Chat Screen WebSocket implementation

import React, {useEffect, useState} from 'react';

import {

View,

Text,

SafeAreaView,

TextInput,

ScrollView,

TouchableOpacity,

Modal,

StyleSheet,

FlatList,

ToastAndroid,

} from 'react-native';

import {Avatar, Icon} from 'react-native-elements';

import CommonTab from '../reusable\_components/CommonTab';

import ChatItem from '../reusable\_components/ChatItem';

import Entypo from 'react-native-vector-icons/dist/Entypo';

import Colors from '../../constants/Colors';

import { showToastMessage } from '../../constants';

import AsyncStorage from '@react-native-community/async-storage';

import { useIsFocused } from '@react-navigation/native';

const ChatScreen = ({navigation, route}) => {

const {data, pkId} = route.params;

console.log(data,"jhhj787jhhgj");

const [visible, setvisible] = useState(false);

const [modalVisible, setModalVisible] = useState(false);

const [Messages, setMessages] = useState()

console.log(Messages,"fkjdkjfjkfdf78d78df");

const focused = useIsFocused()

useEffect(()=>{

setMessages()

if(Platform.OS==='android'){

// AndroidKeyboardAdjust.setAdjustResize();

}

const ws=new WebSocket(wss://lenen.keycorp.in/ws/connection/gc/${pkId}/${data.id},null,{

headers: {

'Accept-Language': 'en,en-US;q=0.9,ru;q=0.8,de;q=0.7',

'Cache-Control': 'no-cache',

Pragma: 'no-cache',

'User-Agent':

'Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_13\_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/67.0.3396.87 Safari/537.36',

},

})

ws.onopen=()=>{

//setLoading(true)

}

ws.onmessage = ( data ) => {

console.log("message-->",data.data)

if(JSON.stringify(data.data)==='"[]"'){

// setLoading(false)

//setIsLoading(false)

}else{

// setLoading(false)

//setIsLoading(false)

let msg=JSON.parse((data.data))

// msg.sort((a, b) => parseFloat(a.id) - parseFloat(b.id));

msg.reverse()

let msgArr=[];

msg.map((item)=>{

let rg=/([\{]{3})([a-zA-Z0-9]+)([\}]{3})/gi

var text=item?.message

if(rg.test(item?.message)){

text=item?.message.replace(rg,function (x) {

return String.fromCodePoint(x.replaceAll(/{|}/gi, ""));

})

}

let msgObj={};

if(item?.created\_by?.profile?.picture){

msgObj= {

\_id: item.id,

text: text ,

createdAt: item?.created\_on,

user: {

\_id: item?.created\_by?.id,

name: item?.created\_by?.first\_name+" "+item?.created\_by?.last\_name,

//avatar: require('../../assets/ProfileAvatar.png'),

// avatar: BASEURL+item.created\_by.profile?.picture

},

}

}else{

msgObj= {

\_id: item.id,

text: text,

createdAt: item?.created\_on,

user: {

\_id: item?.created\_by?.id,

name:item?.created\_by?.first\_name+" "+item?.created\_by?.last\_name,

// avatar: require('../../Assets/default\_user.png'),

//avatar: "https://vwam.keycorp.in//"+item.created\_by.profile\_pic

},

}

}

msgArr.unshift(msgObj)

})

setMessages(msgArr)

}

};

return () => {

ws.close();

if(Platform.OS==='android'){

// AndroidKeyboardAdjust.setAdjustPan();

}

}

},[focused])

const [text, setText] = useState()

const sentMsg =async ()=>{

if(text){

setText()

var raw = JSON.stringify({

"message": text,

"group": data.id,

"created\_by": pkId

});

const Accesstoken = await AsyncStorage.getItem('acsess\_token');

var requestOptions = {

method: 'POST',

headers: {

Authorization: Token ${Accesstoken},

'Content-Type': 'application/json',

},

body: raw,

};

await fetch("https://lenen.keycorp.in/api/v1/chat/message/send", requestOptions)

.then(response => response.json())

.then(result => {console.log(result), ToastAndroid.show("Sent",2000)})

.catch(error => console.log('error', error));

}

else{

ToastAndroid.show("Text field is required",2000)

}

}

const handleExit = async id => {

const Accesstoken = await AsyncStorage.getItem('acsess\_token');

var requestOptions = {

method: 'PUT',

headers: {

Authorization: Token ${Accesstoken},

'Content-Type': 'application/json',

},

body: JSON.stringify({group: id}),

};

fetch(

'https://lenen.keycorp.in/api/v1/account/group\_leave/',

requestOptions,

)

.then(response => response.json())

.then(result => {showToastMessage("You left the group"),setModalVisible(!modalVisible),navigation.navigate("GroupList")})

.catch(error => console.log('error', error));

};

return (

<SafeAreaView style={{flex: 1, backgroundColor: '#fff'}}>

<CommonTab

title={data.group\_name}

icon={<Avatar source={{uri: https://lenen.keycorp.in${data.group\_icon}}} size={'small'} rounded/>}

left={

<TouchableOpacity

style={{paddingRight: 10}}

onPress={() => navigation.goBack()}>

<Icon

name="arrowleft"

type="antdesign"

size={16}

color={'#6f6f6f'}

/>

</TouchableOpacity>

}

right={

<TouchableOpacity

onPress={() => {

setvisible(!visible);

}}>

<Entypo name="dots-three-vertical" size={16} color={'#6f6f6f'} />

</TouchableOpacity>

}

/>

{visible ? (

<View

style={{

position: 'absolute',

right: 40,

top:10,

backgroundColor: '#fff',

borderWidth:0.5,

elevation:6,

zIndex: 1,

paddingRight: 40,

paddingLeft: 20,

paddingVertical: 10,

}}>

<TouchableOpacity

onPress={() => {

setvisible(false);

navigation.navigate("Groupinfo",{datas: data})

}}>

<Text

style={{

color: '#000',

fontSize: 16,

marginBottom: 10,

fontWeight: '600',

}}>

Group info

</Text>

</TouchableOpacity>

<TouchableOpacity

onPress={() => {

setvisible(false);

setModalVisible(true);

}}>

<Text style={{color: '#000', fontSize: 16, fontWeight: '600'}}>

Leave

</Text>

</TouchableOpacity>

</View>

) : null}

<ScrollView style={{flex: 1, backgroundColor: '#fff'}}>

<View style={styles.centeredView}>

<Modal

animationType="fade"

transparent={true}

visible={modalVisible}

onRequestClose={() => {

Alert.alert('Modal has been closed.');

setModalVisible(!modalVisible);

}}>

<View style={styles.centeredView}>

<View style={styles.modalView}>

<Text style={styles.modalText}>

Exit "{data.group\_name}" group?

</Text>

<View

style={{

flexDirection: 'row',

justifyContent: 'space-between',

marginHorizontal:20,

marginVertical:10

}}>

<TouchableOpacity

onPress={() => setModalVisible(!modalVisible)}>

<Text style={styles.textStyle}>Cancel</Text>

</TouchableOpacity>

<TouchableOpacity

onPress={() => handleExit(data.id)}>

<Text style={styles.textStyle}>Exit</Text>

</TouchableOpacity>

</View>

</View>

</View>

</Modal>

</View>

<FlatList

data={Messages}

ListEmptyComponent={<Text style={{alignSelf:'center'}}>No messages here yet...</Text>}

inverted

contentContainerStyle={{ flexDirection:'column-reverse' }}

style={{flex: 1, backgroundColor: '#fff',marginBottom:65}}

renderItem={({item})=>(

<>

{item?.user?.\_id===pkId ?<ChatItem datas={item} sent={true} />:<ChatItem datas={item}/>

}

</>

)}

/>

{/\* <Text

style={{

color: 'black',

fontFamily: 'Poppins-Bold',

marginVertical: 6,

padding: 2,

alignSelf: 'center',

fontSize: 12,

}}>

Tuesday 6, 2021

</Text> \*/}

{/\* <ChatItem />

<ChatItem sent={true} />

<ChatItem />

<ChatItem sent={true} /> \*/}

</ScrollView>

<View

style={{

flexDirection: 'row',

flex: 1,

position: 'absolute',

bottom: 16,

marginHorizontal: 16,

}}>

<TextInput

placeholder="Message"

value={text}

onChangeText={msg=>setText(msg)}

placeholderTextColor={'black'}

style={{

backgroundColor: 'rgba(246,246,246,255)',

borderRadius: 8,

flex: 1,

height: 42,

padding: 5,

}}

/>

<TouchableOpacity

onPress={()=>sentMsg()}

style={{

backgroundColor: '#dc4133',

borderRadius: 20,

width: 40,

height: 42,

justifyContent: 'center',

alignItems: 'center',

marginLeft: 8,

}}>

<Icon name="send" type="feather" size={20} color="white" />

</TouchableOpacity>

</View>

</SafeAreaView>

);

};

export default ChatScreen;

const styles = StyleSheet.create({

centeredView: {

flex: 1,

justifyContent: 'center',

alignItems: 'center',

},

modalView: {

margin: 20,

backgroundColor: '#fff',

borderRadius: 20,

paddingHorizontal: 35,

paddingVertical: 20,

shadowColor: '#000',

shadowOffset: {

width: 0,

height: 2,

},

shadowOpacity: 0.25,

shadowRadius: 4,

elevation: 5,

},

textStyle: {

color: Colors.primary,

fontWeight: 'bold',

textAlign: 'center',

},

modalText: {

marginBottom: 15,

textAlign: 'center',

color: '#000',

fontSize: 16,

fontWeight: '400',

},

})