Mobile Application Development Internship Assignment

Name : Balaji Ramasamy

E-mail : balajir2222@gmail.com

Ph-: 8072601329

Section A: Multiple Choice Questions (20 marks)

1. What is React Native?

a) A cross-platform framework for building mobile apps using JavaScript

2. What is the purpose of the Flexbox layout in React Native?

a) To create a responsive design that adapts to different screen sizes

3. What is the difference between props and state in React Native?

a) Props are used to pass data between components, while state is used to manage data within a component

4. Which of the following is NOT a valid way to style a React Native component?

c) JavaScript styles

5. What is the purpose of the fetch() method in React Native?

a) To make HTTP requests to a server

6.What is the difference between setState() and forceUpdate() in React Native?

a) setState() updates the state of a component and triggers a re-render, while forceUpdate() re-renders a component without updating its state

7. Which of the following is used to handle user input in React Native?

d) All of the above

8. What is the purpose of the AsyncStorage module in React Native?

a) To store data permanently on the device

9. What is the purpose of the Animated API in React Native?

a) To create complex animations using JavaScript

10. Which of the following is used to navigate between screens in a React Native app?

d) All of the above

11. Create a new React Native app called "MyApp".

a) What command(s) would you use to create this app?

To create project(app) in React native use command :

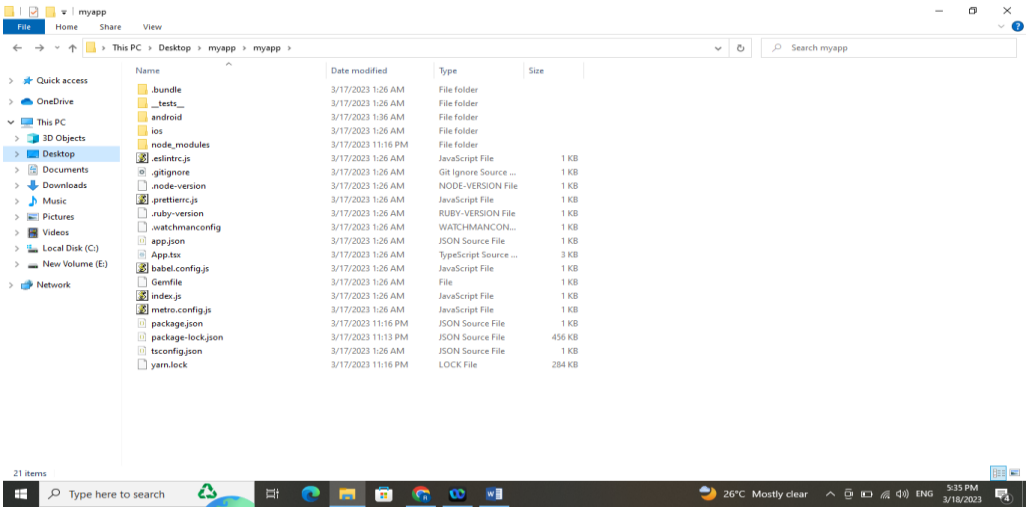
npx react-native@latest init MyApp

And the command that used to Build our app is

Go to the project and select directory and run the project.

npx react-native run-android

b) What is the directory structure of the app?



c) What file(s) would you modify to change the app's appearance?

To change the default appearance of the app edit app.tsx file which is created on appbuild.

App.tsx

12. Create a new component called "MyButton" that displays a button with the text "Clickme".

a) What props would you pass to this component to change the button's appearance?

The props used in a basic Button creation is,

Button text, onPress, type ,border ,size

b) What state would you use to handle clicks on the button?

Initially create constructor to access states

Example syntax is

```
Constructor(){
 super();
    this.state= {
 count :1,}}  **  for code see button.js in below page**
```

13. Create a new component called "MyList" that displays a list of items.

a) What props would you pass to this component to change the list's appearance?

There are two types of lists:

Flatlist and Sectionlist

The props using :

1.Data

2. renderitem.data

b) What data structure would you use to store the list items?

```
state = {
    names: [
      {
        id: 0,
name: 'Balaji' },   {
        id: 1,
        name: 'Susan',
      }]}
```

c) How would you render the list items?

```
render() {
    return (
      <View>
        {
          this.state.names.map((item, index) => (
          <TouchableOpacity
           key = {item.id}
           style = {styles.container}
           onPress = {() => this.alertItemName(item)}>
```

```
<Text style = {styles.text}>
 {item.name}
 </Text>
 </TouchableOpacity>
  ))
  }
  </View>
```

14. Write a function called "getWeather" that makes an HTTP GET request to the OpenWeatherMap API and returns the temperature for a given city.

a) What parameters would you pass to this function?

- City Name
- City ID
- Geographical Cordinates
- Zipcode

b) What is the URL for the OpenWeatherMap API?

'https://api.openweathermap.org/data/2.5'

c)How would you handle errors or exceptions in this function?

The problem may accurs with following statements

ApI call ,cityname,openweathermap

15. Create a new screen in your app called "ProfileScreen" that displays the user's profile information.

a) What navigation method would you use to navigate to this screen?

Use **REACT NAVIGATION** (navigate components) to navigate a new page in app.

Example :

```
Function ProfileScreen ({navigation})

{

Return(

<Button title=”balaji’s profile ”

Onpress={() =>

Navigation.navigate(‘profile’,{name : ‘Balaji’}) }

/>

);}
```

b) What props would you pass to this screen to display the user's information?

1,Displayname 2,Email 3.Phone 3.uid

c)What component(s) would you use to display the user's information?

- Core Components
- ActivityIndicator.
- Button.
- FlatList.
- API

17. Create a new component called "MyImagePicker" that allows the user to select an image from their device's photo gallery.

a) What external library or module would you use to implement this component?

LIBRARY : `react-native-image-picker`

b) What props would you pass to this component to customize its appearance?

1.selectimage

2.choosefromgallary

3.deny

c) How would you handle errors or exceptions when selecting an image?

Errors may occur with many problems but u should check

Image picker verion

React native version

19. Write a function called "generatePassword" that generates a random password with agiven length and complexity.

a) What parameters would you pass to this function?

To generate a password we need to create function with following parameters**,**

setPassword,setLength, setIsLowerCase,setIsUpperCase setIsNumbers ,setIsSymbols.

b) What algorithm or library would you use to generate the password?

First create a library that contains generate password package

import React, { useState } from "react";

import generator from "generate-password";

c)How would you ensure that the password meets the required complexity criteria?

Check following contions : password contains atleast an uppercace ,a lowercase,a symbol and a number andpassword length should be 8.

21. Explain the concept of "props drilling" in React Native.

Props drilling is data is often shared between components using props.

Prop drilling is basically a situation when the same data is being sent at almost every level due to requirements in the final level Anyone who has worked in React would have faced this and if not then will face it definitely.

The problem with Prop Drilling is that whenever data from the Parent component will be needed, it would have to come from each level, Regardless of the fact that it is not needed there and simply needed in last.

Parent Data ->child1->child2->child3

To overcome Props Drilling issue,

A better approach to doing this is using React context to handle the data.

22.What is the difference between a "controlled" and "uncontrolled" component in React Native?

Controlled Component	Uncontrolled Component
The component is under control of the component's state.	Components are under the control of DOM.
These components are predictable as are controlled by the state of the component.	Are Uncontrolled because during the life cycle

	methods the data may loss
Internal state is not maintained	Internal state is maintained
It accepts the current value as props	We access the values using refs
Does not maintain its internal state.	Maintains its internal state.
Controlled by the parent component.	Controlled by the DOM itself.
Have better control on the form data and values	Has very limited control over form values and data

23.What is the difference between "component state" and "application state" in React Native?

Componenent state :

Componenet State is mutable.

This means that state can be updated in the future while props can’t.

we can initialize state in the constructor, and then call setState when we want to change it.

Application state :

Application State can tell you if the app is in the foreground or background, and notify you when the state changes.

AppState is frequently used to determine the intent and proper behavior when handling push notifications.

24. What is Redux and how does it relate to React Native?

REDUX :

Redux is state management tool. With the help of Redux, all state in your application are kept in store and that can be accessible from all components.

USAGE OF REDUX:

As your application become big & more complex, it's hard to mange your states & pass data from one component to other components or child components.

This is where Redux helps - Giving global access to state or data which can be used in any component without depending on other components.

Functionalities of Redux :

1.Actions 2.Reducers 3. Store 4.Components 5.**Middleware**

25. How would you optimize the performance of a React Native app that has a large number of components?

We can use some process to optimize the performance of react native app ,

1. Reduce Unnecessary Re-Renders With useMemo Hook
2. Effective State Data Handling
3. Implement a Performance Monitoring System
4. Remove Console.log Statements
5. Build an Efficient Navigation System
6. Reduce App Size With Code Splitting and Lazy Loading

Sourcecode :

Button Creation :

```
import React from 'react';

import {
    StyleSheet, Button, View, SafeAreaView,
    Text, Alert
} from 'react-native';

export default function App() {
    return (
        <View style={styles.container}>

       <Button
  title="clickme"
  onPress={() => Alert.alert('BalajiSystem !')}
           />
  </View>
    );
     }
   const styles = StyleSheet.create({
    container: {
    flex: 1,
    backgroundColor: '#71EC4C',
    alignItems: 'center',
    justifyContent: 'center',
    },
    });
```

LISTCREATION:

```
import React, { Component } from 'react'

import { Text, View, TouchableOpacity, StyleSheet } from 'react-native'

class List extends Component {

state = {

names: [

{

id: 0,

 name: 'Balaji',

},

 {
```

```
id: 1,
 name: 'Ramasamy',
 },
{
 id: 2,
 name: 'dhoni',
 },
  {
 id: 3,
 name: 'kohli',
 }
 ]
}
 alertItemName = (item) => {
alert(item.name)
}
render() {
return (
 <View>
          {
 this.state.names.map((item, index) => (
                <TouchableOpacity
                  key = {item.id}
                  style = {styles.container}
```

```
                onPress = {() => this.alertItemName(item)}>
                <Text style = {styles.text}>
                  {item.name}
                </Text>
              </TouchableOpacity>
            ))
          }
        </View>
      )
    }}
export default List
const styles = StyleSheet.create ({
  container: {
    padding: 10,
    marginTop: 3,
    backgroundColor: '#d9f9b1',
    alignItems: 'center',
  },
  text: {
    color: '#4f603c'
  }
})
```

OPENWEATHERMAP API:

```
impo
rt
Reac
t, {
useS
tate }
from
"reac
t";
```

```
import styled from "styled-components";
import Axios from "axios";
import CityComponent from "./modules/CityComponent";
import WeatherComponent from
"./modules/WeatherInfoComponent";


export const WeatherIcons = {
  "01d": "/react-weather-app/icons/sunny.svg",
  "01n": "/react-weather-app/icons/night.svg",
  "02d": "/react-weather-app/icons/day.svg",
  "02n": "/react-weather-app/icons/cloudy-night.svg",
  "03d": "/react-weather-app/icons/cloudy.svg",
  "03n": "/react-weather-app/icons/cloudy.svg",
  "04d": "/react-weather-app/icons/perfect-day.svg",
  "04n": "/react-weather-app/icons/cloudy-night.svg",
  "09d": "/react-weather-app/icons/rain.svg",
  "09n": "/react-weather-app/icons/rain-night.svg",
  "10d": "/react-weather-app/icons/rain.svg",
  "10n": "/react-weather-app/icons/rain-night.svg",
  "11d": "/react-weather-app/icons/storm.svg",
  "11n": "/react-weather-app/icons/storm.svg",
};


const Container = styled.div`
  display: flex;
  flex-direction: column;
  align-items: center;
  width: 380px;
```

```
  padding: 20px 10px;
  margin: auto;
  border-radius: 4px;
  box-shadow: 0 3px 6px 0 #555;
  background: white;
  font-family: Montserrat;
`;


const AppLabel = styled.span`
  color: black;
  margin: 20px auto;
  font-size: 18px;
  font-weight: bold;
`;
const CloseButton = styled.span`
  padding: 2px 3px;
  background-color: black;
  border-radius: 50%;
  color: white;
  position: absolute;
`;


function App() {
  const [city, updateCity] = useState();
  const [weather, updateWeather] = useState();
  const fetchWeather = async (e) => {
    e.preventDefault();
    const response = await Axios.get(

`https://api.openweathermap.org/data/2.5/weather?q=${city}&app
id=fe4feefa8543e06d4f3c66d92c61b69c`,
    );
    updateWeather(response.data);
  };
  return (
    <Container>
      <AppLabel>React Weather App</AppLabel>
      {city && weather ? (
```

```
            <WeatherComponent weather={weather} city={city} />
          ) : (
            <CityComponent updateCity={updateCity}
        fetchWeather={fetchWeather} />
          )}
        </Container>
      );
    }


    export default App;
```

Imagepicker from device :

```
var options = {
title: 'Select Image',
   customButtons: [
   {
   name: 'customOptionKey',
   title: 'Choose Photo from Custom Option'
   },
   ],
   storageOptions: {
   skipBackup: true,
   path: 'images',
   },
   };
   ImagePicker.showImagePicker(options, response => {
   console.log('Response = ', response);
   if (response.didCancel) {
   console.log('User cancelled image picker');
   } else if (response.error) {
   console.log('ImagePicker Error: ', response.error);
   } else if (response.customButton) {
   console.log('User tapped custom button: ',
       response.customButton
     );
     alert(response.customButton);
   } else {
     setFilePath(response);
   }});
```

Display user information :

```
import React, { Component } from 'react';
import { View, ScrollView, ActivityIndicator } from 'react-native';
import { user_read, user_update } from './store/actions';
import { NavigationEvents } from 'react-navigation';
import { styles, Color } from './styles';
import { connect } from 'react-redux';
import EditUser from './edit';
import ViewUser from './view';

export class UserDetail extends Component {
  constructor(props) {
    super(props);
    this.state = {
      loading: false,
    };
  }

  load() {
    const { navigation } = this.props;
    const { token, auth_user } = this.props;

    if (!token && auth_user !== {}) {
      this.props.navigation.navigate('BasicLoginSignup');
```

```
    return;
   }

   const id = navigation.getParam('id', null) || auth_user.id;
   this.props.getUser(id, token);
  }

  componentDidUpdate(prevProps) {
   const { loading } = this.state;
   const { api } = this.props;

   if (loading && prevProps.api.isLoading && !api.isLoading) {
    this.setState({ loading: false });
   }
  }

  render() {
   const { loading } = this.state;
   const { isEdit } = this.props;
   return (
    <ScrollView                                    style={styles.container}
contentStyle={styles.content}>
     <NavigationEvents
      onDidFocus={() => this.load()}
```

```
        onWillFocus={() => this.setState({ loading: true })}
        onDidBlur={() => {
          this.props.navigation.setParams({ id: null });
        }}
      />
      {loading ? (
        <View>
          <ActivityIndicator color={Color.steel} />
        </View>
      ) : (
        <View>
          {isEdit ? (
            <EditUser {...this.props} />
          ) : (
            <ViewUser {...this.props} />
          )}
        </View>
      )}
    </ScrollView>
  );
 }
}

const mapStateToProps = (state, ownProps) => {
```

```
  const id =
    ownProps.navigation.getParam('id',                null)                ||
state.authReducer.user.id;

  return {
    token: state.authReducer.token,
    auth_user: state.authReducer.user,
    api: state.userReducer.api,
    user: state.userReducer.users.find(user => user.id == id) || {},
    isEdit: id === state.authReducer.user.id
  };
};

const mapDispatchToProps = dispatch => {
 return {
    getUser: (id, token) => dispatch(user_read(id, token)),
    updateUser: (data, token) => dispatch(user_update(data, token)),
  };
};
export default connect(
 mapStateToProps,
  mapDispatchToProps,
)(UserDetail);
```

Password Generator :

```
function App() {
 const [password, setPassword] = useState('');
 const [length, setLength] = useState(10);
 const [isLowerCase, setIsLowerCase] = useState(true);
 const [isUpperCase, setIsUpperCase] = useState(false);
 const [isNumbers, setIsNumbers] = useState(false);
 const [isSymbols, setIsSymbols] = useState(false);

 const generatePassword = () => {
  const pwd = generator.generate({
   length: length,
   lowercase: isLowerCase,
   uppercase: isUpperCase,
   numbers: isNumbers,
   symbols: isSymbols
  });
  setPassword(pwd);
 }

 return (
  <div>
```

```
<h5>Generate a random password in React - <a href="https://www.cluemediator.com/" target="_blank" rel="noopener noreferrer">Clue Mediator</a></h5>

<div class="container">
 <div class="row">
  <div class="col">
   <label>
    <span classname="lbl-len">Length:</span>
    <input type="number" classname="input-len form-control" value="{length}" onchange="{e" ==""> setLength(e.target.value)}
    />
   </label>
  </div>
 </div>
 <div class="row">
  <div class="col">
   <label classname="form-control">
    <input type="checkbox" classname="mr-1" checked="{isLowerCase}" onchange={() ==> setIsLowerCase(val => !val)}
    />
    <span>LowerCase</span>
   </label>
  </div>
  <div class="col">
```

```
          <label classname="form-control">
            <input type="checkbox" classname="mr-1" checked="{isUpperCase}" onchange={() ==> setIsUpperCase(val => !val)}
            />
            <span>UpperCase</span>
          </label>
        </div>
      </div>
      <div class="row">
        <div class="col">
          <label classname="form-control">
            <input type="checkbox" classname="mr-1" checked="{isNumbers}" onchange={() ==> setIsNumbers(val => !val)}
            />
            <span>Numbers</span>
          </label>
        </div>
        <div class="col">
          <label classname="form-control">
            <input type="checkbox" classname="mr-1" checked="{isSymbols}" onchange={() ==> setIsSymbols(val => !val)}
            />
            <span>Symbols</span>
          </label>
```

```
        </div>
      </div>
      <small>Note: At least one should be true.</small>
      <div class="row">
        <div class="col">
          <input type="button" classname="btn btn-dark mt-2 mb-3" value="Generate Password" onclick="{generatePassword}">
          <div>
            Password: {password}
          </div>
        </div>
      </div>
    </div>
  </div>
 );
}
export default App;
```

THANK YOU