1. Ans a
2. Ans a
3. Ans a
4. Ans b
5. Ans a
6. Ans a
7. Ans d
8. Ans b
9. Ans d
10. Ans a
11. A- can create by using init command

11.B -No define pattern can create according to our project requirenment

11. C- user appearance prefer.

12. – A- Ans a - customize the <TouchableOpacity /> component and include the <Text /> component inside of it to display the button text.

12 – B-b-click event handler

Enables

13.A- Ans a - A component with a render prop takes a function that returns a React element and calls it instead of implementing its own render logic

13.B-Ans b - const list = [{

name: 'Robin', },

name: 'Dennis', },

const App = () => { return (

<ul> {list. map((item) => (

<li key={item. id}>{item. name}</li>))}

</ul>);

13.C-Ans c- we will use the map() function for traversing the list element, and for updates, we enclosed them between curly braces {}.Now, can include this new list inside <ul> </ul> elements and render it to the DOM

Ans 14 b - The API key is all you need to call any of our weather APIs. Once we sign up using our email, the API key (APPID) will be sent to in a confirmation email. our API keys can always be found on our account page, where i can also generate additional API keys if needed.

Ans 15 a- Moving from one screen to another is performed by using the navigation prop, which passes down our screen components. It is similar to write the below code for a web browser: <a href="profiles. html">Go to Profile</a>

Ans 15 b- navigate and push accept an optional second argument to let we pass parameters to the route we are navigating to.

we can read the params through route.params inside a screen.

we can update the screen's params with navigation.setParams.

Initial params can be passed via the initialParams prop on Screen.

Ans 15 c - can easily pass rich data through your app and keep state out of the DOM.

Ans 16 a - React js. getItem() is the primary method for fetching or retrieving data from the storage object.

14. Ans a - Fahrenheit, Celsius and Kelvin units.

Ans 16 b - Errors caused by a problem within our system. The users can't do much about this.

Ans 17 a -choose any name for variables and constants.

Ans 17 b- add them to the JSX

Ans 19 a- generate_random_password(int $len = 8)

Ans 19. B- PKCE uses the SHA 256 Cryptographic Hash Algorithm.

Ans 20 a - initialRegion prop

Ans 17 c - Create a basic React Native app.Set up React Native Image Picker.Use React Native Image Picker to pick images in app.

Ans 18 a- land navigation, marine navigation, aeronautic navigation, and space navigation.

Ans 19 c- No common names or dictionary words.

No sequences of more than 4 digits in a row.

Include at least one character from at least 3 of these categories: Uppercase letter. Lowercase letter. ...

Password reset/expiration period as follows: 10-20 characters = no periodic reset/expiration required.

Ans 20 b - Errors caused by invalid user input or action. Users must be informed of this type of error and how to adjust and continue their interaction. It's good to show these errors in context. ...

Errors caused by a problem within our system. The users can't do much about this.

Ans 20 c - Four sets of binaries compiled for different CPU architectures.

A directory with resources, such as images, fonts, etc.

A JavaScript bundle with business logic and your React components.

Other files.

Ans 21 - Props drilling (threading) refers to the process of passing data from the parent component to the exact child component. But, in between, other components owning the props just to pass it down the chain.

Ans 22 - In a controlled component, form data is handled by a React component. The alternative is uncontrolled components, where form data is handled by the DOM itself.

Ans 23 - application state is global, and our component state is local. Flux or a flux-like library like Redux, can use what they call "stores" to hold application state. That means any component, anywhere in the app can access it (kind of like a database) so long as i hook into it.

Ans 24 - Redux is a predictable state container designed to help you write JavaScript apps that behave consistently across client, server, and native environments, and are easy to test.

Ans 25 - Use FlatList or SectionList to render large lists in React Native.