Secure Storage for Mobile Applications using React Native.

Date: 2023-11-13
By: csr13

Download as PDF

When storing sensitive information with mobile applications, there are many ways to do it, but one that I use a lot when working on mobile applications uses `AsyncStorage` from react-native-async-storage and `SecureStore` from expo-secure-store.

Here is the component I use for handling for example, authentication tokens to allow the user to fetch data from authorized endpoints

```
import AsyncStorage from '@react-native-async-storage/async-storage';
import * as SecureStore from 'expo-secure-store';

class StorageService {
  storeData = async (keyName, value) => {
    try {
      await SecureStore.setItemAsync(`${keyName}`, value);
    } catch (e) {
      return null;
    }
  };

  storeObjectData = async (keyName, value) => {
    try {
      const jsonValue = JSON.stringify(value);
      await SecureStore.setItemAsync(`${keyName}`, jsonValue);
    } catch (e) {
      return null;
    }
  };

  getData = async (keyName) => {
    try {
      const value = await SecureStore.getItemAsync(`${keyName}`);
      if (value !== null) {
        return value;
      }
      return null;
    } catch (e) {
      return null;
    }
  };

  getObjectData = async (keyName) => {
    var value;
    try {
      const jsonValue = await SecureStore.getItemAsync(`${keyName}`).succ;
      return jsonValue != null ? JSON.parse(jsonValue) : null;
    } catch (e) {
      value = null;
    }
    return value;
  };

  clearData = async () => {
    await SecureStore.deleteItemAsync('access_token');
    await SecureStore.deleteItemAsync('user');
  };
}

export default StorageService;
```

It is very useful for storing things for the current user in session, use cases depends on how the current application works, but the most important use case is to be able to store auth token, and load it to headers to make axios requests to the backend server.

Related Notes