```
src/
├── screens/
│   ├── HomeScreen.tsx
│   ├── TaskScreen.tsx
│   └── WalletScreen.tsx
├── services/
│   ├── auth.ts
│   ├── firestore.ts
│   └── ads.ts
├── store/
│   ├── slices/
│   │   └── userSlice.ts
│   └── store.ts
├── components/
│   ├── VideoTask.tsx
│   └── QuizTask.tsx
```

App.tsx

```
Import React from 'react';
Import { NavigationContainer } from '@react-navigation/native';
Import { createStackNavigator } from '@react-navigation/stack';
Import HomeScreen from './src/screens/HomeScreen';
Import TaskScreen from './src/screens/TaskScreen';
Import WalletScreen from './src/screens/WalletScreen';
Import { Provider } from 'react-redux';
Import store from ‘./src/store/store’;


Const Stack = createStackNavigator();


Export default function App() {
  Return (
    <Provider store={store}>
      <NavigationContainer>
        <Stack.Navigator>
          <Stack.Screen name=”Home” component={HomeScreen} />
          <Stack.Screen name=”Tasks” component={TaskScreen} />
          <Stack.Screen name=”Wallet” component={WalletScreen} />
        </Stack.Navigator>
      </NavigationContainer>
    </Provider>
  );
}import { configureStore } from ‘@reduxjs/toolkit’;
Import userReducer from ‘./slices/userSlice’;


Export const store = configureStore({
  Reducer: {
    User: userReducer,
  },
});
```

```
Export type RootState = ReturnType<typeof store.getState>;
Export type AppDispatch = typeof store.dispatch;
Import { createSlice, PayloadAction } from '@reduxjs/toolkit';

Interface UserState {
  Uid: string | null;
  Points: number;
}

Const initialState: UserState = {
  Uid: null,
  Points: 0,
};

Const userSlice = createSlice({
  Name: 'user',
  initialState,
  reducers: {
    setUser: (state, action: PayloadAction<string>) => {
      state.uid = action.payload;
    },
    addPoints: (state, action: PayloadAction<number>) => {
      state.points += action.payload;
    },
  },
});

Export const { setUser, addPoints } = userSlice.actions;
Export default userSlice.reducer;
Import auth from '@react-native-firebase/auth';
Import { GoogleSignin } from '@react-native-google-signin/google-signin';
Import { AppDispatch } from '../store/store';
Import { setUser } from '../store/slices/userSlice';

GoogleSignin.configure({
  webClientId: 'YOUR_GOOGLE_WEB_CLIENT_ID',
});

Export const signInWithGoogle = async (dispatch: AppDispatch) => {
  Try {
    Const { idToken } = await GoogleSignin.signIn();
    Const credential = auth.GoogleAuthProvider.credential(idToken);
    Const userCredential = await auth().signInWithCredential(credential);
    Dispatch(setUser(userCredential.user?.uid || ''));
  } catch (error) {
    Console.error("Google Sign-In Error:", error);
  }
```

```
};
Import React from ‘react’;
Import { View, Button, Alert } from ‘react-native’;
Import { useDispatch, useSelector } from ‘react-redux’;
Import { RootState } from ‘../store/store’;
Import { addPoints } from ‘../store/slices/userSlice’;
Import VideoTask from ‘../components/VideoTask’;
Import QuizTask from ‘../components/QuizTask’;

Const TaskScreen = () => {
  Const dispatch = useDispatch();
  Const points = useSelector((state: RootState) => state.user.points);

  Const handleQuizComplete = (reward: number) => {
    Dispatch(addPoints(reward));
    Alert.alert(‘Success!’, `You earned ${reward} points!`);
  };

  Return (
    <View style={{ padding: 20 }}>
      <VideoTask onComplete={() => dispatch(addPoints(10))} />
      <QuizTask onComplete={() => handleQuizComplete(20)} />
      <Button
        Title=”Redeem Points”
        onPress={() => Alert.alert('Redeem', `Your balance: ${points} points`)}
      />
    </View>
  );
};

Export default TaskScreen;
Import React, { useEffect } from 'react';
Import { RewardedAd, RewardedAdEventType } from 'react-native-google-mobile-ads';

Const adUnitId = __DEV__ ? RewardedAd.TestIds : 'ca-app-pub-xxxxxxxxxxxxx/yyyyyyyyyy';

Const VideoTask = ({ onComplete }: { onComplete: () => void }) => {
  useEffect(() => {
    const rewardedAd = RewardedAd.createForAdRequest(adUnitId);
    const unsubscribe = rewardedAd.addAdEventListener(RewardedAdEventType.EARNED_REWARD, () => {
      onComplete();
    });
    rewardedAd.load();
    return unsubscribe;
  }, []);
  Return null; // ভিডিও প্লেয়ার UI যোগ করুন
};

Export default VideoTask;
Import React from ‘react’;
Import { useDispatch, useSelector } from ‘react-redux’;
Import { RootState } from ‘../store/store’;
Import { CardField, useStripe } from ‘@stripe/stripe-react-native’;

Const WalletScreen = () => {
  Const { confirmPayment } = useStripe();
  Const points = useSelector((state: RootState) => state.user.points);

  Const handlePayment = async () => {
    Const response = await fetch(‘https://your-api/create-payment-intent’, {
      Method: ‘POST’,
      Headers: { ‘Content-Type’: ‘application/json’ },
      Body: JSON.stringify({ amount: points * 0.01 }), // 1 point = $0.01
    });
    Const { clientSecret } = await response.json();
    Const { error } = await confirmPayment(clientSecret);
    If (error) Alert.alert(‘Payment Failed’);
    Else Alert.alert(‘Success!’, ‘Payment completed!’);
  };
  Return (
    <View>
      <CardField postalCodeEnabled={false} />
      <Button title={`Redeem $${(points * 0.01).toFixed(2)}`} onPress={handlePayment} />
    </View>
  );
};

Export default WalletScreen;
```

```
Npm install @react-native-firebase/app @react-native-firebase/auth @react-native-firebase/firestore react-native-google-mobile-ads @stripe/stripe-react-native @react-navigation/native @reduxjs/toolkit react-redux
Npx react-native run-android –variant=debug
# অথবা
Npx react-native run-ios –scheme=Debug
```