Integration Document- Tele-Health Services (Android) – PI-1802.0

**Document History**

**Version**

**Date**

**Author**

**Section**

**Changes**

1.0

5/15/2017

Spoorti Hallur

All

Release for 1802.0

12/06/2018

Anurag Gautam

All

Release for 1802.0

Administrative Information

Project Name

CoCo Telehealth-services

Project Identification

NA

Scrum Master

Mahabaleshwara Adiga

Date

12/06/2018

Quality Leader

Form Filled by

Spoorti Hallur

Name

Address

Phone

Fax

All Projects in CDP2

**Authorization**

Quality Leader Signature

Scrum Master Signature

Mahabaleshwara Adiga

**For Code:**

**For Code:**

**Product : Telehealth Services**

**Platform Release Version :** **1802.0**

**Amwell SDK: : 3.2.2**

**Release Date : 12-June-2018**

User manual (Optional): NA Customer Version ID (Optional): NA

Contents

**Authorization** 1

**Introduction** 3

**Prerequisites** 3

**Library Integration** 4

**Maven repository Integration** 4

**Library Integration** 4

Telehealth Dependency to be added by Propositions 6

**User Registration** 6

**Telehealth standard micro-app interfaces to be implemented by Verticals are as follows** 6

Reference **APP** link 6

**Notes** 6

**Introduction**

Telehealth services is built as a common-component (micro-app). The purpose of this micro-app is to facilitate end-to-end telehealth consultation the user. Telehealth brings in a unique face to face experience for the user for consulting the doctor remotely.

The micro-app is build using a 3rd party Library (Amwell). Amwell SDK provides the necessary tools to build the micro-app and gives the control over the app for look and feel of app and the way the app operates.

This document provides an overview of integrating Telehealth Services Microapp in Android mobile application.

**Source Path:**

[THS code repository](http://tfsemea1.ta.philips.com:8080/tfs/TPC_Region24/CDP2/_git/plf-android?path=%2FSource%2Fths)

**Prerequisites**

• Vertical project is configured for Android Studio

• Setting->Developer Options->Don’t Keep Activities should be unchecked. [If Developer Mode is on]

• Android API version should on 21

**Library Integration**

**Maven repository Integration**

The easiest and preferred way to use these components is using maven. Only we need to add maven repositories in app build gradle:

buildscript {

apply **from**: projectDir.absolutePath + **'/../../../../../build-support/gradle/versions.gradle'**

repositories {

maven {

url repos.jcenter

credentials {

username = **"readonly"**

password = **"123qweasdzxc"**

}

}

maven {

url repos.plugin

credentials {

username = **"readonly"**

password = **"123qweasdzxc"**

}

}

}

dependencies {

classpath builddeps.gradle

classpath builddeps.buildinfoextractor

classpath(builddeps.cdpSCM)

classpath builddeps.androidapt

classpath builddeps.bintrayplugin

classpath builddeps.mobilej

}

}

Here dependency fields are defined in common file plf-android/build-support/gradle/version.gradle

**Library Integration**

• Checkout the code from above path wherein developer can find sample app which depends on Telehealth Services library’s aar file :

• Telehealth Services needs other libraries to build which are as below

* User registration
* App-Infra

• Please make sure all these are added to application with the latest versions along with Telehealth library aar file under libs folder.

• Clean project

• Refer Telehealth Services dependencies along with .aar files in build.gradle as mentioned below

Telehealth Dependency to be added by Propositions

   philipsdeps.**telehealth**

**User Registration**

User Registration HSDP flow is used to get HSDP token. The token along with the uuid of the user are used for authentication with Amwell server

**Telehealth standard micro-app interfaces to be implemented by Verticals are as follows**

• The below files are supposed to be added in assets folder

• ADBMobileConfig.json

• AppConfig.json

• logging.properties

• Propositions must initialize App-Infra in the Application class and inject to telehealth services as a dependency through microapp interface

• Propositions must initialize Registration and launch Amwell component post the User Login. This is mandatory else the application crashes

• Propositions must implement ActionBarListener for handling the back navigation

• Propositions must handle Back navigation only when micro-app passes “true” through handleBackEvent

• Use the below code for launching the TeleHealth Componant

PTHMicroAppLaunchInput = new THSMicroAppLaunchInput("Launch Uapp Input" , THSCompletionProtocol);

PTHMicroAppInterface = new THSMicroAppInterface();  
PTHMicroAppInterface.init(new THSMicroAppDependencies(((THSDemoApplication) this.getApplicationContext()).getAppInfra()), new THSMicroAppSettings(this.getApplicationContext()));  
PTHMicroAppInterface.launch(fragmentLauncher, PTHMicroAppLaunchInput);

**• For more details please refer the Demo-App of Telehealth services or Reference app**.

Reference **APP**

[Reference app code repository](http://tfsemea1.ta.philips.com:8080/tfs/TPC_Region24/CDP2/_git/plf-android?path=%2FSource%2Frap)

**Notes**

• Registration is developed as separate library project. App is expected to invoke registration library before calling Telehealth component.

• Library should be initialized as per document or sample app otherwise library would throw runtime exception.

• Please refer sample application for more details

• All dependencies can be referred as it is.

• Configuration can be followed [all sample configurations are provided in sample]

**Deep Linking:**

Proposition side change:

Proposition can launch THS in two ways:

1. Direct Launch (eg from option, Hamburger Menu)

// Method signature to launch THS from proposition without deep linking

**public** THSMicroAppLaunchInput(String welcomeMessage, THSCompletionProtocol thsCompletionProtocol)

1. Deep Linking (THS launch from a html link )

// Method signature to launch THS from proposition as deep linking

THSMicroAppLaunchInput(String welcomeMessage, THSCompletionProtocol thsCompletionProtocol, **boolean** appointmentFlow);

THSMicroAppLaunchInput microAppLaunchInput = **new** THSMicroAppLaunchInput(**""**,**this**,**true**);

**• App side change:**

Below intent needs to be added under launching activity of Application(Proposition/RefApp) manifiest

<**intent-filter**>  
 <**action android:name="android.intent.action.VIEW"**/>  
 <**category android:name="android.intent.category.DEFAULT"**/>  
 <**category android:name="android.intent.category.BROWSABLE"**/>  
 <**data  
 android:host="telehealth.com"  
 android:scheme="philips"**/>  
</**intent-filter**>

**• Amwell Admin portal change:**

**• Login to amwell (Staging/iot/production etc)**

**e.g. https://stagingoc169.mytelehealth.com/startAdmin.htm**

**User: Sramamurthy**

**Password: Philips@123**

**• Go to “System Admin” -> “Manage Mobile App Settings”**

**Select(add/edit) the SDK App with same APP key as that of your app** apiKey **key in Appconfig.json**

Provide mandatory fields, Application id(Android) , Play Store URL, Add Launch Scheme as **“philips://telehealth.com”** and SDK API Key as your App apiKey in Appconfig.json

**"ths"**: {  
 **"apiKey"**: **"dc573250"**,  
 **"gdprEnabled"**:**false**}

Now Appointment mail can be tested for launch of app.

For reference: https://developer.android.com/training/app-links/deep-linking.html

**GDPR Toggle**

Proposition can launch THS with or without GDPR option (THS consent(s)).

Proposition needs to set the THS GDPR key (AppConfig.json) as true to have THS GDPR consent(s).

**"ths"**: {  
 **"apiKey"**: **"dc573250"**,  
 **"gdprEnabled"**:**true**}

If **gdprEnabled** is set to false , user will not see THS location consent screen within THS.