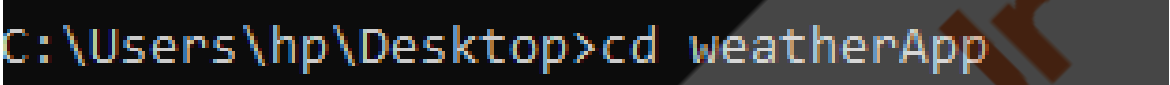


```

1  import React, { Component } from 'react';
2  import { Text, View, StyleSheet, Image } from 'react-native';
3
4  export default class WeatherScreen extends Component {
5    constructor() {
6      super();
7      this.state = {
8        weather: '',
9      };
10   }
11
12   getWeather = async () => {
13     //change latitude and longitude
14     var url = 'https://fcc-weather-api.glitch.me/api/current?lat=35&lon=139';
15     return fetch(url)
16       .then(response => response.json())
17       .then(responseJson => {
18         this.setState({
19           weather: responseJson,
20         });
21       })
22       .catch(error => {
23         console.error(error);
24       });
25   };
26
27   componentDidMount = () => {
28     this.getWeather();
29   };
30
31   render() {
32     if (this.state.weather === '') {
33       return (
34         <View style={styles.container}>
35           <Text>Loading...</Text>
36         </View>
37       );
38     } else {
39       return (

```

| | | |
|--|--|---|
| | Get Json data from the API Change the state of the weather using the data Use the weather state and display it on the App User Interface | The student writes code to create a small weather app which displays projected weather information. |
| | Now, to test the output open your terminal again and navigate to the project folder. | |

| | | |
|--|---|---|
| | Do you remember how to navigate to different folders on your computer using the terminal? | ESR: Yes. Using cd command. 'cd' stands for 'change directory'. |
| | Awesome. Let's do it. Note: The exact folder might be different for the student. | The student navigates to the directory where there is the project folder |
|  | | |
| | <p>To run the project we'll use a command : npx expo start --tunnel</p> <p>This will start your project. It will generate a QR code. You can scan the QR code on an expo client installed on your phone to open the app.</p> <p>Note: Your computer and your phone must be connected to the same network for this to work.</p> | The student starts and tests the project on their phone using an expo-client. |

```
cmd Select C:\windows\system32\cmd.exe

C:\Users\hp\Desktop>cd weatherApp

C:\Users\hp\Desktop\weatherApp>npx expo start --tunnel
Starting project at C:\Users\hp\Desktop\weatherApp
Starting Metro Bundler
Tunnel connected.
Tunnel ready.



> Metro waiting on exp://udafa-u.bijoya1010.19000.exp.direct:80
> Scan the QR code above with Expo Go (Android) or the Camera app (iOS)

> Press a | open Android
> Press w | open web



> Press j | open debugger
> Press r | reload app
> Press m | toggle menu

> Press ? | show all commands

Logs for your project will appear below. Press Ctrl+C to exit.
```

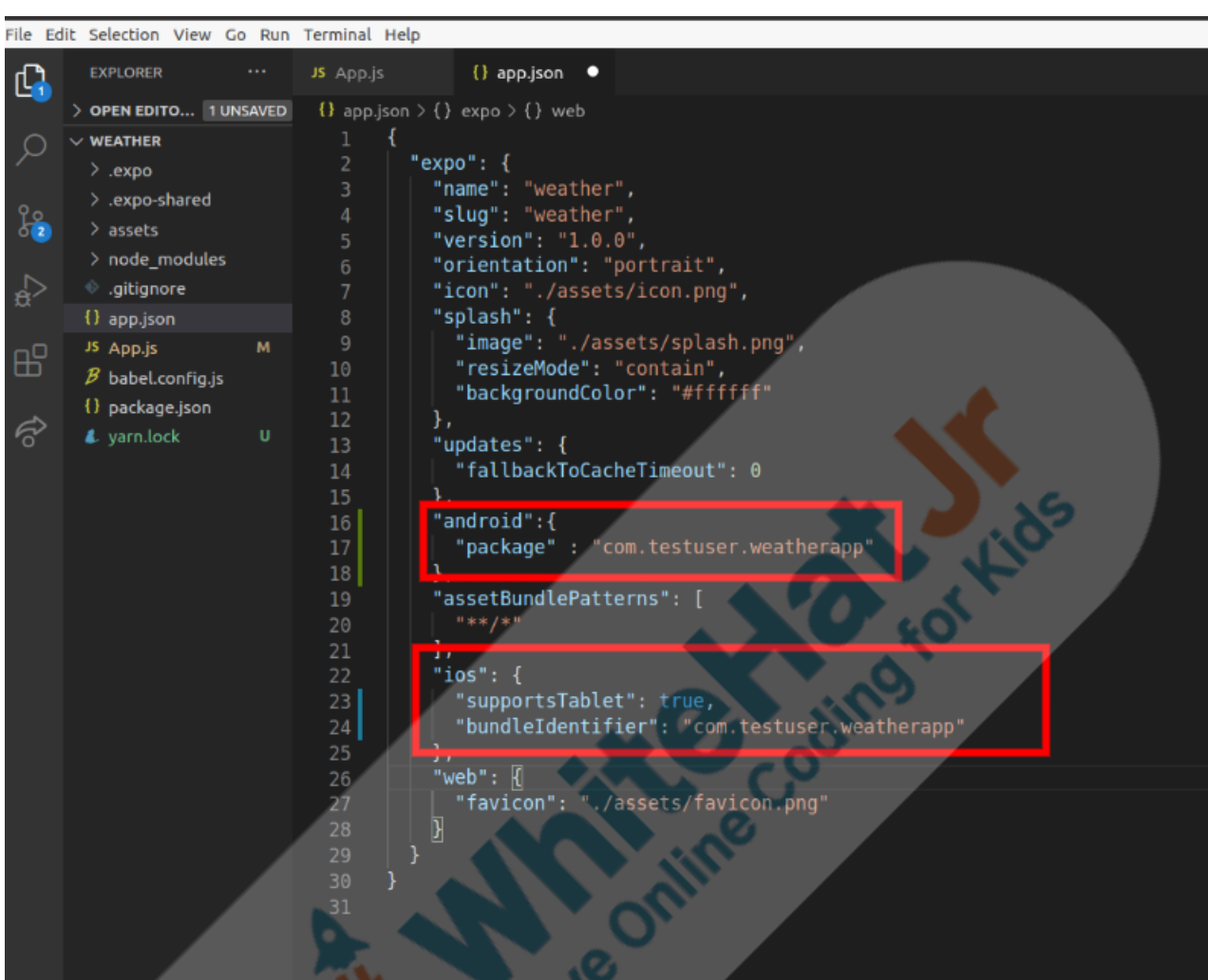
STUDENT-LED ACTIVITY - 15 mins

- Ask Student to press ESC key to come back to panel
- Guide Student to start Screen Share
- Teacher gets into Fullscreen

| ACTIVITY | | |
|---|---|--|
| <ul style="list-style-type: none"> Build the aab or ipa file. | | |
| <div>  </div> <p>Teacher starts slideshow for slide 17 and 18.</p> | | |
| | Now it's your turn. Please share your screen with me. | |
| <div>  </div> <p>Teacher ends slideshow</p> | | |
| Step 3: Student-Led Activity (15 min) | <p>Isn't this amazing!</p> <p>Now let's quickly learn how to build aab or ipa files from this project.</p> <p>aab stands for Android App bundle. aab is a publishing format that includes all the resources and compiled code for an application. (aab file can published on playstore, but cannot be directly installed on your phone)</p> <p>Before building the aab or ipa, we need to add a unique identifier for playstore and appstore to remember our app with. This is done using a reverse web domain name inside app.json file- since each user's web domain of each user will be different and unique. You can use any dummy domain name for now.</p> | <p>The student runs the build command.</p> |

| | | |
|--|---|--|
| | <p>Now press Ctrl + C to stop the metro bundler you ran using expo start.</p> <p>Go to the project folder →</p> <p>For building aab or ipa file, run the following commands -</p> <ol style="list-style-type: none"> 1. Install the latest EAS CLI. EAS CLI is the command-line app that you will use to interact with EAS services from your computer. npm install -g eas-cli 2. Login to your expo account using the following command- eas login 3. Configure your project for android or iOS- eas build:configure 4. Build your project - eas build --platform android or, eas build --platform ios <p>Note 1: There might be an error like "unable to resolve react-native-gesture handler." This means that the above library did</p> | |
|--|---|--|

| | | |
|--|---|--|
| | <p>not get correctly installed.</p> <p>Run: npx expo install react-native-gesture-handler</p> <p>This will install the above package. And then you can run build commands again.</p> <p>Note 2: For ios build ,the app icon shouldn't be transparent so make sure your app icon is not transparent and also you will need apple id and password for your paid developer account. It will authenticate the developer account. The Student will have to create a paid developer account for this purpose.</p> <p>Note 3: Expo builds aab on a shared server machine. Build will fail if one of expo's server machines is not available for building.</p> | |
|--|---|--|



```

1  {
2    "expo": {
3      "name": "weather",
4      "slug": "weather",
5      "version": "1.0.0",
6      "orientation": "portrait",
7      "icon": "./assets/icon.png",
8      "splash": {
9        "image": "./assets/splash.png",
10       "resizeMode": "contain",
11       "backgroundColor": "#ffffff"
12     },
13     "updates": {
14       "fallbackToCacheTimeout": 0
15     },
16     "android": {
17       "package": "com.testuser.weatherapp"
18     },
19     "assetBundlePatterns": [
20       "**/*"
21     ],
22     "ios": {
23       "supportsTablet": true,
24       "bundleIdentifier": "com.testuser.weatherapp"
25     },
26     "web": {
27       "favicon": "./assets/favicon.png"
28     }
29   }
30 }
31

```

```

C:\Users\hp\Desktop\weatherApp>npm install -g eas-cli
changed 351 packages, and audited 352 packages in 6s

35 packages are looking for funding
  run `npm fund` for details

2 high severity vulnerabilities

Some issues need review, and may require choosing
a different dependency.

Run `npm audit` for details.

C:\Users\hp\Desktop\weatherApp>eas login
Log in to EAS
✓ Email or username ... bijoya1010
✓ Password ... *****
Logged in

C:\Users\hp\Desktop\weatherApp>eas build -p android
✓ Generated eas.json
EAS project not configured.
✓ Would you like to automatically create an EAS project for @bijoya1010/weatherApp? ... yes
✓ Created @bijoya1010/weatherApp (https://expo.dev/accounts/bijoya1010/projects/weatherApp) on Expo
✓ Linked local project to EAS project 6a3b1042-a136-4297-b786-a4785bf6b530


Android application id Learn more: https://expo.fyi/android-package
✓ What would you like your Android application id to be? ... com.bijoya1010.weatherApp
✓ Using remote Android credentials (Expo server)
✓ Generate a new Android Keystore? ... yes
Detected that you do not have keytool installed locally.
✓ Generating keystore in the cloud...
✓ Created keystore

Compressing project files and uploading to EAS Build. Learn more: https://expo.fyi/eas-build-archive
✓ Uploaded to EAS
  
```

The build command takes a while.
You can visit the build link given in the terminal to see the progress.


Once the build is finished, you can download the .aab file and publish it on playstore.

After some time, you can see the link to the aab file. You can click on it to

| | | |
|---|--|--|
| | download and publish it on the playstore. | |
| <pre>File Edit View Search Terminal Help Unable to find an existing Expo CLI instance for this directory, starting a new one... (node:30009) [DEP0066] DeprecationWarning: OutgoingMessage.prototype._headers is deprecated Starting Metro Bundler on port 19001. Tunnel ready. Publishing to channel 'default'... Building iOS bundle Building Android bundle Building JavaScript bundle [=====] 100%Finished building JavaScript bundle in 70356ms. Analyzing assets Building JavaScript bundle [=====] 100%Finished building JavaScript bundle in 78795ms. Finished building JavaScript bundle in 1924ms. Uploading assets Building JavaScript bundle [=====] 100%Finished building JavaScript bundle in 1743ms. No assets changed, skipped. Uploading JavaScript bundles Published Your URL is https://exp.host/@rajeevtf1/snack-ec4d012b-6a11-4113-8661-f141246ab09e > Closing Expo server > Stopping Metro bundler Checking if this build already exists... Build started, it may take a few minutes to complete. You can check the queue length at https://expo.io/turtle-status You can monitor the build at https://expo.io/builds/b1f2f085-8a3e-4c06-a29f-e86e68ebce26 Waiting for build to complete. You can press Ctrl+C to exit. ✓ Build finished. Successfully built standalone app: https://expo.io/artifacts/6e805551-e84f-43b7-8d52-a944c7fdb0e6 rajeev@atlantis ~/quizbuzzerfinal></pre> | | |
| Teacher Guides Student to Stop Screen Share | | |
| WRAP-UP SESSION - 5 Mins | | |
| <div>Teacher starts slideshow  from slide 19 to slide 29</div> | | |
| Activity details | Solution/Guidelines | |
| <p>Run the presentation from slide 19 to slide 29</p> <p>Following are the wrap-up session deliverables:</p> <ul style="list-style-type: none">● Explain the facts and trivias● Next class challenge● Project for the day● Additional Activity | <p>Guide the student to develop the project and share with us.</p> | |
| Quiz time - Click on in-class quiz | | |

| Question | | Answer |
|--|---|--------------|
| Which function is used - to get the JSON data from the API? A. json() B. fetch() C. response() D. url() | | B |
| All the libraries that we used in snack including react, react-native, Firebase, react-navigation, they all come as node packages called as _____ A. node B. expo C. expo-cli D. npm | | D |
| Which of the following commands is used to run the project in expo? A. expo init B. expo run C. expo start D. expo client | | C |
| | | |
| FEEDBACK <ul style="list-style-type: none"> Encourage students to explore more of Expo documentation on what is available in Expo environment. | | |
| | Amazing! Are you finding this journey of building react native apps exciting? | ESR: Yes! |
| | Awesome. In the next class we will work on another case study to create an App which solves a practical problem. | |

| | | |
|--|--|---|
| | <p>While working on the next app, we will learn about many more components which are available in React native using which you can create professional grade application!</p> <p>I am very excited. Hope you are too.</p> | |
| | <p>You get a “hats off”.</p> <p>Till next class then. See you. Bye!</p> | <p>Make sure you have given at least 2 Hats Off during the class for:</p> <div>Creatively Solved Activities +10</div> <div>Great Question +10</div> <div>Strong Concentration +10</div> |
| | <p>Congratulations! You have achieved a new milestone.</p> <p>In this Capstone project, your goal is to apply the learnings and outcomes from previous classes and get started on publishing the Student Attendance App.</p> | |
| Project Pointers and Cues (5 min) | <p>* This Project will take only 45 mins to complete. Motivate students to try and finish it immediately after the class.</p> <p>App Publishing and Local environment setup</p> <p>Goal of the Project:</p> | |

| | | |
|---|---|--|
| | <p>In the class you have learnt to install expo on a local machine and generate an aab or ipa for publishing the app.</p> <p>In this project, we shall practice the same concept and generate an aab (for android) and ipa (for iOS) for the app you already made called “Student Attendance App”.</p> <p>Story:</p> <p>You have already helped the school team in creating an application where teachers can see the list of students, and mark present/absent for a particular date. You are finally done with the attendance app. You want the others also to try and test this app.</p> <p>I am very excited to see your project solution and I know you both will do really well.</p> <p>Bye Bye!</p> | |
| <div> <div>Teacher Clicks</div> <div>✕ End Class</div> </div> | | |
| <div> <div>Teacher ends slideshow</div>  </div> | | |
| Additional Activities | <p>Encourage the student to write reflection notes in their reflection journal using markdown.</p> <p>Use these as guiding questions:</p> | <p>The student uses the markdown editor to write her/his reflection in a reflection journal.</p> |

| | | |
|--|---|--|
| | <ul style="list-style-type: none"> • What happened today? <ul style="list-style-type: none"> - Describe what happened - Code I wrote • How did I feel after the class? • What have I learned about programming and developing games? • What aspects of the class helped me? What did I find difficult? | |
|--|---|--|

| | | |
|-------------------------|---|--|
| Project Overview | <ol style="list-style-type: none"> 1) Guide the student towards starting/continuing the after-class project for the class. 2) Check for student progress in previous project/s. 3) Resolve any student doubts over projects. | Student engages with the teacher over the project. |
|-------------------------|---|--|

| Links | | |
|--------------------|-------------------------|---|
| Activity | Activity Name | Links |
| Teacher Activity 1 | Expo installation steps | https://expo.io/learn |
| Teacher Activity 2 | Node Installation Link | https://nodejs.org/en/ |
| Teacher Activity 3 | Quiz Buzzer App Link | https://snack.expo.io/@rajeevtfti/3eff2d |
| Teacher Reference | Weather App | https://snack.expo.io/@rajeevtfti/868223 |

| | | |
|-----------------------------------|--|---|
| Student Activity 1 | Expo installation steps | https://expo.io/learn |
| Student Activity 2 | Node Installation Link | https://nodejs.org/en/ |
| Student Activity 3 | Quiz Buzzer App Link | https://snack.expo.io/@rajeevt/3eff2d |
| Student Activity 4 | EAS build documentation | https://docs.expo.dev/build/setup/ |
| Project Solution | App Publishing and Local environment setup | Solution depends on students' submission. |
| Teacher Reference visual aid link | Visual aid link | https://s3-whjr-curriculum-uploads.whjr.online/04783692-6e96-4bc9-89aa-95c5afdc7ddc.html |
| Teacher Reference In-class quiz | In-class quiz | https://s3-whjr-curriculum-uploads.whjr.online/67d76307-3527-4260-a783-3d5e6c35f519.pdf |