import React, { useState, useRef } from "react";

import {

View,

Text,

TouchableOpacity,

StyleSheet,

ScrollView,

SafeAreaView,

StatusBar,

Alert,

Dimensions,

} from "react-native";

export default function App() {

const [selectedUser, setSelectedUser] = useState(null);

const [audioOutput, setAudioOutput] = useState("بلندگو");

const [cameraType, setCameraType] = useState("دوربین عقب");

const scrollViewRef = useRef(null);

const users = Array.from({ length: 27 }, (\_, i) => ({

id: i + 1,

name: `کاربر ${i + 1}`,

image: null,

}));

const adminUsername = "AliAdmin";

const natureEmojis = [

"🌳", "⭐", "⚡", "🌊", "🌸", "⛰️", "☀️", "☁️", "🍃", "🌙",

"🌴", "🌺", "🏞️", "🌞", "⛅", "🍂", "🌕", "🌈", "❄️", "🌬️",

"🍁", "🌲", "🌵", "🌾", "🍄", "🌋", "🌼", "🍀", "🌹", "🌑",

"🍎", "🍒", "🍇", "🍉", "🍊", "🍋", "🍍", "🥭", "🍓", "🥝",

"🥕", "🌽", "🍅", "🥔", "🥑", "🍆", "🥒", "🥬", "🥦", "🌰",

"🐾", "🦋", "🐞", "🐝", "🐦", "🦅", "🦇", "🐟", "🐬", "🐳",

"🦀", "🐙", "🦑", "🐊", "🦎", "🐍", "🦕", "🦖", "🐘", "🦒",

"🐅", "🐆", "🐴", "🦄", "🐪", "🐫", "🐐", "🐑", "🐖", "🐓",

"🦃", "🦚", "🦜", "🦢", "🦩", "🐕", "🐈", "🐇", "🐿️", "🦔",

"🦦", "🦝", "🦡", "🐀", "🐁", "🦥", "🦨", "🦘", "🦧", "🦍",

];

const defaultEmoji = "👤";

const availableAudioOutputs = ["بلندگو", "هندزفری", "بلوتوث"];

const availableCameras = ["دوربین عقب", "دوربین جلو", "دوربین USB"];

const getDefaultEmoji = (userId) => {

const emoji = userId <= 100 ? natureEmojis[(userId - 1) % natureEmojis.length] : defaultEmoji;

console.log(`User ${userId} using emoji: ${emoji}`);

return emoji;

};

const handleExit = () => {

Alert.alert("خروج از جلسه آنلاین انجام شد!");

};

const handleSettings = () => {

Alert.alert(

"تنظیمات",

"انتخاب کنید:",

[

{

text: "تنظیمات صدا",

onPress: () =>

Alert.alert(

"انتخاب خروجی صدا",

"یکی را انتخاب کنید:",

availableAudioOutputs.map((output) => ({

text: `${output}${audioOutput === output ? " (فعال)" : ""}`,

onPress: () => {

setAudioOutput(output);

Alert.alert(`خروجی صدا به ${output} تغییر کرد!`);

},

})).concat({ text: "بستن", style: "cancel" })

),

},

{

text: "تنظیمات تصویر",

onPress: () =>

Alert.alert(

"انتخاب دوربین",

"یکی را انتخاب کنید:",

availableCameras.map((camera) => ({

text: `${camera}${cameraType === camera ? " (فعال)" : ""}`,

onPress: () => {

setCameraType(camera);

Alert.alert(`دوربین به ${camera} تغییر کرد!`);

},

})).concat({ text: "بستن", style: "cancel" })

),

},

{ text: "بستن", style: "cancel" },

],

{ cancelable: true }

);

};

const handleChat = () => {

Alert.alert("چت باز شد!");

};

const handleBack = () => {

setSelectedUser(null);

if (scrollViewRef.current) {

scrollViewRef.current.scrollTo({ y: 0, animated: true });

}

};

const handleUserTap = (userId) => {

const previousUser = selectedUser;

setSelectedUser(selectedUser === userId ? null : userId);

if (scrollViewRef.current && userId && previousUser !== userId) {

setTimeout(() => {

scrollViewRef.current.scrollTo({ y: 0, animated: true });

}, 100);

}

};

const renderFullScreenUser = () => {

if (!selectedUser) return null;

const user = users.find((u) => u.id === selectedUser);

const { width, height } = Dimensions.get("window");

const emoji = getDefaultEmoji(user.id);

return (

<TouchableOpacity

onPress={() => handleUserTap(user.id)}

style={[styles.fullScreenUserBox, { width: width \* 0.9, height: height \* 0.6 }]}

>

<Text style={styles.fullScreenEmoji}>{emoji}</Text>

<Text style={styles.fullScreenUserName}>{user.name}</Text>

</TouchableOpacity>

);

};

const { width } = Dimensions.get("window");

const margin = 10;

const baseBoxWidth = 90;

const boxWidthWithMargin = baseBoxWidth + margin;

const numColumns = Math.floor(width / boxWidthWithMargin);

const renderMiniUser = (item) => {

const emoji = getDefaultEmoji(item.id);

return (

<TouchableOpacity

key={item.id}

onPress={() => handleUserTap(item.id)}

style={styles.userBox}

>

<Text style={styles.userEmoji}>{emoji}</Text>

<Text style={styles.userName}>{item.name}</Text>

</TouchableOpacity>

);

};

const renderUserGrid = () => {

const rows = [];

for (let i = 0; i < users.length; i += numColumns) {

const rowUsers = users.slice(i, i + numColumns);

rows.push(

<View key={i} style={styles.userRow}>

{rowUsers.map((user) => renderMiniUser(user))}

</View>

);

}

return rows;

};

return (

<SafeAreaView style={styles.container}>

<StatusBar barStyle="light-content" />

<View style={styles.mainContainer}>

<View style={styles.header}>

<TouchableOpacity style={styles.backButton} onPress={handleBack}>

<Text style={styles.backButtonText}>⬅️</Text>

</TouchableOpacity>

<View style={styles.headerTextContainer}>

<Text style={styles.headerTitle}>جلسه آنلاین</Text>

<View style={styles.userInfo}>

<Text style={styles.headerUsername}>{adminUsername}</Text>

<View style={styles.adminEmojiContainer}>

<Text style={styles.headerEmoji}>🌳</Text>

</View>

</View>

</View>

</View>

<ScrollView

ref={scrollViewRef}

style={styles.scrollView}

contentContainerStyle={styles.scrollContent}

showsVerticalScrollIndicator={true}

bounces={true}

alwaysBounceVertical={true}

scrollEnabled={true}

>

{renderFullScreenUser()}

{renderUserGrid()}

</ScrollView>

<View style={styles.footerButtons}>

<TouchableOpacity style={styles.settingsButton} onPress={handleSettings}>

<Text style={styles.circleButtonText}>⚙️</Text>

</TouchableOpacity>

<TouchableOpacity style={styles.chatButton} onPress={handleChat}>

<Text style={styles.circleButtonText}>💬</Text>

</TouchableOpacity>

<TouchableOpacity style={styles.exitButton} onPress={handleExit}>

<Text style={styles.circleButtonText}>✖️</Text>

</TouchableOpacity>

</View>

</View>

</SafeAreaView>

);

}

const styles = StyleSheet.create({

container: {

flex: 1,

backgroundColor: "#0A1D3A",

},

mainContainer: {

flex: 1,

position: "relative",

},

header: {

padding: 20,

paddingTop: StatusBar.currentHeight || 40,

backgroundColor: "#0A1D3A",

flexDirection: "row",

alignItems: "center",

justifyContent: "space-between",

},

backButton: {

padding: 5,

},

backButtonText: {

fontSize: 24,

color: "#fff",

},

headerTextContainer: {

alignItems: "flex-end",

},

headerTitle: {

color: "#fff",

fontSize: 18,

fontWeight: "bold",

},

userInfo: {

flexDirection: "row",

alignItems: "center",

marginTop: 5,

},

headerUsername: {

color: "#fff",

fontSize: 14,

marginRight: 10,

},

adminEmojiContainer: {

width: 40,

height: 40,

borderRadius: 20,

borderWidth: 2,

borderColor: "#FF00FF",

alignItems: "center",

justifyContent: "center",

},

headerEmoji: {

fontSize: 30,

},

scrollView: {

flex: 1,

},

scrollContent: {

padding: 10,

paddingBottom: 80,

},

userRow: {

flexDirection: "row",

justifyContent: "flex-end",

marginBottom: 10,

},

userBox: {

width: 90,

height: 120,

backgroundColor: "#07152B",

borderRadius: 10,

alignItems: "center",

justifyContent: "center",

margin: 5,

elevation: 2,

},

userEmoji: {

fontSize: 50,

},

userName: {

marginTop: 5,

fontSize: 14,

color: "#fff",

},

fullScreenUserBox: {

backgroundColor: "#07152B",

borderRadius: 10,

alignItems: "center",

justifyContent: "center",

margin: 10,

elevation: 2,

borderColor: "#00FF00",

borderWidth: 2,

},

fullScreenEmoji: {

fontSize: 100,

},

fullScreenUserName: {

marginTop: 10,

fontSize: 20,

fontWeight: "bold",

color: "#fff",

},

footerButtons: {

position: "absolute",

bottom: 15,

left: 0,

right: 0,

flexDirection: "row",

justifyContent: "space-around",

},

settingsButton: {

width: 70,

height: 70,

borderRadius: 35,

backgroundColor: "#FFFFFF",

alignItems: "center",

justifyContent: "center",

elevation: 5,

},

chatButton: {

width: 70,

height: 70,

borderRadius: 35,

backgroundColor: "#00FFFF",

alignItems: "center",

justifyContent: "center",

elevation: 5,

},

exitButton: {

width: 70,

height: 70,

borderRadius: 35,

backgroundColor: "#FF0000",

alignItems: "center",

justifyContent: "center",

elevation: 5,

},

circleButtonText: {

fontSize: 24,

color: "#fff",

},

});