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	• Set up expo on the local machine. • Generate aab or ipa files for apps to be published on playstore or appstore.	

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**
WARM-UP SESSION - 5 mins		
Teacher starts slideshow  **from slides 1 to 16** Refer to speaker notes and follow the instructions on each slide.		
Activity details		**Solution/Guidelines**

Hi, how have you been? Are you excited to learn something new? ***Run the presentation from slide 1 to slide 4.*** **The following are the warm-up session deliverables:** • Reconnect with previous class topics. • Warm-Up quiz session.	**ESR**: Varied Response. Click on the slide show tab and present the slides.
QnA Session	
Question	**Answer**
How do we resolve the error which comes when pressing the button repeatedly wherein the team name comes on the quiz master app repeatedly? A. By emptying the teams[] array B. By resetting the database C. By using enabled field in the database D. By using the status state in the app	**A**
getTime() function which is used to get the time from the client machine can be replaced by _____ ? A. New Date() B. firebase.database.ServerValue.TIMESTAMP C. firebase.ServerValue.TIMESTAMP D. firebase.database.TIMESTAMP	**B**
Continue the warm-up session	
Activity details	**Solution/Guidelines**
Run the presentation from slide 4 to slide 16 to set the problem statement. **The following are the warm-up session deliverables:** • **Review code from the previous class.** • **Setting up expo on the local environment.**	

Teacher ends slideshow 		
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
CHALLENGE • **Build the weather app.**		
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. Do you remember what an API is? Do you remember how to get data from an API in javascript?	**ESR:** API is a service which gives us some data based on our query. We use '**fetch()**' to get data from the 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
    // Here latitude = 40.7831 and  longitude = -73.9712
    var url = 'https://weather-l6tl.onrender.com/api/getCurrentWeather/40.7831/-73.9712';
    return fetch(url)
      .then(response => response.json())
      .then(responseJson => {
        this.setState({
          weather: responseJson.data,
        });
      })
      .catch(error => {
        console.error(error);
      });
  };
```

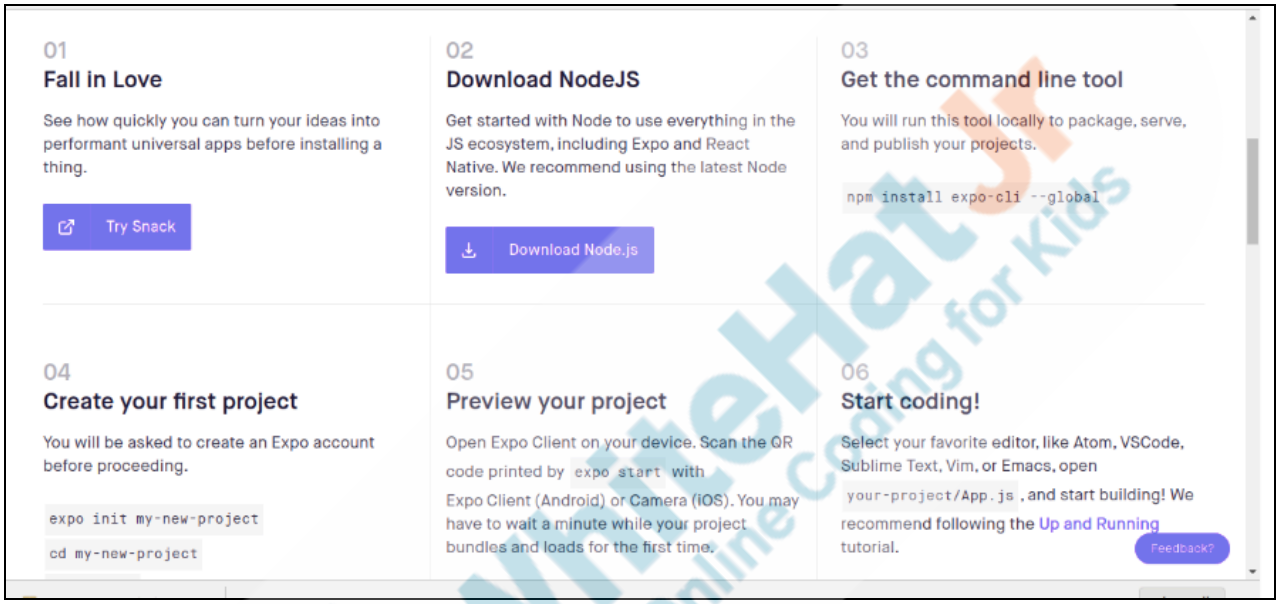
	Now we need to show the data on the interface. We'll show a loading message while our function is running and get the data from the API and once we have the data we'll show the forecast. To do that we'll write a if -else condition that if **this.state.weather === ” ”** then show the “Loading…” message 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

```
componentDidMount = () => {
  this.getWeather();
};

render() {
  console.log(this.state.weather)
  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.temperature}&deg;C
          </Text>
          <Text style={{ fontSize: 20, margin:10}}>
            humidity : {this.state.weather.humidity}
          </Text>
          <Text style={{fontSize: 20}}>
            {this.state.weather.weather_descriptions[0]}
          </Text>
        </View>
        </View>
      </View>
    );
  }
}
}
```

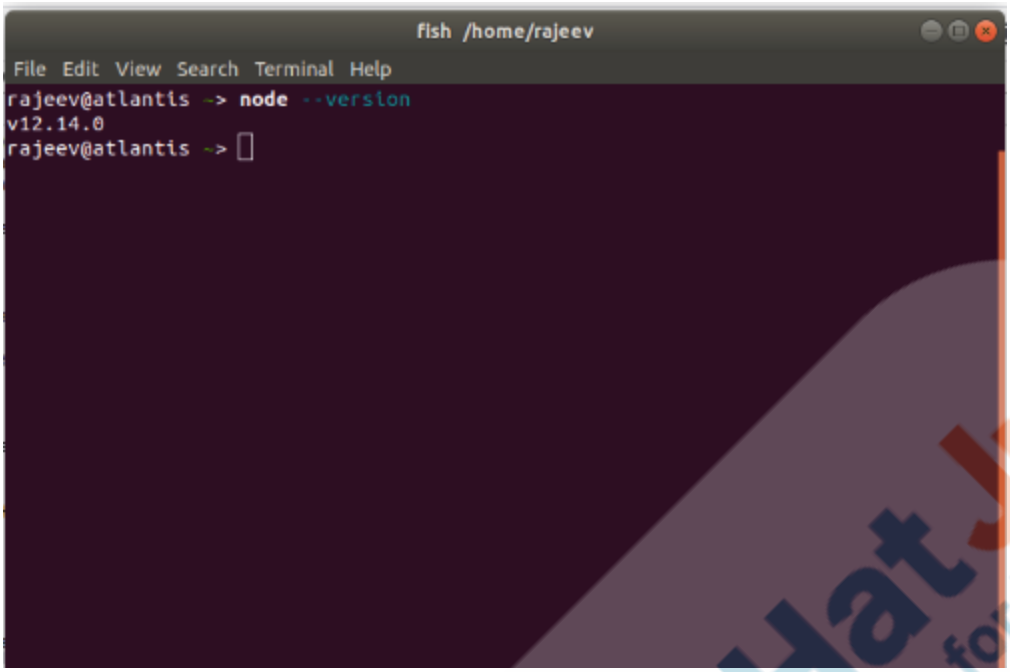
	Now let's check the output.	Student runs the code on the emulator and checks the output

	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.	-

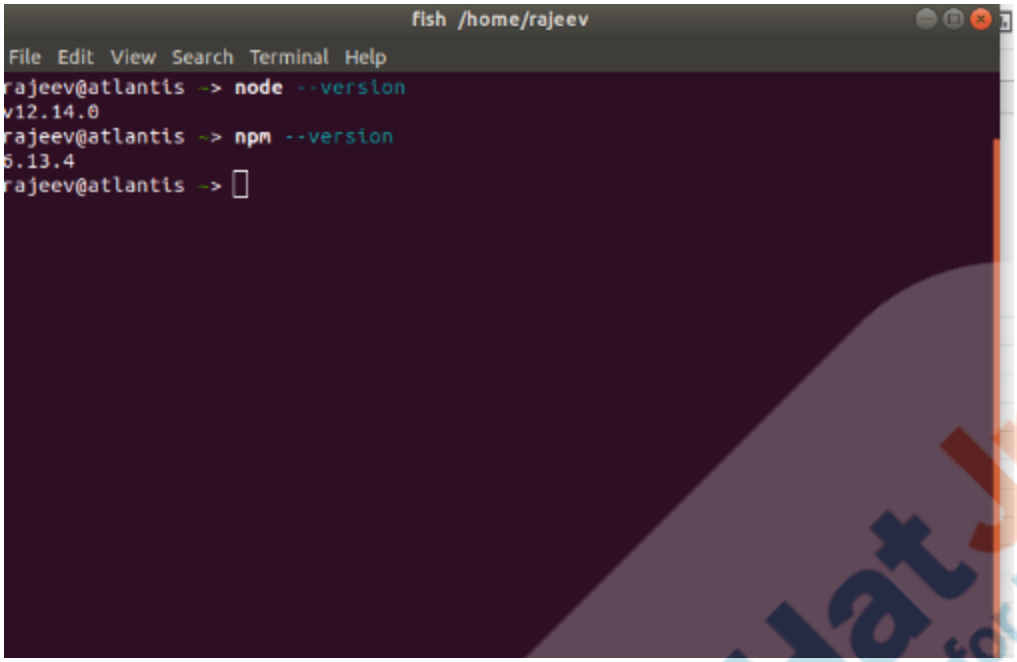
	We will be following instructions given in Expo documentation on its website to first install expo on our local machine. You can open your activity link to look at the instructions as well. Teacher opens **Teacher Activity 1.**	The student opens **Student Activity 1.**
		
	First, we will install node.js 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 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. **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.co	The student installs node from the given link in Student Activity 2 and checks the **node --version.**

	m/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	

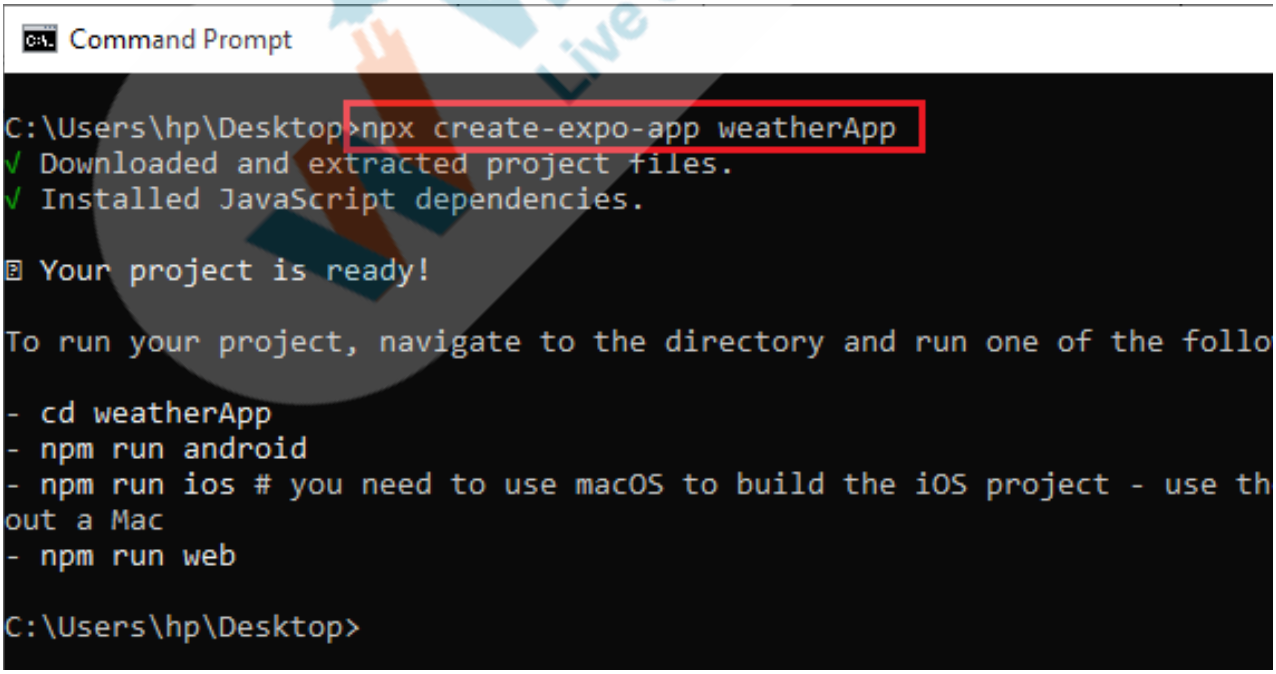


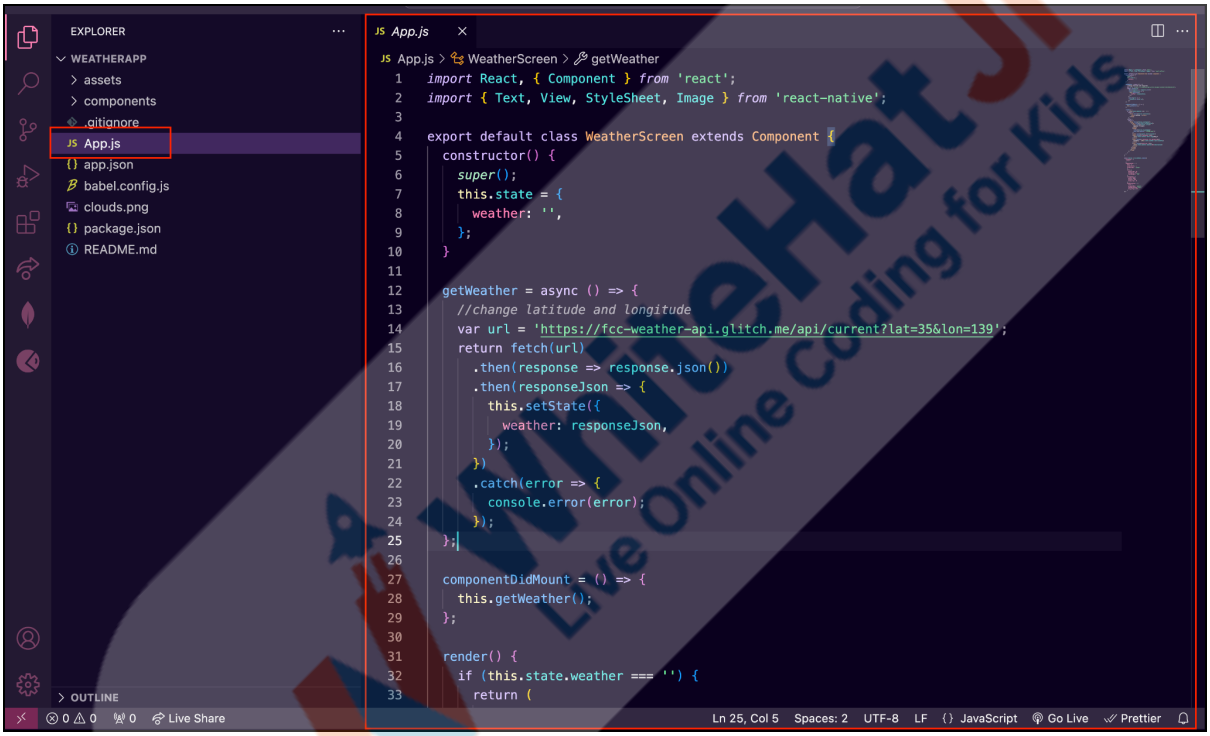
	When you install node, npm also gets installed. 'npm' stands for 'node package manager'. All the libraries that we used in snack including react, react-native, firebase, react-navigation, they all come as node packages. 'npm' helps us in installing and maintaining these packages. You will learn more 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.

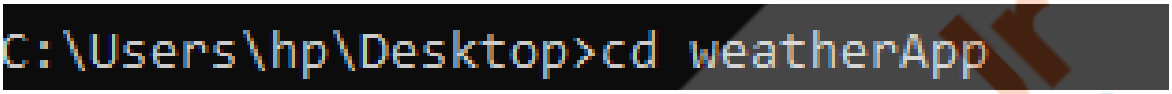


	Great! Now we will be using npm to install the expo command line tool. Expo command line tool or 'expo-cli' comes with many libraries and tools already installed which help us in quickly getting started with building react native apps. To install 'expo-cli', on your terminal type: **npm install expo-cli --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.	
	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 **npx create-expo-app <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 **npx create-expo-app weatherApp** -Then selects the blank template , presses enter button and waits till the process is completed.



	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
		
	To run the project we'll use a command : **npx expo start --tunnel** 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.

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**

<table>
<tr><td colspan="3"><u>ACTIVITY</u>
• Build the aab or ipa file.</td></tr>
<tr><td colspan="3">Teacher starts slideshow  for slide 17 and 18.</td></tr>
<tr><td></td><td>Now it's your turn. Please share your screen with me.</td><td></td></tr>
<tr><td colspan="3">Teacher ends slideshow  </td></tr>
<tr><td>Step 3: Student-Led Activity (15 min)</td><td>Isn't this amazing!

Now let's quickly learn how to build aab or ipa files from this project.

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)

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.</td><td>The student runs the build command.</td></tr>
</table>

Now press Ctrl + C to stop the metro bundler you ran using expo start.

Go to the project folder →

For building aab or ipa file, run the following commands -

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 -

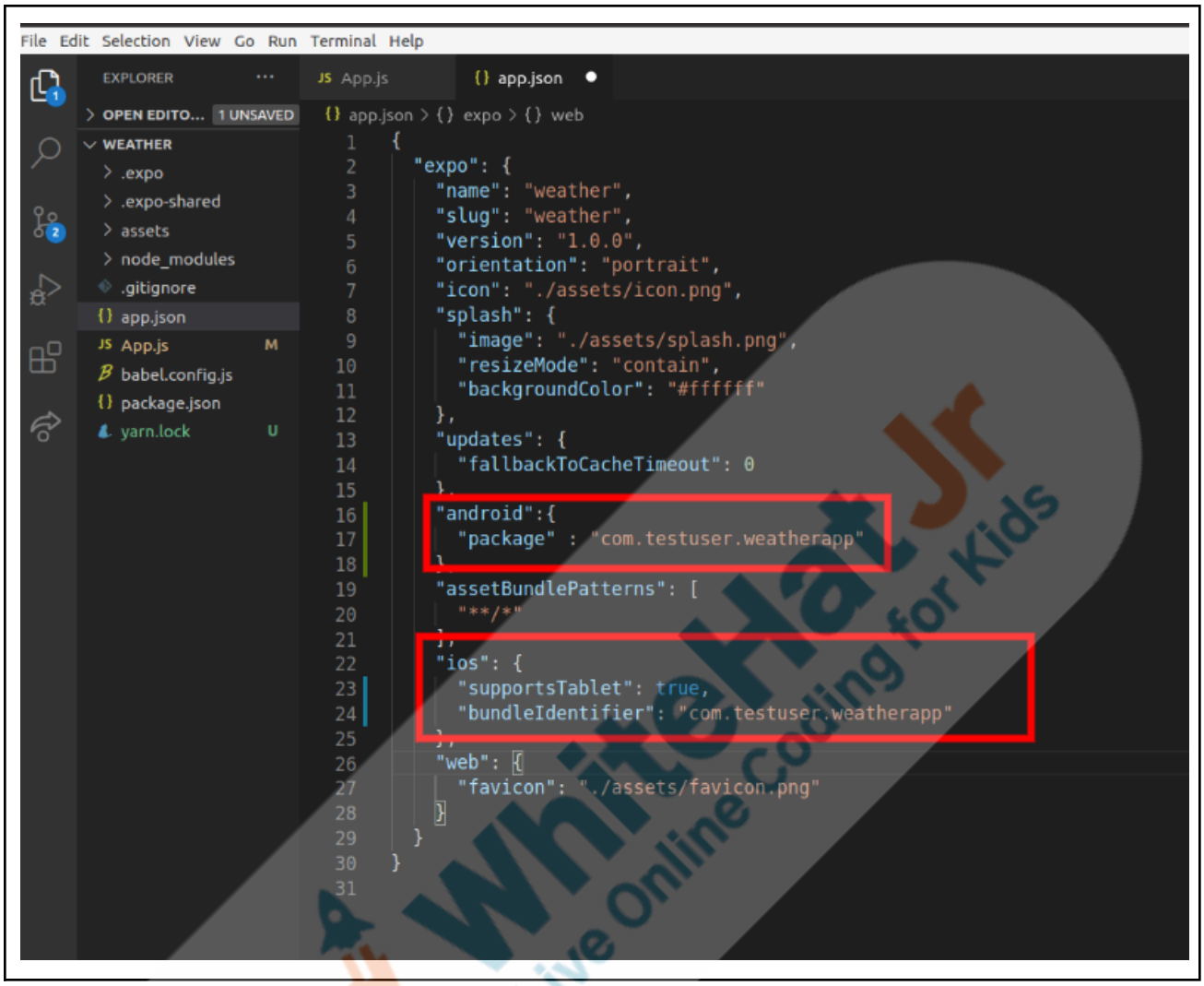
   eas build --platform android

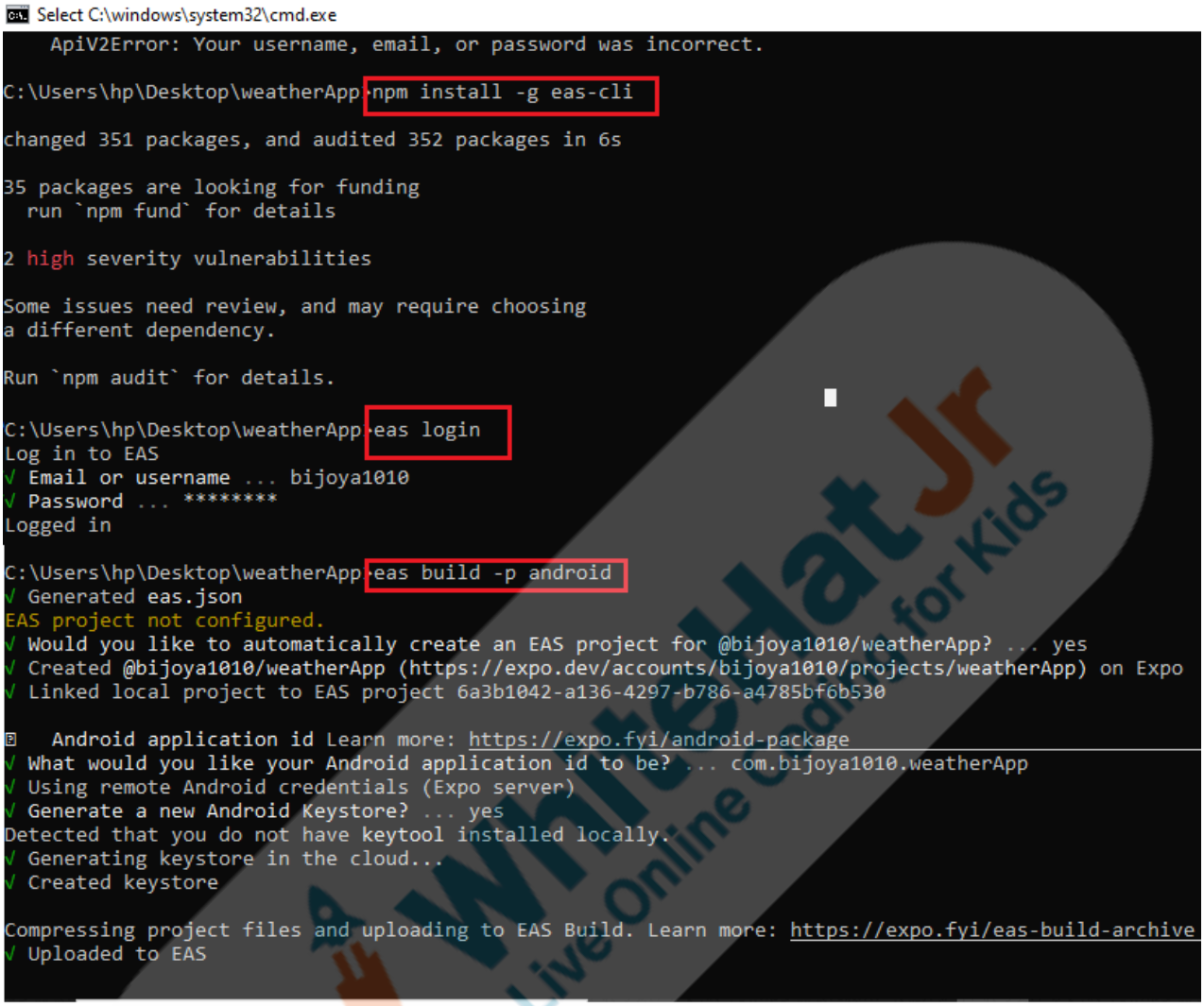
   or,

   eas build --platform ios

Note 1: There might be an error like **"unable to resolve react-native-gesture handler."** This means that the above library did

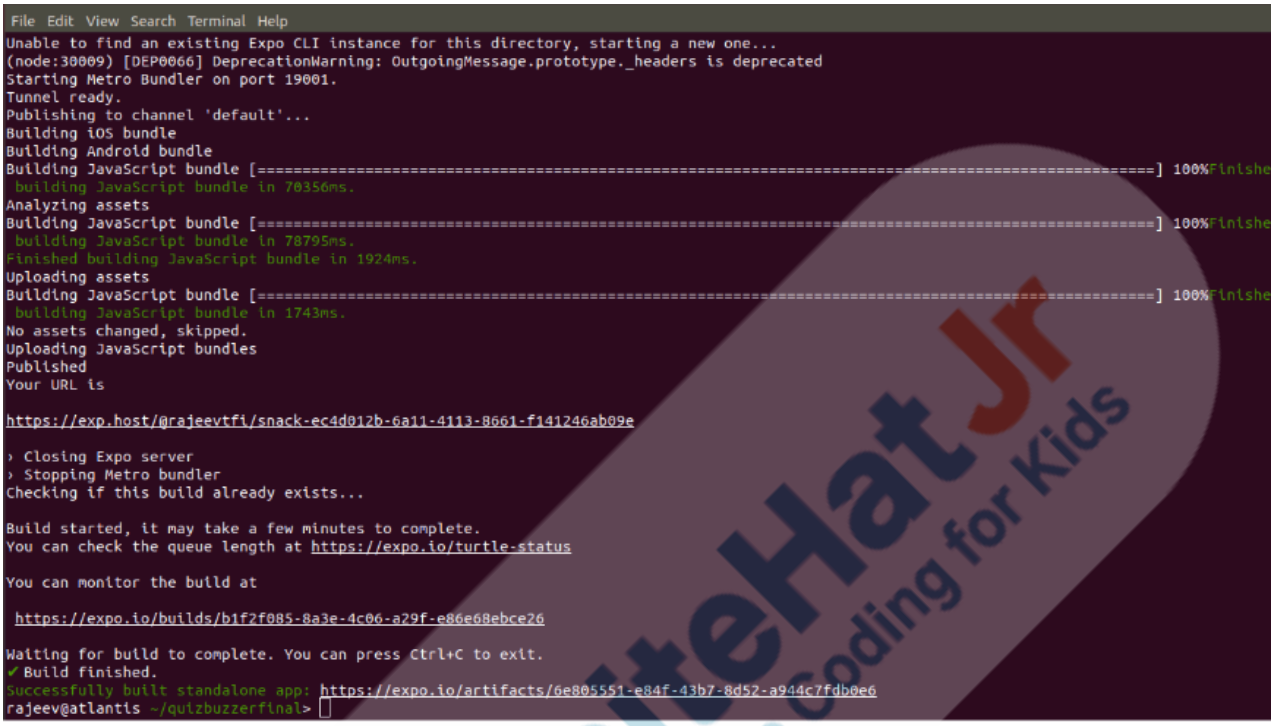
	not get correctly installed. Run: **npx expo install react-native-gesture-handler** This will install the above package. And then you can run build commands again. Note 2: For ios build ,the app icon shouldn't be transparent so make sure your app icon is not transparent and also you will need apple id 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 aab on a shared server machine. Build will fail if one of expo's server machines is not available for building.	





	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.	



Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 5 Mins

Teacher starts slideshow  **from slide 19 to slide 29**

Activity details	Solution/Guidelines
Run the presentation from slide 19 to slide 29 **Following are the wrap-up session deliverables:** • **Explain the facts and trivias** • **Next class challenge** • **Project for the day** • **Additional Activity**	Guide the student to develop the project and share with us.

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 • **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.	

	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.	
Project Pointers and Cues (5 min)	*** This Project will take only 45 mins to complete. Motivate students to try and finish it immediately after the class.** **App Publishing and Local environment setup** **Goal of the Project:**	

	In the class you have learnt to install expo on a local machine and generate an aab or ipa for publishing the app. 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”. **Story:** 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. I am very excited to see your project solution and I know you both will do really well. Bye Bye!	
Teacher Clicks ✖ End Class		
Teacher ends slideshow 		
Additional Activities	Encourage the student to write reflection notes in their reflection journal using markdown. Use these as guiding questions:	The student uses the markdown editor to write her/his reflection in a reflection journal.

	• 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?	

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 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.dev/@procodingclass/teacher-reference:-fixing-bugs
Teacher Reference	Weather App	https://snack.expo.dev/@procodingclass/weatherapp

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.dev/@procodingclass/teacher-reference:-fixing-bugs
Student Activity 4	EAS build documentation	https://docs.expo.dev/build/setup/
Project Solution	App Publishing and Local environment setup	The solution depends on students' submissions.
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