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 apk 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	● Set up expo on the local machine. ● Generate apk or files for apps to be published on playstore. ● Build an app to keep track of certain trading stocks.	
Resources Required	● Teacher Resources ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Android/iOS Smartphone with Expo App installed ● Student Resources ○ 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**

CONTEXT

- **Setting up expo on the local environment.**

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Hi! Big day today! Welcome to the Capstone Class.	ESR: We are going to learn to

	Do you recall what we are going to do today?	build apk or ipa files to be published on playstore/appstore.
	Yes!! You will then be able to build and publish as many apps as you want during the course while you are learning more about React Native. We will also be doing two more things. So far, you have been building the react native app on an online editor - expo snack. We will now install expo on our local machine so that you can build apps from your local machine. We will then be building a small but useful app on our local machine. Today, we will build a weather app to capture the weather of the current location.	The student listens and asks questions.
	We have a lot to cover in today's class. So, without wasting any more time, let's get started.	
Teacher Asks student to screen share		
<u>CHALLENGE</u> ● **Build apk or ipa files for apps to be published on the playstore or appstore.**		
Step 2: Teacher-led Activity (15 min)	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.	ESR: API is a service which gives us some data based on our query. We use 'fetch()' to get data from API in javascript.

	Do you remember what an API is? Do you remember how to get data from API in javascript?	
	Correct! Now open a new expo-snack. You can open your App.js file to start writing your code.	Student opens a new expo snack
	First we'll write a getWeather() function which will get the json data from the api. <Teacher helps the student write the getWeather() function>	Student code to write the getWeather() function which will get the json data from the api and set it to the state. -Student uses fetch() to get the json data from the api and sets it to the weather in the state

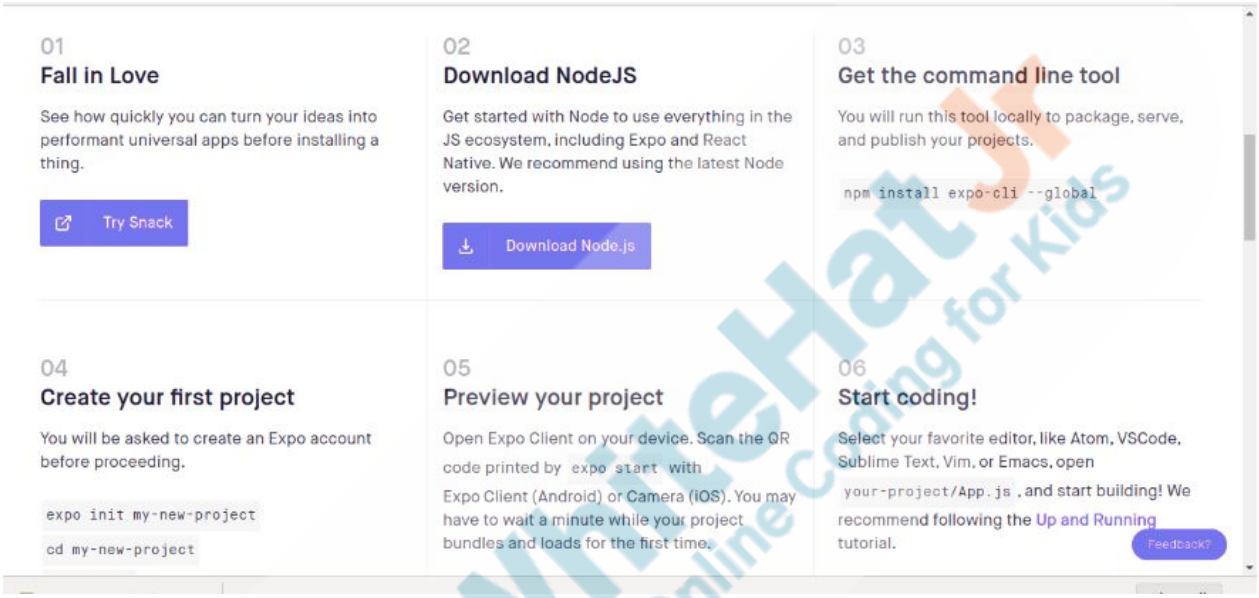
```
export default class WeatherScreen extends Component {
  constructor() {
    super();
    this.state = {
      weather: '',
    };
  }

  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);
      });
  };
```

	Now we need to show the data on the interface. we'll show loading message while our function is running and getting 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 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.	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

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

	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. Teacher opens **Teacher Activity 1.**	The student opens **Student Activity 1.**
		
	First, we will install node on our system. So far, we have only run javascript inside a browser. Node allows us to run javascript outside our browser as well. Let's follow the instructions to install node. **For Windows users:** 1. Download node directly from the given link.	The student installs node and checks the node --version.

	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. **For Mac users:** 1. Install homebrew first. Homebrew is a package manager for your operating system. It helps you in easily installing programs from the terminal. To install homebrew, open your terminal and type: /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)" Note: You might have to add "sudo" before the command if you do not have permission to install packages on your OS. "sudo" stands for "do as a super user". You might have to run: sudo /bin/bash -c “$(curl -fsSL https://raw.githubsercontent.com/Homebrew/install/master/install.sh)”	

2. Now install node. On your terminal type:
   sudo brew install node

3. Check if node is installed on your system by typing in the terminal:
   node --version.

For Ubuntu users:

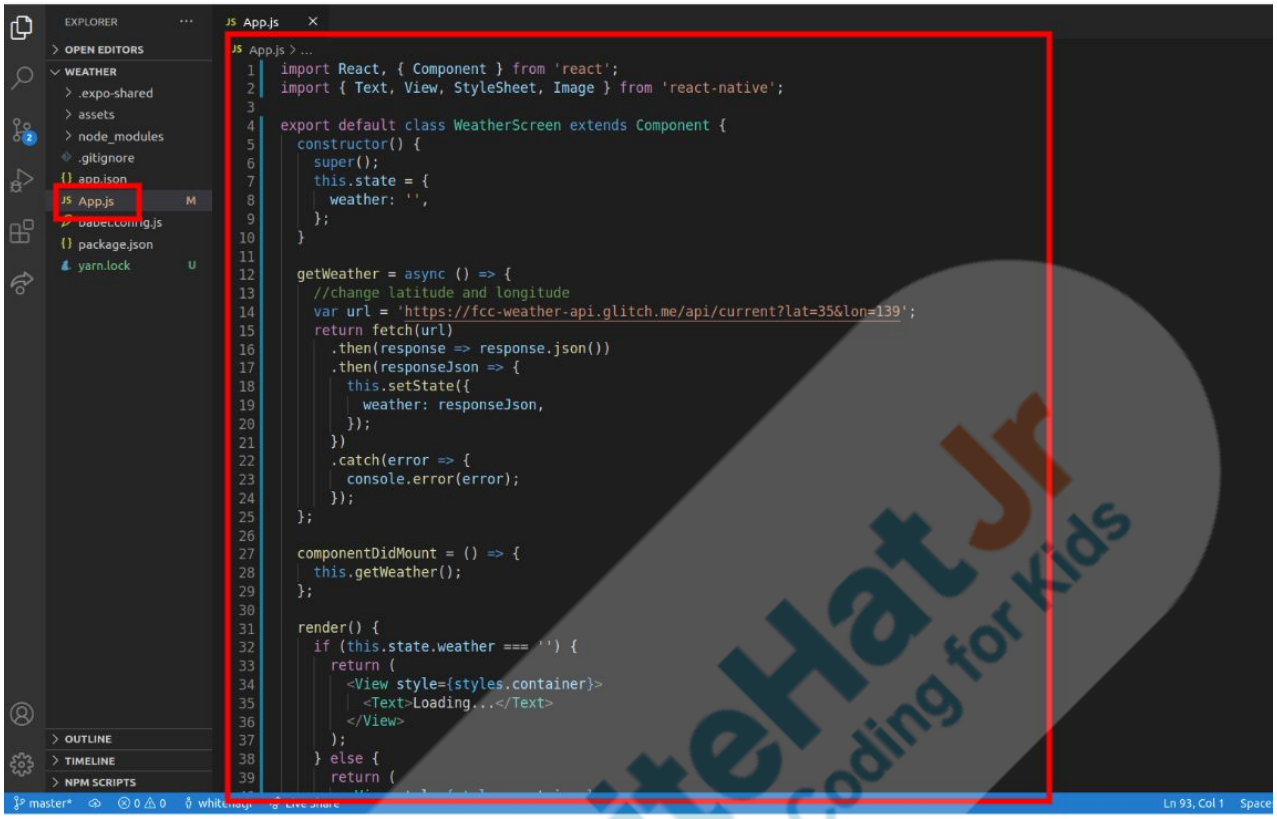
1. Open your terminal and type:
   sudo apt install node
   This will install node on your system.
2. Check the node installation by typing:
   node --version

	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.
		
	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	The student installs ‘expo-cli’ on their system. Note: Installing ‘expo-cli’ can take some time.

	quickly getting started with building react native apps. To install ‘expo-cli’, on your terminal type: **npm install expo-cli --global** 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 use expo anywhere on your system. Without global tag, expo will be installed only in the folder in which you are running the command.	
	Alright ! We have ‘expo’ installed on our system now. Now let’s start coding in the the local environment. First we need to create a new project .To create a new project write **expo init <project name>** on your terminal. Choose a blank template , press enter and wait for some time until the process is finished.	Student opens the terminal and writes **expo init weather app.** **-**Then selects the blank template , presses enter button and waits till the process is completed.

```
:~$ expo init weather app

   There is a new version of expo-cli available (3.22.3).
   You are currently using expo-cli 3.21.5
   Install expo-cli globally using the package manager of your choice;
   for example: `npm install -g expo-cli` to get the latest version

? Choose a template: expo-template-blank

Using Yarn to install packages. You can pass --npm to use npm instead.

✔ 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 following yarn commands.

- cd weather
- yarn start # you can open iOS, Android, or web from here, or run them directly with the commands be
low.
- yarn android
- yarn ios # requires an iOS device or macOS for access to an iOS simulator
- yarn web
```

	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.



	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
`cd weather/`		
	To run the project we'll use a command : **expo start** 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. Note: Your computer and your phone must be connected to the same network for this to work.	The student starts and tests the project on their phone using an expo-client.

```
ashura@zeros:~/weather$ expo start

   There is a new version of expo-cli available (3.22.3).
   You are currently using expo-cli 3.21.5
   Install expo-cli globally using the package manager of your choice;
   for example: `npm install -g expo-cli` to get the latest version

Starting project at /home/ashura/weather
Expo DevTools is running at http://localhost:19002
Opening DevTools in the browser... (press shift-d to disable)
Starting Metro Bundler on port 19001.

  exp://192.168.8.120:19000
```

	Now it's your turn. Please share your screen with me.	

	• **Ask Student to press ESC key to come back to panel** • **Guide Student to start Screen Share** • **Teacher gets into Fullscreen**	
	ACTIVITY • **Build the apk file.**	
Step 3: Student-Led Activity (15 min)	Isn't this amazing! Now let's quickly learn how to build apk or ipa files from this project. Before building the apk 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. Now press Ctrl + C to stop the metro bundler you ran using expo start. In the same folder, run the following commands - For building apk, run the command - sudo expo build:android For building ipa, run the command - sudo expo build:ios Note 1: There might be an error like "unable to resolve react-native-gesture handler."	The student runs the build command.

	This means that the above library did not get correctly installed. Run: npm install react-native-gesture-handler This will install the above package. And then you can run build commands again. 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. Note 3: Expo builds apk on a shared server machine. Build will fail if one of expo's server machines is not available for building.	

File Edit Selection View Go Run Terminal Help

EXPLORER

OPEN EDITO... 1 UNSAVED

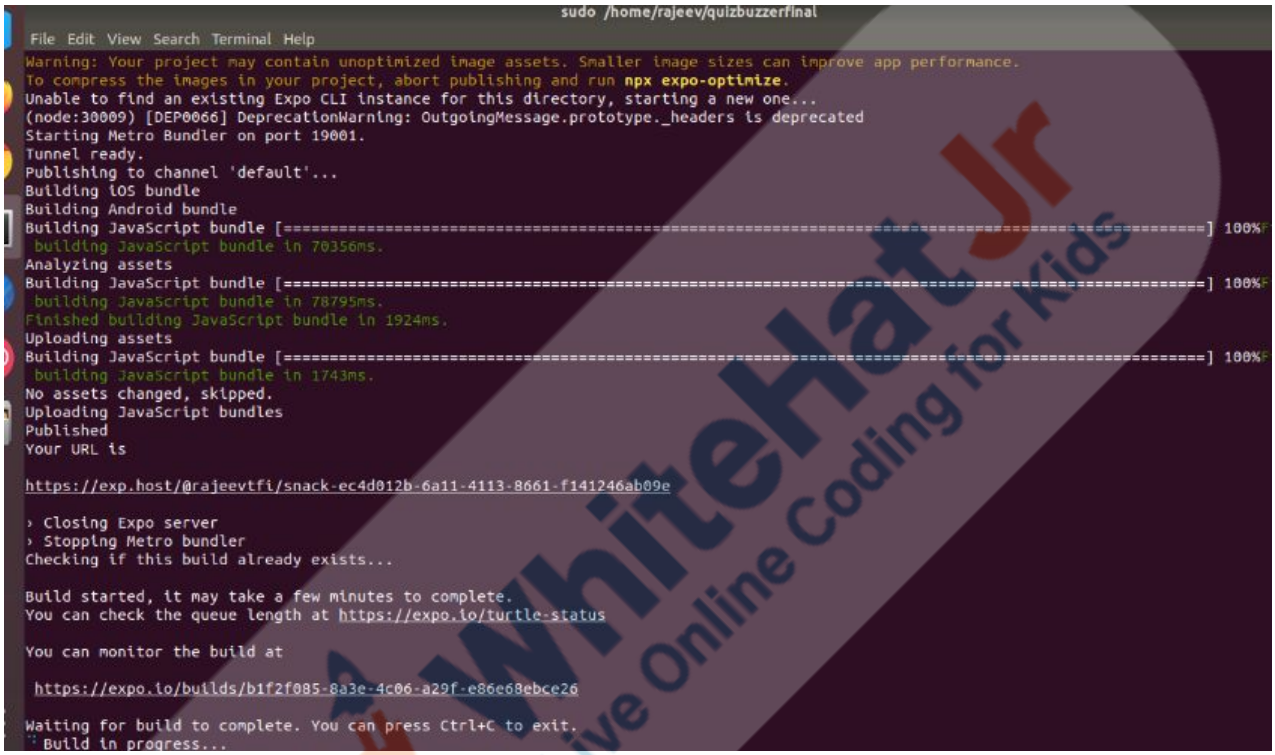
WEATHER
- .expo
- .expo-shared
- assets
- node_modules
- .gitignore
- app.json
- App.js M
- babel.config.js
- package.json
- yarn.lock U

App.js | app.json

app.json > {} expo > {} web

```
{
  "expo": {
    "name": "weather",
    "slug": "weather",
    "version": "1.0.0",
    "orientation": "portrait",
    "icon": "./assets/icon.png",
    "splash": {
      "image": "./assets/splash.png",
      "resizeMode": "contain",
      "backgroundColor": "#ffffff"
    },
    "updates": {
      "fallbackToCacheTimeout": 0
    },
    "android":{
      "package" : "com.testuser.weatherapp"
    },
    "assetBundlePatterns": [
      "**/*"
    ],
    "ios": {
      "supportsTablet": true,
      "bundleIdentifier": "com.testuser.weatherapp"
    },
    "web": {
      "favicon": "./assets/favicon.png"
    }
  }
}
```

```
ashura@zeros:~/weather$ expo build:android

   There is a new version of expo-cli available (3.22.3).
   You are currently using expo-cli 3.21.5
   Install expo-cli globally using the package manager of your choice;
   for example: `npm install -g expo-cli` to get the latest version

: Making sure project is set up correctly...
```

	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 apk directly from there.	
		
	After sometime, you can see the link of the apk file. You can click on it to download and install it.	The student can click on the link to download/install apk or ipa on the phone.

```
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%Finishe
 building JavaScript bundle in 70356ms.
Analyzing assets
Building JavaScript bundle [==================================================================================================] 100%Finishe
 building JavaScript bundle in 78795ms.
Finished building JavaScript bundle in 1924ms.
Uploading assets
Building JavaScript bundle [==================================================================================================] 100%Finishe
 building JavaScript bundle in 1743ms.
No assets changed, skipped.
Uploading JavaScript bundles
Published
Your URL is

https://exp.host/@rajeevtfi/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>
```

Teacher Guides Student to Stop Screen Share		
FEEDBACK • **Encourage students to explore more of Expo documentation on what is available in Expo environment.**		
Step 4: Wrap-Up (5 min)	Let's quickly summarize what we have learned today	We learned to install expo tools on our local machine. We also learned to build apk or ipa files. We used weather API to create a weather forecasting app on our local machine and test it!
	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. 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! I am very excited. Hope you are too.	
	You get a “hats off”. Till next class then. See you. Bye!	Make sure you have given at least 2 Hats Off during the class for:   
	Congratulations! You have achieved a new milestone. 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.	

Teacher Clicks ✖ End Class		
Additional Activities	Encourage the student to write reflection notes in their reflection journal using markdown. Use these as guiding questions: ● What happened today? - 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?	The student uses the markdown editor to write her/his reflection in a reflection journal.

Project Overview	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 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/@rajeevtfi/3eff2d
Teacher Reference	Weather App	https://snack.expo.io/@rajeevtfi/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/@rajeevtfi/3eff2d