

Mobile Application Builder-Android Guide
Oracle Banking Digital Experience
Patchset Release 22.2.6.0.0

Part No. F72987-01

April 2025

Mobile Application Builder-Android Guide
April 2025
Oracle Financial Services Software Limited
Oracle Park
Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India
Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax: +91 22 6718 3001

www.oracle.com/financialservices/

Copyright © 2006, 2025, Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.



Table of Contents

1. Preface	1-5
1.1 Purpose	1-5
1.2 Audience	1-5
1.3 Documentation Accessibility	1-5
1.4 Critical Patches	1-5
1.5 Diversity and Inclusion	1-5
1.6 Conventions	1-5
1.7 Screenshot Disclaimer	1-6
1.8 Acronyms and Abbreviations	1-6
2. OBDX Servicing Application.....	2-1
2.1 Prerequisites	2-1
2.2 Create project using Remote UI	2-3
2.3 Local UI by running on local machine or local server.	2-3
2.4 Importing in Android Studio	2-5
2.5 Widget Functionality	2-5
2.6 Scan to Pay from Application Icon –.....	2-6
2.6 Passkey (Passwordless login)	2-6
2.7 Deeplinking - To open reset password, claim money links with the application	2-9
2.8 Device Registration and Push Registration Functionality.....	2-11
2.9 Location Tracking Metrics	2-12
2.10 Displaying Rate Option to Redirect to Playstore Page	2-12
2.11 Enabling Force Update	2-12
2.12 Splash Screen Migration	2-13
2.13 App Update Manager.....	2-13
2.14 Auto OTP Configuration	2-13
3. Google Play Integrity	3-1
4. FCM Push Notifications.....	4-1
5. Build Release Artifacts.....	5-1
6. OBDX Authenticator Application	6-1
6.1 Authenticator UI (Follow any one step below)	6-1
6.2 Authenticator Application Workspace Setup	6-3
7. Application Security Configuration	7-1

8. Adding Custom Cordova Plugin	8-2
9. ODA Chatbot Inclusion.....	9-1
10. Push Notification 2FA configuration.....	10-1

1. Preface

1.1 Purpose

Welcome to the User Guide for Oracle Banking Digital Experience. This guide explains the operations that the user will follow while using the application.

1.2 Audience

This manual is intended for Customers and Partners who setup and use Oracle Banking Digital Experience.

1.3 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit, <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

1.4 Critical Patches

Oracle advises customers to get all their security vulnerability information from the Oracle Critical Patch Update Advisory, which is available at [Critical Patches, Security Alerts and Bulletins](#). All critical patches should be applied in a timely manner to ensure effective security, as strongly recommended by [Oracle Software Security Assurance](#).

1.5 Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1.6 Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

<i>Italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1.7 **Screenshot Disclaimer**

The images of screens used in this user manual are for illustrative purpose only, to provide improved understanding of the functionality; actual screens that appear in the application may vary based on selected browser, theme, and mobile devices.

1.8 **Acronyms and Abbreviations**

The list of the acronyms and abbreviations that you are likely to find in the manual are as follows:

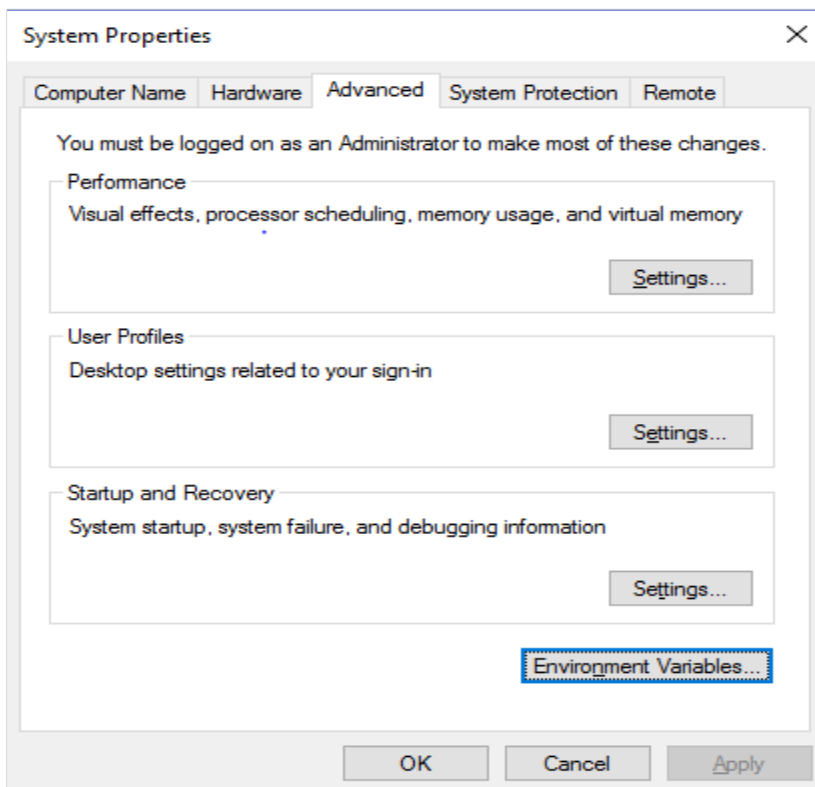
Abbreviation	Description
OBDX	Oracle Banking Digital Experience

2. OBDX Servicing Application

2.1 Prerequisites

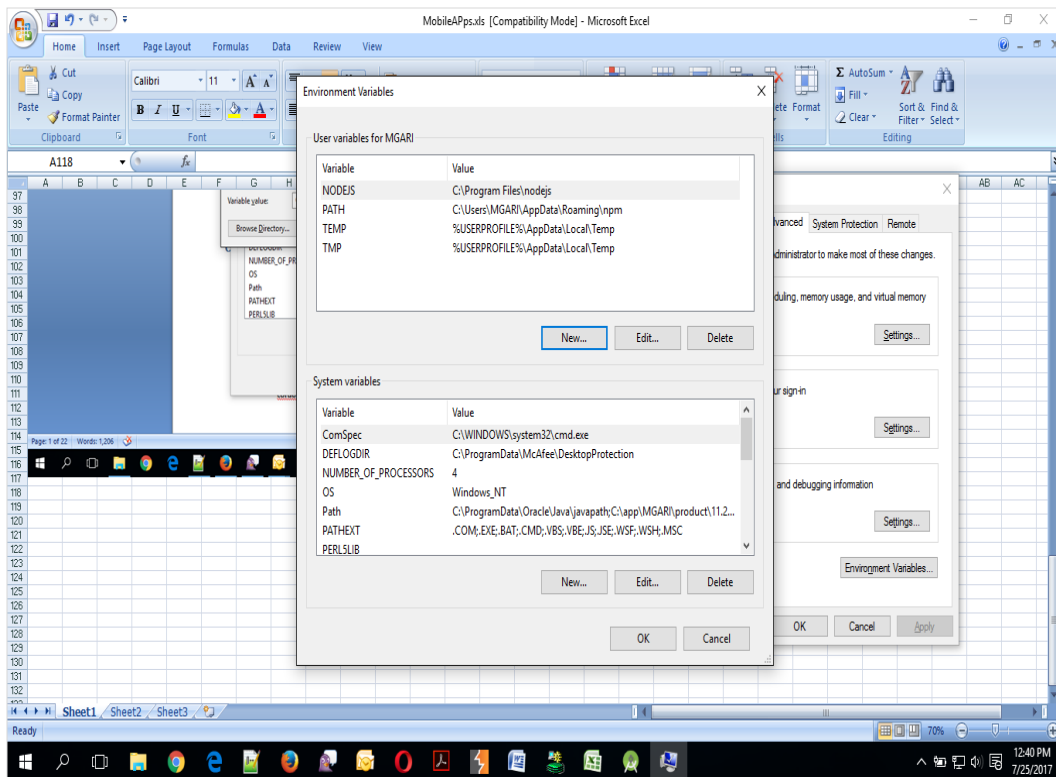
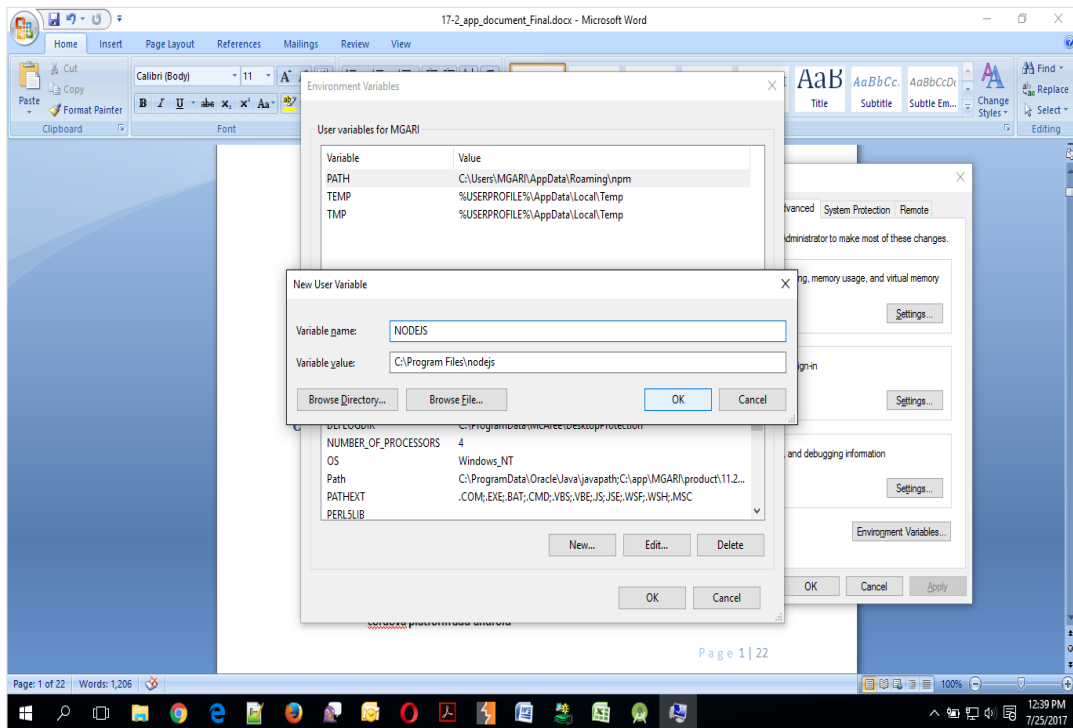
OBDX Android App is supported only on versions n (current) and n-1 release.

- a. Download and Install node Js (will be downloaded to default path)
- b. Install node js from <https://nodejs.org>
- c. DOWNLOAD AND INSTALL ANDROID STUDIO
- d. Download and install Android Studio from <https://developer.android.com/studio/index.html>
- e. Download and Install Android platforms
- f. Update Android SDK to latest API Level.
- g. Gradle Version: gradle-7.5
- h. Android Gradle Plugin Version (7.4.2): 'com.android.tools.build:gradle:7.4.2' or above
- i. Set Environment variables
- j. Set following system variables:
 1. Click on Windows key and type Environment Variables.
 2. A dialog box will appear. Click on the Environment Variables button as shown below



3. NODEJS <nodejs_path> Example: "C:\Program Files\nodejs\".

k. Add the above variables in “PATH” system variable.



2.2 Create project using Remote UI

a. Index.html changes (use Android Studio or any other editor)

- Update the server URL in app.properties against KEY_SERVER_URL key. This is the URL where the UI is also hosted.

After this proceed to **2.4 Importing in Android Studio** directly.

2.3 Local UI by running on local machine or local server.

Building un-built UI (required in case of customizations)

1. For this version, since the UI is built with webpack, the built UI cannot be modified from with the mobile workspace as it is minified code. Hence, either bank can hoist the UI is two ways:
 - Use local machine as local server and host the UI on local development machine and connect the application using localhost.
 - OR host the UI on local development server and point the application to that server URL
2. UI is same for internet and mobile, same build process of internet to be followed.
Bank can follow the UI build steps from “Oracle Banking Digital Experience User Interface Guide”.
3. Additionally, building UI for mobile, Open scripts->webpack->webpack.dev.js and add below line in devServer object:

as below:

```
headers: {  
    "Access-Control-Allow-Origin": "**"  
},
```

SAMPLE:

```
devServer: {  
    static: path.join(__dirname,  
        "../dist"),  
    compress: true,  
    port: 4000,  
    hot: false,  
    client: false,  
    headers: {  
        "Access-Control-Allow-Origin": "**"
```

$\},$

4. Also, in `webpack.dev.js` comment out below lines inside “entry” key.

```
entry: {  
  // main: "framework/js/configurations/require-config.js",  
  // Runtime code for hot module replacement  
  //hot: 'webpack/hot/dev-server.js',  
  // Dev server client for web socket transport, hot and live reload logic  
  //client: 'webpack-dev-server/client/index.js?hot=true&live-//reload=true',  
},
```

5. Once the UI is built, run below command to start a local server on the development machine using below command:

- `npm run start`

```
isaak@isaakpauls-mac: ~ % npm start
> obsd-build-tool@20.1.0 start
> webpack serve --open --config scripts/webpack/webpack.dev.js

<1> [webpack-dev-server] [HMR] Proxy created: /dix -> http://ossm-wm-715.snbomprshared1.gbucloud102bom.oracleevcn.com:17777/
<1> [webpack-dev-server] Project is running at:
<1> [webpack-dev-server] localhost: http://localhost:4000/
<1> [webpack-dev-server] On Your Network (IPv4): http://192.168.29.50:4000/
<1> [webpack-dev-server] On Your Network (IPv6): http://:::1:4000/
<1> [webpack-dev-server] Content not from webpack is served from: /Users/isaakpaul/Documents/work/svn/trunk/core/channel_11Sept/channel/dist* directory
<1> [webpack-dev-middleware] wait until bundle finished: *
```

- Once this server starts, below is the window which appears. This indicates local server is started.

```

critical dependency: require function is used in a way in which dependencies cannot be statically extracted
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/ sync ^\\.\.\\..*\.min/ojmodule-element-utils./ .min/ojmodule-element-utils.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojthemacmap.js 2617:47-149
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojthemacmap.js sync ^.*.*.*/ojthemacmap./ .ojthemacmap.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojconform.js 139:51-152
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojtranslation.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojconvertutils-i18n.js
@_framework/js/dom-util.js 6:0-61 446:15-58 643:0-609:2
@_framework/js/view-model/generic-view-model.js 2:18-9 29:14 50:5-20 56:5-20 62:5-20 84:5-26 165:21-28
@_framework/js/configurations/require-config.js 20:4-56

WARNING in @_node_modules/@oracle/oraclejet/dist/js/libs/oj/min/ojmodule-element-utils.js 8:558-565
critical dependency: require function is used in a way in which dependencies cannot be statically extracted
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/min/ojmodule-element-utils.js 8:558-565
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojthemacmap.js 2617:47-149
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojthemacmap.js sync ^.*.*.*/ojthemacmap./ .ojthemacmap.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojconform.js 139:51-152
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojtranslation.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojconvertutils-i18n.js
@_framework/js/dom-util.js 6:0-61 446:15-58 643:0-609:2
@_framework/js/view-model/generic-view-model.js 2:18-9 29:14 50:5-20 56:5-20 62:5-20 84:5-26 165:21-28
@_framework/js/configurations/require-config.js 20:4-56

WARNING in @_node_modules/@oracle/oraclejet/dist/js/libs/oj/min/ojmodule.js 9:2080-20907
critical dependency: require function is used in a way in which dependencies cannot be statically extracted
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/ sync ^\\.\.\\..*\.min/ojmodule./ .min/ojmodule.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojthemacmap.js 2617:47-149
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojthemacmap.js sync ^.*.*.*/ojthemacmap./ .ojthemacmap.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojconform.js 139:51-152
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojtranslation.js
@_node_modules/@oracle/oraclejet/dist/js/libs/oj/debug/ojconvertutils-i18n.js
@_framework/js/dom-util.js 6:0-61 446:15-58 643:0-609:2
@_framework/js/view-model/generic-view-model.js 2:18-9 29:14 50:5-20 56:5-20 62:5-20 84:5-26 165:21-28
@_framework/js/configurations/require-config.js 20:4-56

7 warnings have detailed information that is not shown.
Use 'stats.errorDetails: true' resp. '--stats-error-details' to show it.

repack 5.89.0 compiled with 27 warnings in 12461 ms

```

- Point the “key_server_url” to <http://localhost:4000> and run the application on simulator. To run on device, the internet proxy should allow localhost domain to accept incoming requests.

If it is blocked, UI should be built and “npm start” command should be executed on a development server machine which is accessible in the network. They “key_server_url” will then point to that local server URL instead of localhost

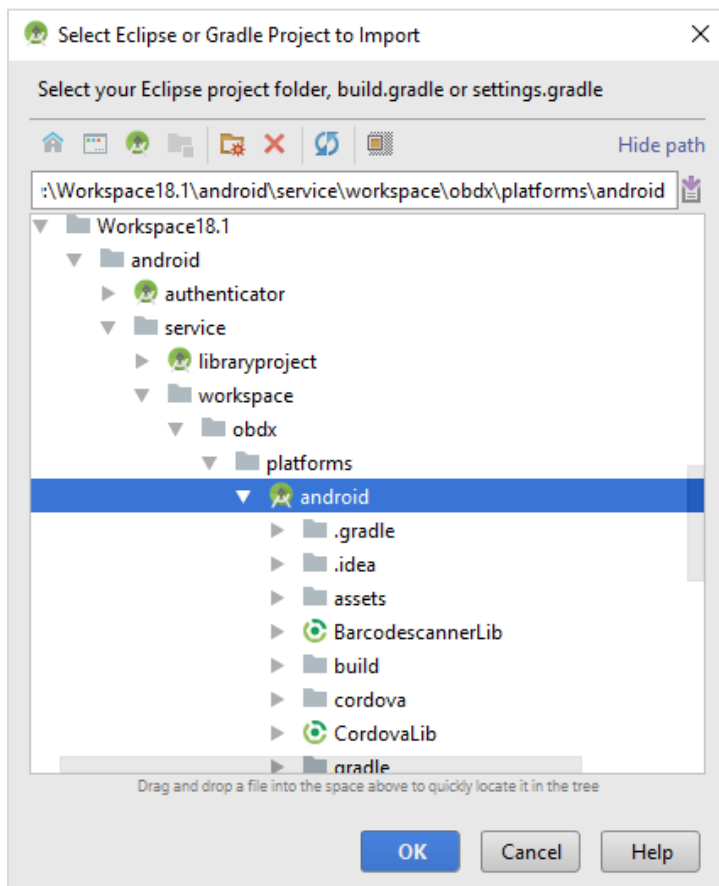
Note: Proper SSL & proper domain needs to be configure to run this on android as android won't support for http url.

6. If banks want to debug UI the update “devtool” configuration. Refer Webpack documentation <https://webpack.js.org/configuration/devtool/> for more details.

2.4 Importing in Android Studio

Open Android Studio

1. Import zigbank/platforms/android in android studio by clicking on Open an Existing Project.



2.5 Widget Functionality

Widgets are Android native feature. Below widgets are available in the application

1. All Accounts Widgets – Widget, showing all accounts balances & account numbers.
2. Account Details Widget - Widget, showing account balance of default account and last 5 transactions of the same account, can be added to the phone home screen. If default account is not set, then the details of the account fetched first is shown.
3. Multi-Functional Widget – Widget showing default account balance. If default account is not present, it shows details of account fetched first. Additionally, it has option to scan to pay feature

4. Scan to Pay Widget – Widget which allows to scan to pay.

Pre-requisite:

Quick Snapshot feature needs to be enabled in the app application from the login screen. (Refer function doc - User Manual Oracle Banking Digital Experience Quick Snapshot.docx)

Enable below property in app.properties file

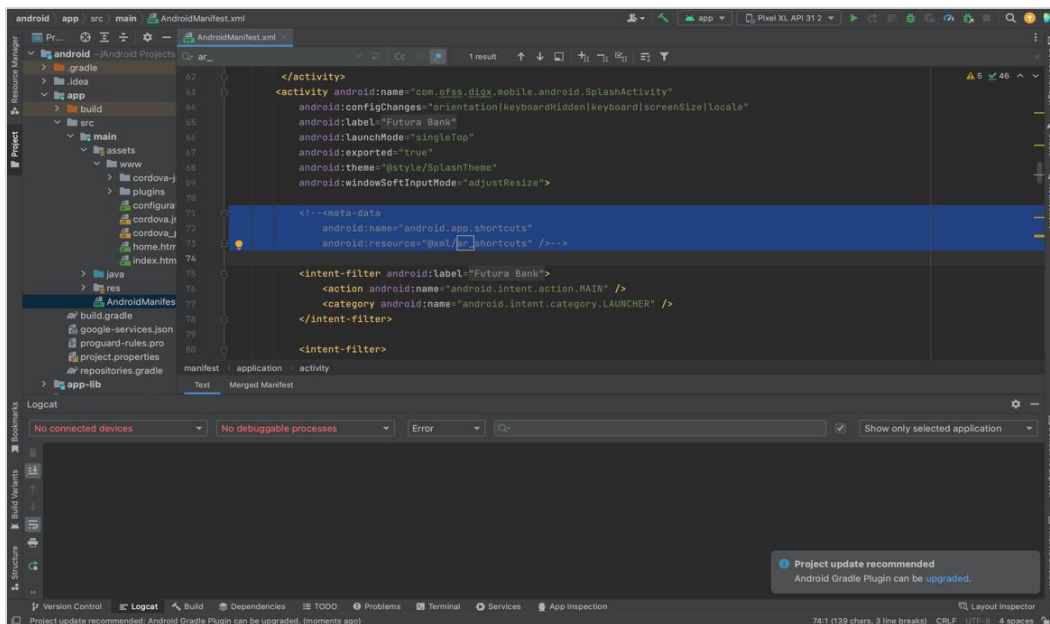
```
<bool name="ENABLE_WIDGET">true</bool>
```

If bank does not want this feature, then they can disable this by making above flag to false.

2.6 Scan to Pay from Application Icon –

Users can long press on bank's application icon on home screen and click on scan-to-pay option to scan QR and make payments.

To enable this feature uncomment below from app's AndroidManifest.xml



2.6 Passkey (Passwordless login)

Passkeys are a safer and easier replacement for passwords. With passkeys, users can sign in to apps and websites using a biometric sensor (such as a fingerprint or facial recognition), PIN, or pattern. This provides a seamless sign-in experience, freeing your users from having to remember usernames or passwords.

Passkeys are supported only on devices that run Android 9 (API level 28) or higher

TO DISBALE THIS OPTION:

By doing this, passkey option will not be available to users withing the application. User will not be able to register for passkey and also will not be able to login using passkey. Follow below steps

- a. Remove RTM access from Client Servicing -> Authentication -> Passkey Setup for Mobile Application/Mobile (Responsive) and Internet touch points



- b. Set this flag in channel-framework-js-configurations-config.js to false

thirdPartyAPIs -> passkey -> required -> false

TO ENABLE THIS OPTION:

1. Add RTM access from Client Servicing -> Authentication -> Passkey Setup for Mobile Application, Mobile (Responsive) and Internet touch points



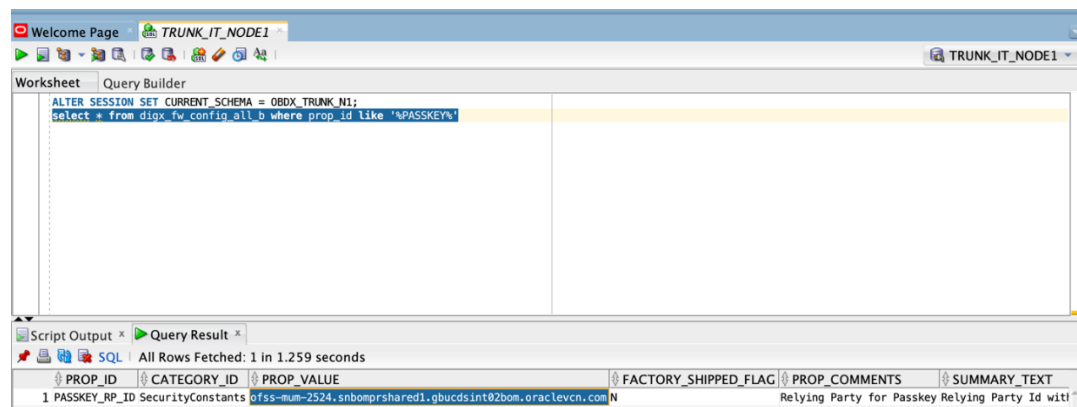
2. Set this flag in channel-framework-js-configurations-config.js to true

thirdPartyAPIs -> passkey -> required -> true

3. Along with above, we need below server side and application side setup

Server-Side Setup:

1. Update the relying party in below property select prop_value from digx_fw_config_all_b where prop_id='PASSKEY_RP_ID'



Note: Relying partId is the domain name if the website to which credentials will be associated. (Eg google.com, example.com etc)

Relying party origin is the relying party of website prefixed with protocol without the port.

(Example <https://google.com>, <https://example.com>)

- a. Create assetlinks file (assetlinks.json) -

A Digital Asset Links JSON file must be published on your website to indicate the Android apps that are associated with the website and verify the app's URL intents.

The following example assetlinks.json file grants link-opening rights to a com.example Android app:

```
{
  "relation": ["delegate_permission/common.handle_all_urls"],
  "target": {
    "namespace": "android_app",
    "package_name": "com.example",

    "sha256_cert_fingerprints":["14:6D:E9:83:C5:73:06:50:D8:EE:B9:95:2F:34:FC:64:16:A0:
83:42:E6:1D:BE:A8:8A:04:96:B2:3F:CF:44:E5"]

  }
}
```

The JSON file uses the following fields to identify associated apps:

package_name: The application ID declared in the app's build.gradle file.

sha256_cert_fingerprints: The SHA256 fingerprints of your app's signing certificate. You can use the following command to generate the fingerprint via the Java keytool:

```
keytool -list -v -keystore my-release-key.keystore
```

b. Publish assestlinks.json file-

This file needs to be on https server with valid SSL certificate

You must publish your JSON verification file at the following location:

<https://domain.name/.well-known/assetlinks.json>

For example, if your sign-in domain is signin.example.com, host the JSON file at <https://signin.example.com/.well-known/assetlinks.json>.

Verify your assetlink json on below statement list tester-

<https://developers.google.com/digital-asset-links/tools/generator>

The MIME type for the Digital Assets Link file needs to be JSON. Make sure the server sends a Content-Type: application/json header in the response.

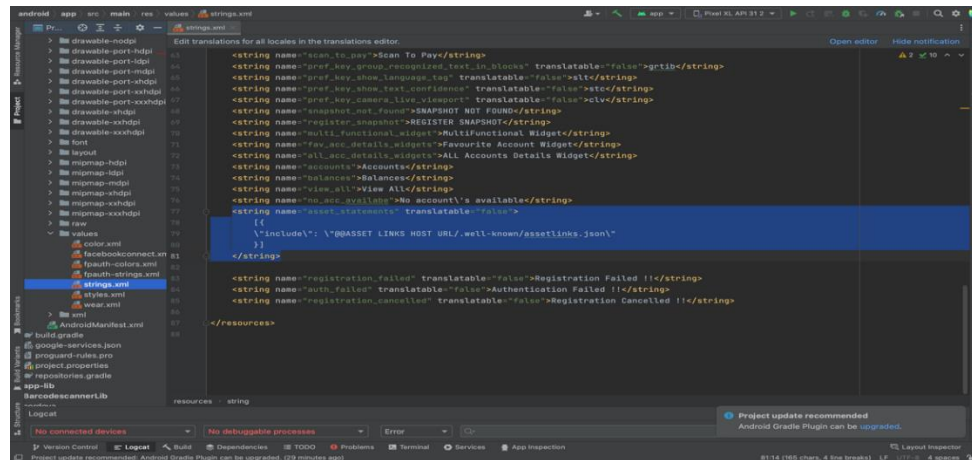
Need to change host and port in Obdx.conf as,

ProxyPass "/.well-known" "http://100.76.157.55:7003/digx-admin/sms/v1/.well-known"

ProxyPassReverse "/.well-known" "http://100.76.157.55:7003/digx-admin/sms/v1/.well-known"

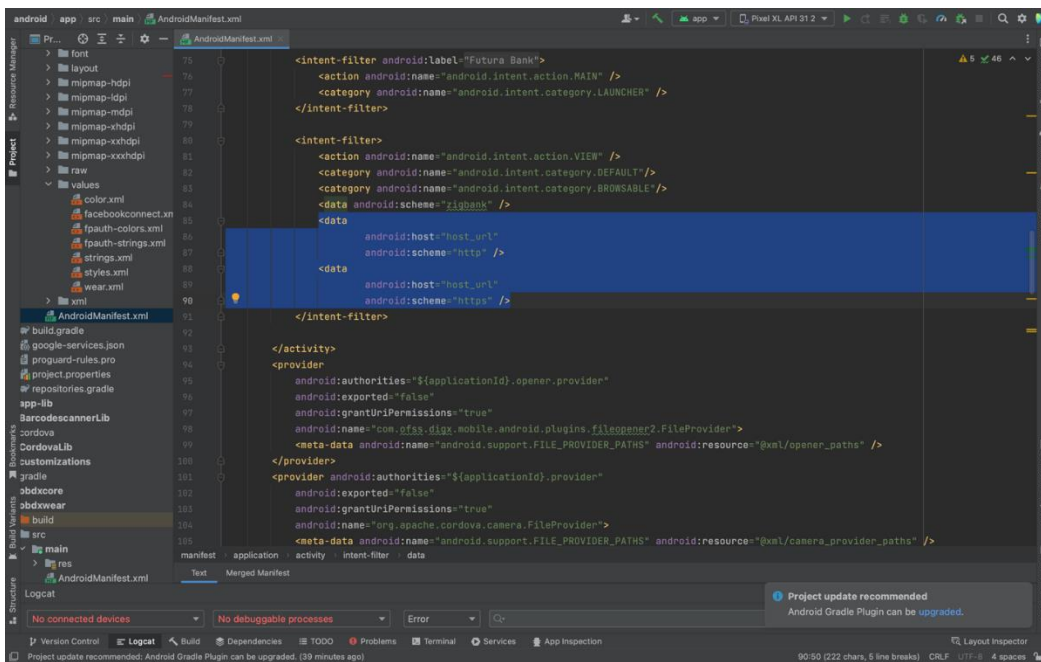
After the setup is done, this file must be accessible on mobile browser with this url. There should not be any redirects for accessing this file.

- c. Add assetlinks.json file host in app's strings.xml file.



2.7 Deeplinking - To open reset password, claim money links with the application

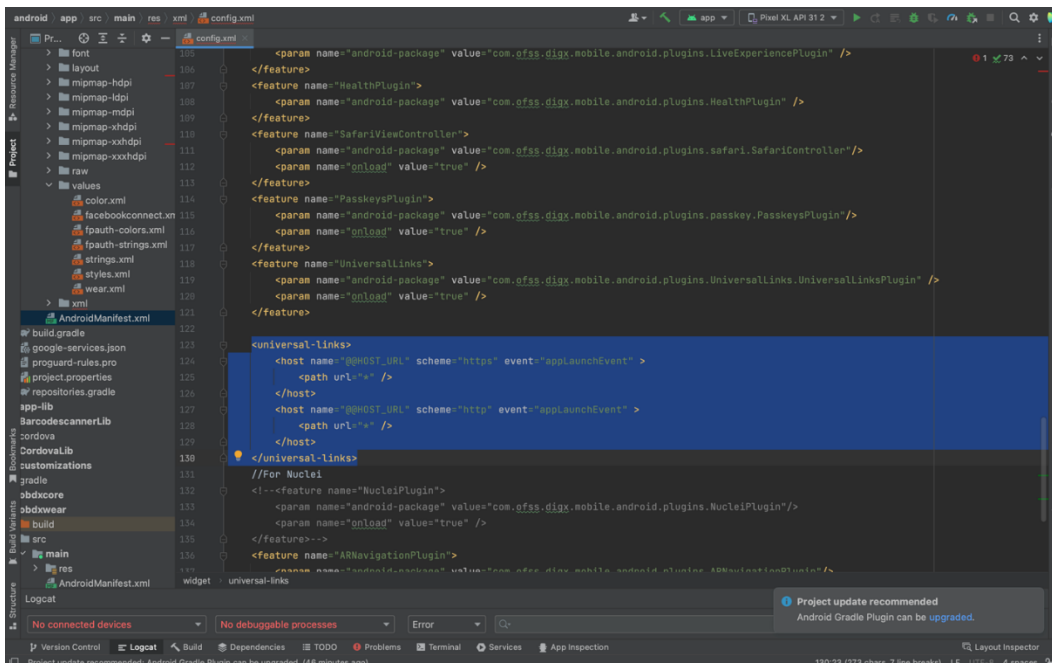
Add host url under data tag in app's AndroidManifest.xml as,



Note: Add host url without https or http.

For example: If your deeplink url is <https://exmple.com/test> then you can add only [example.com](https://exmple.com/test) in the data tag

Similarly you can add the same host url in app's config.xml under universal-links tag as,



2.8 Device Registration and Push Registration Functionality

In this version, only one device is allowed to be registered for alternate login for the same username. If user tries to register another device with same username for alternate login, then the previous registration on other devices will be removed. User will get an error message if he/she tries to use PIN/PATTERN/BIOMETRIC on the de-registered devices.

While user registers his second device or same device again (by re-installing the application), a popup will appear to notify the same.

If user confirms, then the current device will be registered, and all previous registrations will be removed.



If user cancel, the process is exited.

Also, in this version, only one device is allowed to be registered for push.

Bank can allow multiple devices to be registered for same username in their setup by setting below two configurations:

ALLOWED_DEVICE_COUNT to any value between than 1 and 100.

- 1 will allow on one device registration.
- 100 will allow more than one device registration

ALLOWED_PUSH_DEVICE_COUNT any value between 1 and -1

- 1 will only one one device to be registered for push.
- -1 will only multiple devices to be registered for push

2.9 **Location Tracking Metrics**

This is optional. Bank needs to do if they need location tracking metrics for monitoring location-based data.

ALLOW_LOCATION_SHARE

By default, the value is false. If set to true, user will get location permission prompt to allow location tracking. It can be enabled if user's location needs to be tracked.

2.10 **Displaying Rate Option to Redirect to Playstore Page**

This is optional. User can have an option ("Rate Us") in settings to display Play Store rating for the application. This option can be enabled/disabled from UI.

Note: App should be listed on playstore before adding this functionality.

2.11 **Enabling Force Update**

This configuration is optional.

To notify users of a new application version available on the Play Store, consider these options:

1. Within App, when the App detects a new version, prompt users suggesting an update.
2. The flag checks for updates and displays a cancellable popup to the user to update their application.
3. To implement this with the flag `isAppUpdateManagerEnable` to true in `RootCheckFlags`.

Note: Ensure that App update functionality works only when the App is downloaded from the Play Store or via Internal App Sharing.

4. Follow the steps to check force app update: <https://developer.android.com/guide/playcore/in-app-updates/test#internal-app-sharing>

2.12 Splash Screen Migration

The splash screen implementation is migrated according to latest document from google:

<https://developer.android.com/develop/ui/views/launch/splash-screen/migrate>

Steps to generate xml file for svg to be used in splash:

1. Right click on /android/app/src/main/res/drawable and select New/Image Asset
2. Select the path to the svg. (note that svg of bank logo is required. PNG and other image extensions won't work)
3. Resize the image from the scroll bar so that the icon is well inside the circle.
4. Keep all the configurations as it is and create the svg.
5. It will directly generate xml files for different resolution.
6. Refer to the foreground xml in styles.xml @drawable/ic_launcher_foreground

2.13 App Update Manager

Note: In App Update functionality will be work only for the apps which will be downloaded from play store/internal app sharing.

Follow below doc to test the in app update functionality.

<https://developer.android.com/guide/playcore/in-app-updates/test>

2.14 Auto OTP Configuration

Note: User consent otp popup will not be seen from now. OTP will be directly auto populated in the field once the sms is received. For this, 11 digit app hash code is required.

To Generate the 11 digit application hash, follow below doc -

https://developers.google.com/identity/sms-retriever/verify#computing_your_apps_hash_string

f you are signing your app with your own keystore then download below script -

https://github.com/googlearchive/android-credentials/blob/master/sms-verification/bin/sms_retriever_hash_v9.sh

Then run below command -

`./sms_retriever_hash_v9.sh --package YOUR-PKG-NAME --keystore YOUR-KEYSTORE-PATH`

Add this 11 digit app hashcode at the end of each text msg template

You will get the msg template in below table -

```
select * from digx_ep_msg_tmpl_b where txt_msg_tmpl like '%OTP%' and destination_type like 'SMS';
```

Sample SMS will be look like below –

Your OTP is 1111 FA+9qCX9VSu

3. Google Play Integrity

- a. Go to URL <https://console.developers.google.com/>
- b. Create a new Project and set name of you project

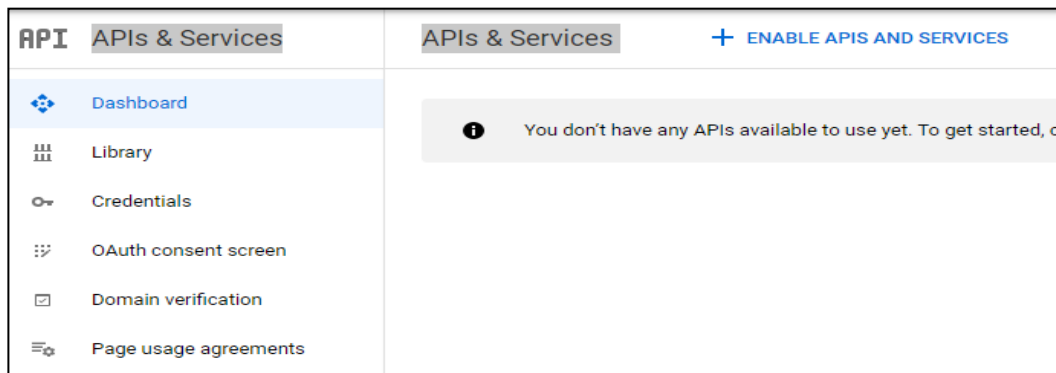
New Project

Project name ?

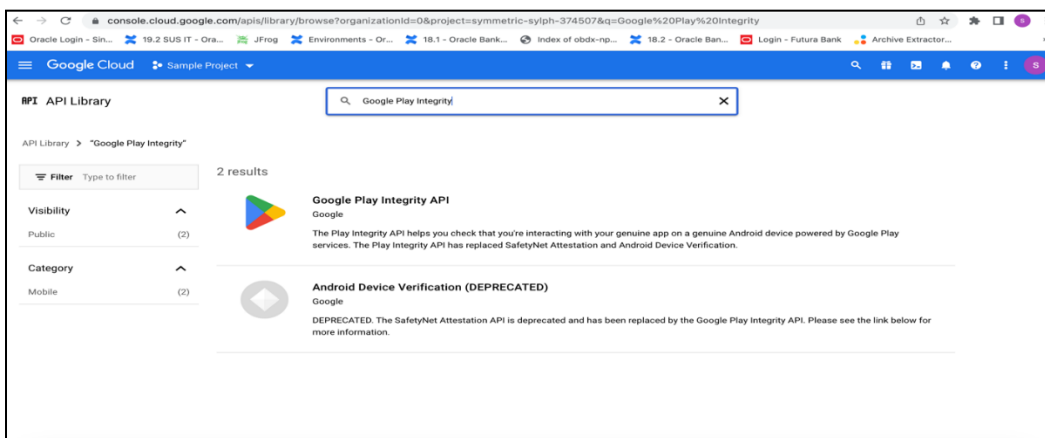
Your project ID will be safetynet-161214 ? [Edit](#)

[CANCEL](#) [CREATE](#)

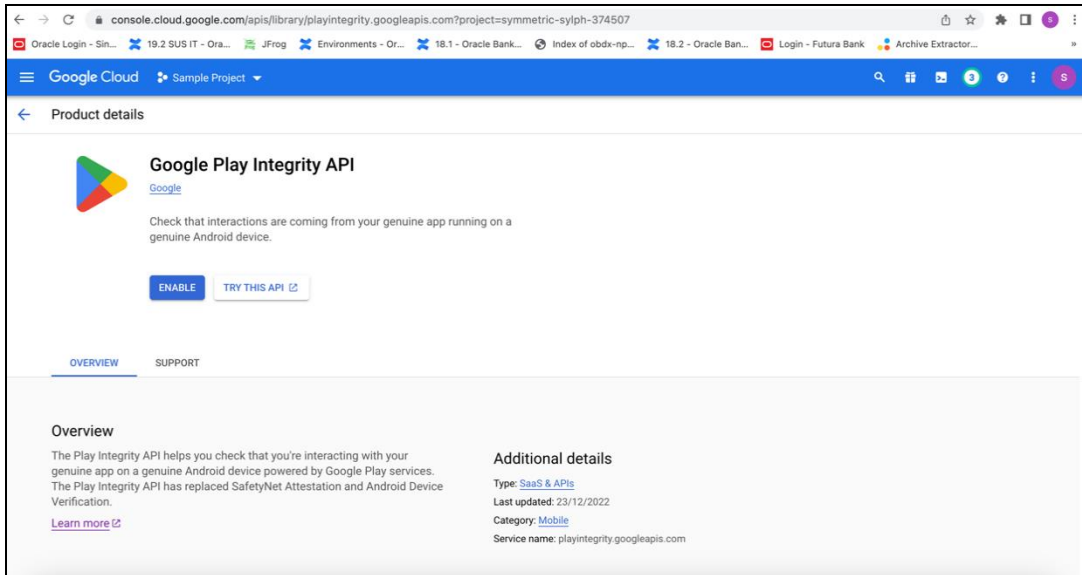
- c. Choose **'API's & Services'** option from side bar.
- d. In API's & Services > Dashboard > Choose **'Enable APIS AND SERVICES'**.



- e. This will redirect to **'Library'** where we need to search **'Google Play Integrity API'**.



- f. Click on Google Play Integrity API and enable it



g. If the application usage is high, the quota request form needs to be submitted. Fill quota request form from below site. Also select below options.

<https://support.google.com/googleplay/android-developer/contact/piagr>

The screenshot shows the Google Play Console Help page for the Play Integrity API. The page has a header with 'Play Console Help' and a search bar. The main content area is titled 'Play Integrity API' and contains a description of the API. Below the description, there is a section titled 'You can use the form below to provide feedback, report issues, or request to increase your daily maximum number of requests.' This section includes instructions on how to use the form and a note that the form is only for requests in English, Chinese, Japanese, and Korean. Below the instructions, there are three radio buttons for selecting the reason for the request: 'Increase maximum number of daily requests' (which is selected), 'Provide feedback', and 'Report issue'. At the bottom of the form, there is a text input field labeled 'Name of requesting organization/person'.

support.google.com/googleplay/android-developer/contact/piaqr

Play Console Help

Describe your issue

How are you calling the Play Integrity API? *

☒ My app is calling the API directly

☐ A third party I'm using in the app is calling the API, please specify

How often will you call the API for each user? *

☐ Once per day or less

☐ Once per hour

☐ Once per 15 min

☒ Once per 5 min or more

Is there any PII or SPII used for the nonce (e.g. user id, user name, phone number, Android ID, SSN, etc)? *

☐ Yes, but hashed or encrypted

☐ Yes, in plain-text

☒ No

support.google.com/googleplay/android-developer/contact/piaqr

Play Console Help

Describe your issue

How are you validating Play Integrity API responses? *

☐ Server side - by calling Play's server to decrypt and verify

☒ Server side - by decrypting and verifying with self-managed API keys

☐ In my app - by calling Play's server to decrypt and verify

☐ In my app - by decrypting and verifying with self-managed API keys

☐ Other, please specify

How does your app retry in case of Play Integrity API errors? *

☒ No retry

☐ A small number of retry attempts within a short time window

☐ Retry with exponential backoff

☐ Other, please specify

support.google.com/googleplay/android-developer/contact/piaqr

Play Console Help

Describe your issue

How will your app act when the Play Integrity API detects risky traffic? *

Please answer with your end goal in mind even if your app is not acting yet. As a reminder, your app should also be able to deal with Play Integrity API errors and the API being unavailable.

☒ Deny access to functionality (for example, users won't be able to log-in). I want unauthorized usage of my app to go down.

☐ Alter or limit specific features (for example, only users on good devices will be allowed on a leaderboard). Overall usage of my app might stay the same.

☐ A mix - deny access for some responses and change features for other responses. I want some unauthorized usage of my app to go down.

☐ No action. I'm only collecting data.

☐ Other, please specify

Quota request - Estimated total queries per day *

☐ 10,000 to 1,000,000 (10K to 1M)

☐ 1,000,000 to 10,000,000 (1M to 10M)

☐ 10,000,000 to 100,000,000 (10M to 100M)

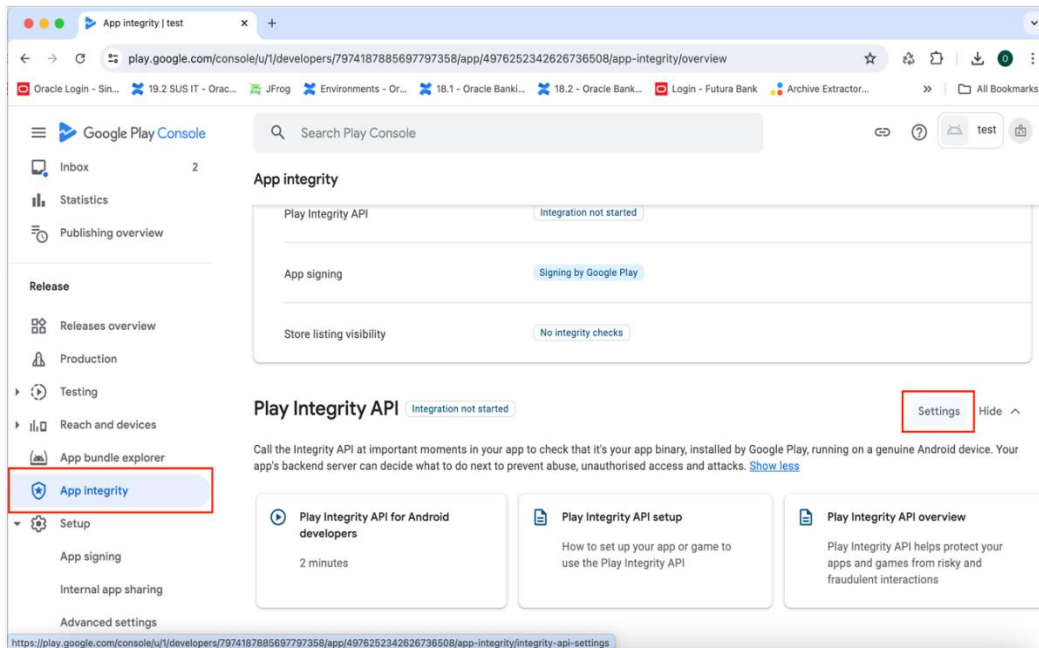
☐ 100,000,000 or more (100M+)

Quota request - Estimated total queries per day * → The approximate load, Play Integrity API is called once each time the app is opened

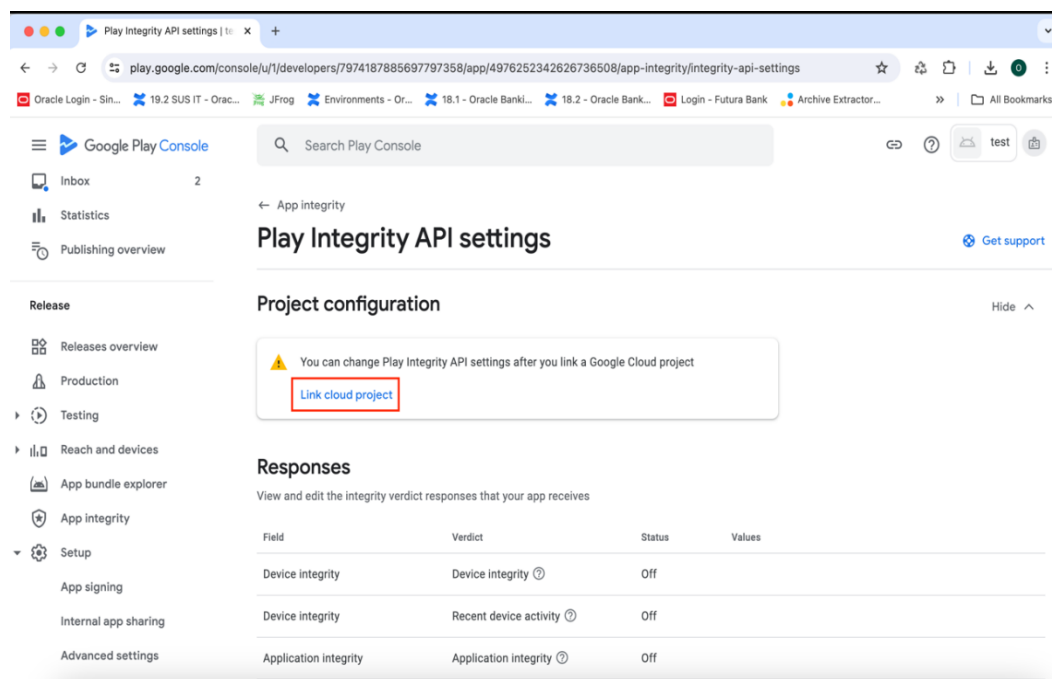
Quota request - Estimated peak queries per second → Leave blank

h. To enable Play Integrity responses please follow below steps-

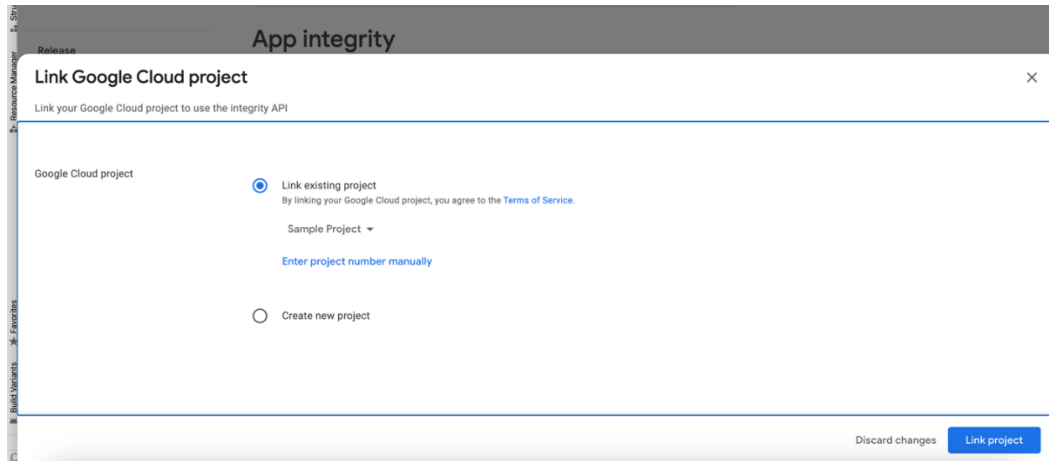
Go to Google Play Console->Side Menu ->App Integrity



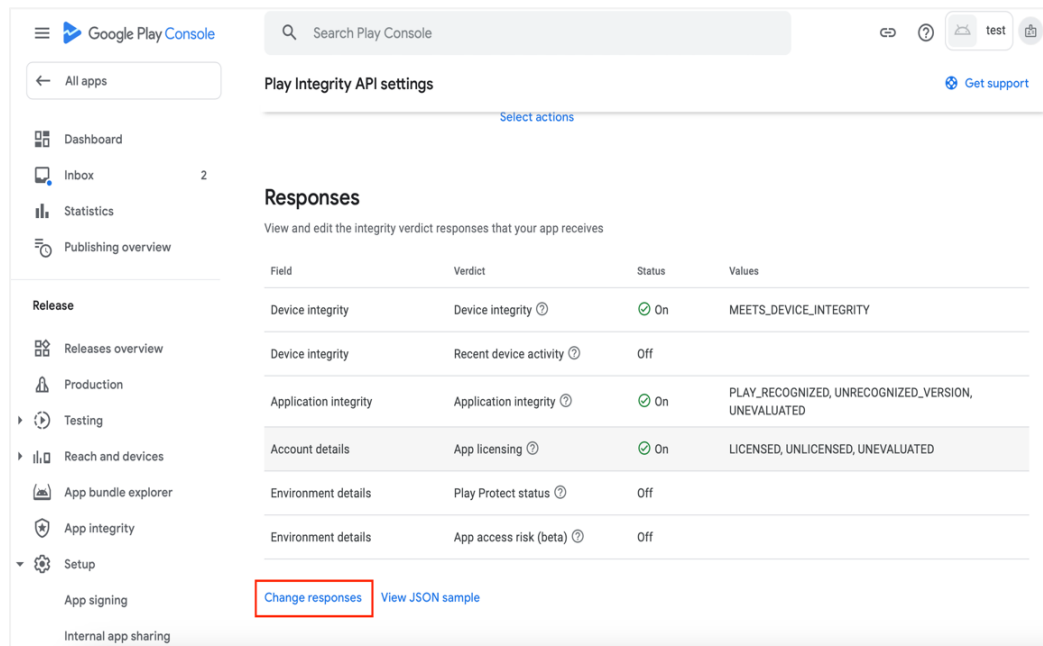
Click on **Settings**.



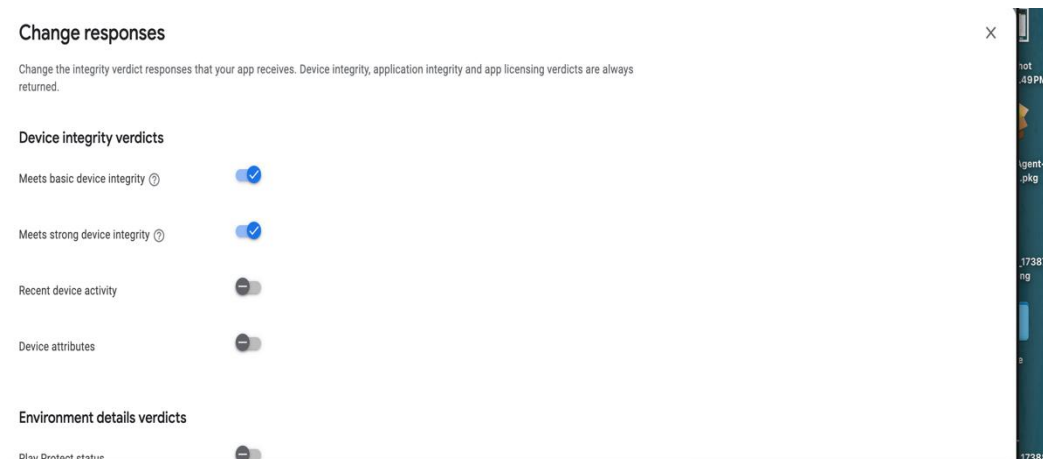
Click on **Link project** and then link your existing google cloud project. If it is not created then create new and link the same.



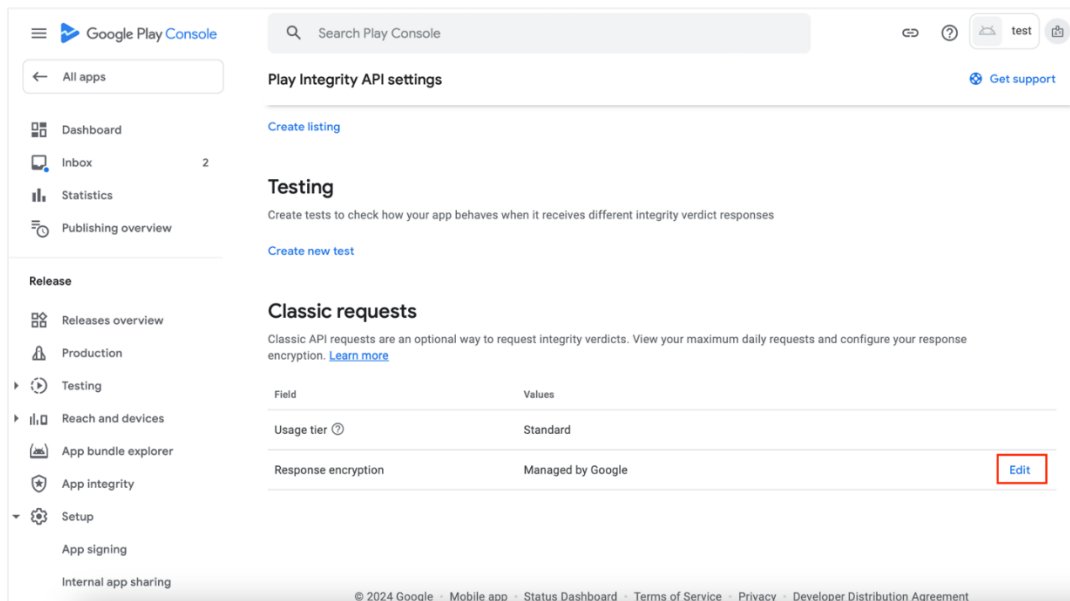
i. Scroll down on the same screen and click on **Change Responses**.



j. Enable the Meet basic Integrity & Meets Strong Integrity option and save the changes.



k. Scroll down on the same screen and click on **Edit** button of classic requests section



l. In the window that appears, select **Manage and download my response encryption keys** and follow below steps to generate response encryption keys-

a. Create a new private-public key pair. RSA key size must be 2048 bits using below command-

```
openssl genrsa -aes128 -out your_path/private.pem 2048
```

Then use your password phrase for creating private.pem and also use the same password for verifying the private.pem. Then hit the below command.

```
openssl rsa -in your_path/private.pem -pubout -out your_path/public.pem
```

Enter the same password which you have used while creating private.pem. These two files will now appear on your mentioned path. Then upload the public.pem file on the window which was appeared after clicking on Manage and download my response encryption keys option. Once you upload the public.pem file it will automatically download your_app_pkg_name.enc file. Then hit below command as,

```
openssl pkeyutl -decrypt -inkey your_path/private.pem -pkeyopt rsa_padding_mode:oaep -in your_path/com.demo.xz.enc > your_path/api_keys.txt
```

Enter the password for private.pem. It will create api_keys.txt file on your path. It must be consist of VERIFICATION_KEY and DECRYPTION_KEY.

b. Maintain this VERIFICATION_KEY and DECRYPTION_KEY in **DIGX_FW_CONFIG_ALL_B** table corresponding to the following keys respectively:

PLAY_INTEGRITY_ENCRYPTION_KEY and **PLAY_INTEGRITY_DECRYPTION_KEY**

An example query will be:

```
update DIGX_FW_CONFIG_ALL_B set prop_value = 'YOUR_DECRYPTION_KEY' where prop_id = 'PLAY_INTEGRITY_DECRYPTION_KEY';
```

```
update DIGX_FW_CONFIG_ALL_B set prop_value = 'YOUR_ENCRYPTION_KEY' where  
prop_id = 'PLAY_INTEGRITY_ENCRYPTION_KEY';
```

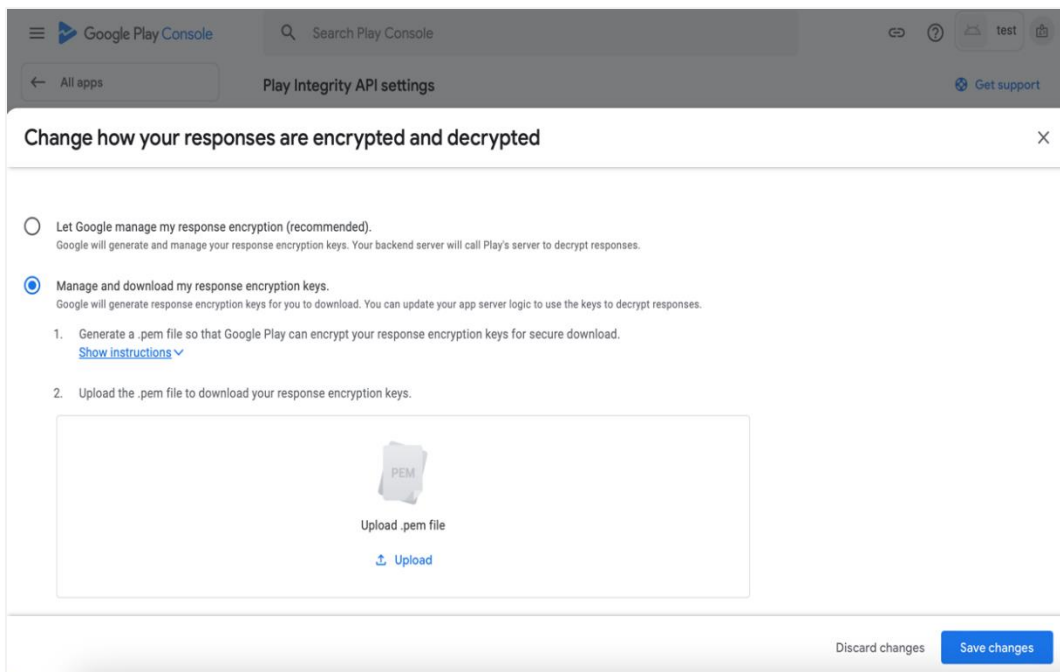
c. Similarly, Obtain the same keys for authenticator app by using above steps and then maintain those in **DIGX_FW_CONFIG_ALL_B** table corresponding to the following keys respectively:

PLAY_INTEGRITY_ENCRYPTION_KEY_AUTHENTICATOR and
PLAY_INTEGRITY_DECRYPTION_KEY_AUTHENTICATOR

An example query will be:

```
update DIGX_FW_CONFIG_ALL_B set prop_value = 'YOUR_DECRYPTION_KEY' where  
prop_id = 'PLAY_INTEGRITY_DECRYPTION_KEY_AUTHENTICATOR';
```

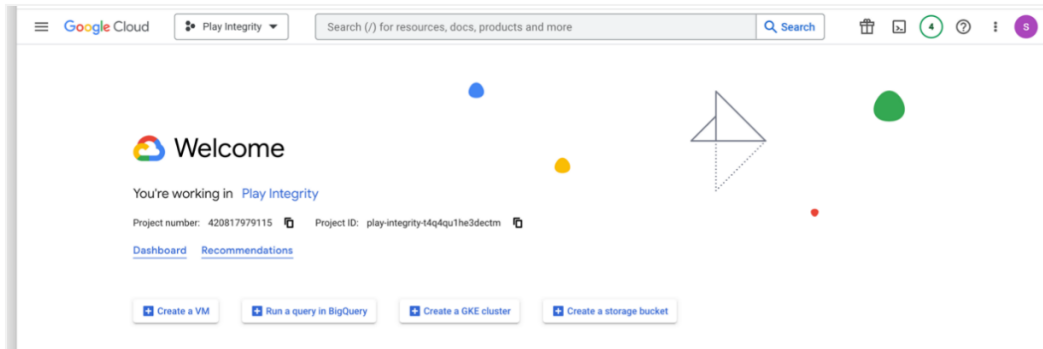
```
update DIGX_FW_CONFIG_ALL_B set prop_value = 'YOUR_ENCRYPTION_KEY' where  
prop_id = 'PLAY_INTEGRITY_ENCRYPTION_KEY_AUTHENTICATOR';
```



m. Add project number in below property of app.properties

```
<string name="GOOGLE_CLOUD_PROJECT_NO">@@GOOGLE_CLOUD_PROJECT  
NO</string>
```

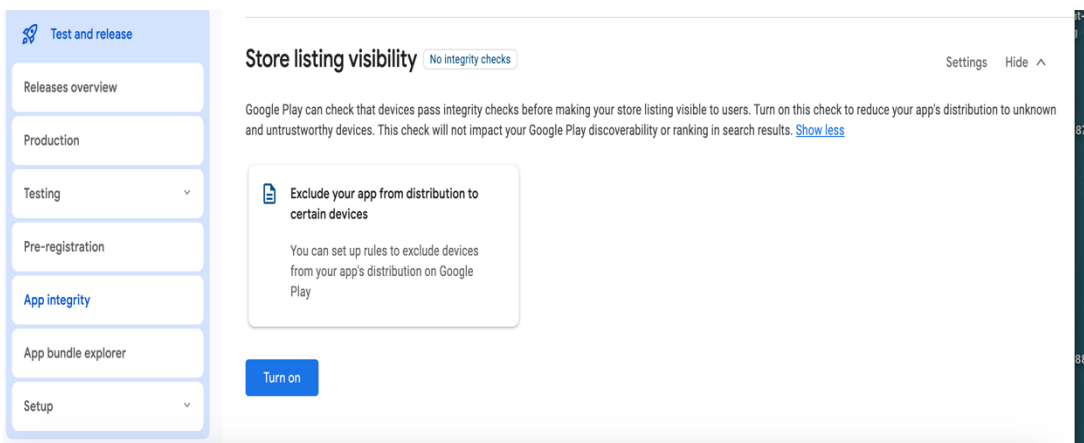
You will get the project number on google cloud console project



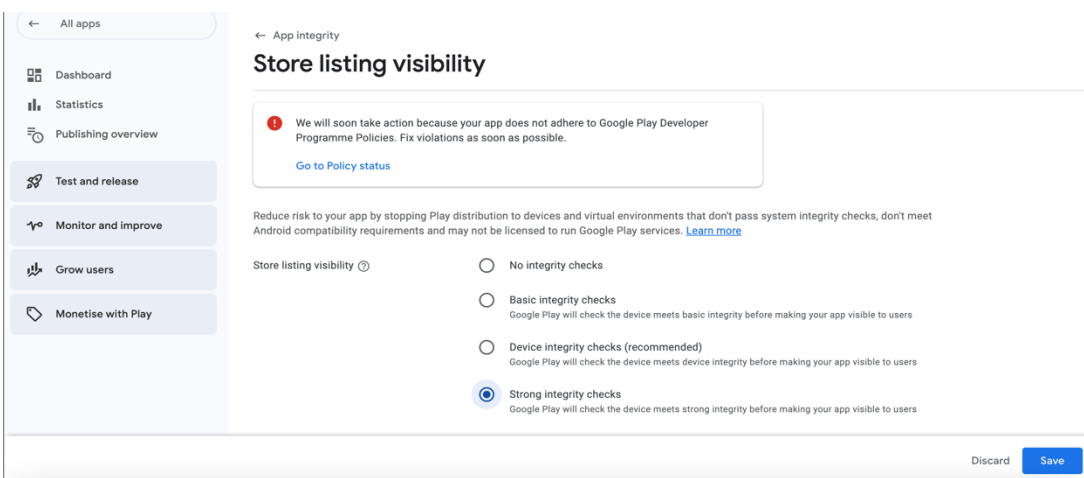
n. Mention the time in seconds to which app can hit the play integrity api. By default it is 300seconds but you can configure as per the requirement. Use below property in RootCheckFlags.java(workspace_installer/zigbank/platforms/android/app/src/main/java/com/ofss/digx/mobile/android/)

long playIntegrityAPICallTime = your_time_in_seconds;

I. Scroll down on the App Integrity Page. Navigate to Store listing visibility. Click on Settings button.



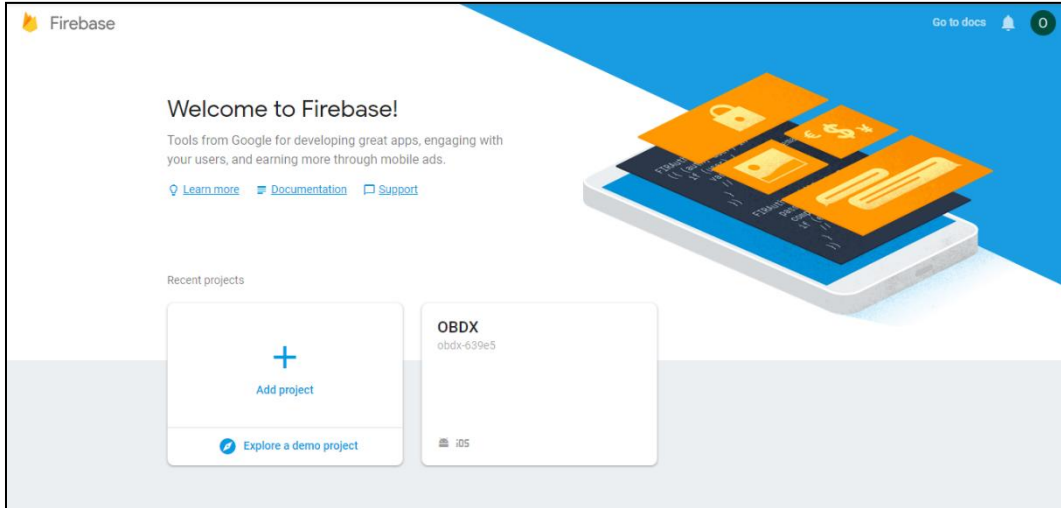
Select Strong Integrity checks option & Save



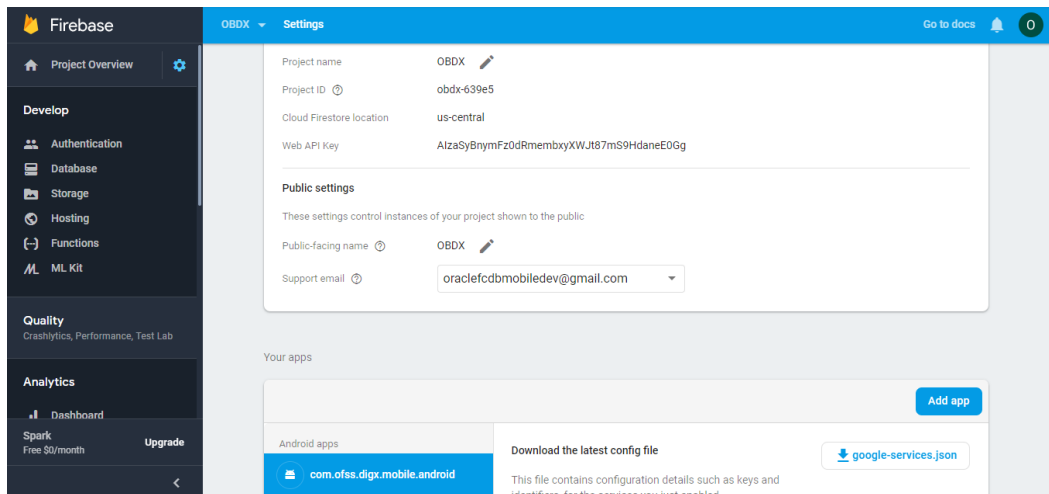
Note: By enabling this setting your app will not be listed on play store of rooted device.

4. FCM Push Notifications

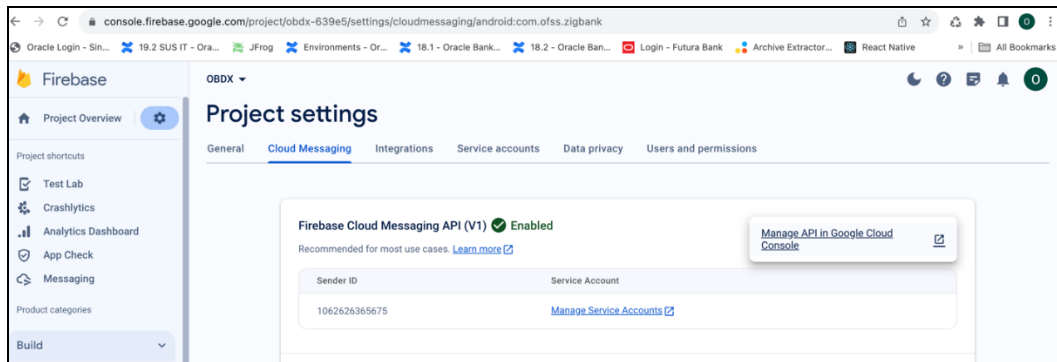
- a. Go to URL <https://firebase.google.com/>
- b. Traverse to console and create a project



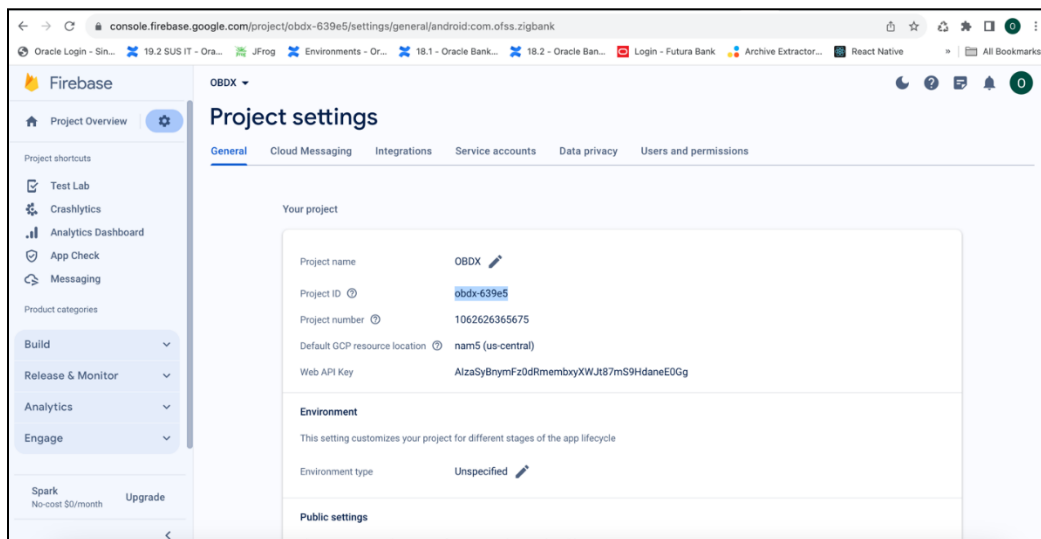
- c. Download google-services.json from below page and save to (zigbank\platforms\android\app) directory.
- d. Remember to keep the projects package name and firebase package name same.



e. Traverse to cloud messaging tab Enable Firebase Cloud Messaging API(V1) by clicking on Manage API in Google Cloud Console.



f. Get the Project ID from Project Setting in Firebase console



g. Update FCM URL in below table as-

update DIGX_FW_CONFIG_ALL_B set prop_value =
'https://fcm.googleapis.com/v1/projects/YOUR_PROJECT_ID/messages:send' where prop_id
= 'FCM_URL';

Add YOUR_PROJECT_ID in url which is captured on above step

h. If proxy address is to be used, provide the same in database as mentioned in point 3.

i. Generate private key for your service account by using below steps-

- In the Firebase console, open **Settings** > [Service Accounts](#)

- Click **Generate New Private Key**, then confirm by clicking **Generate Key**

You can also follow below google doc -

<https://firebase.google.com/docs/cloud-messaging/auth-server#provide-credentials-manually>

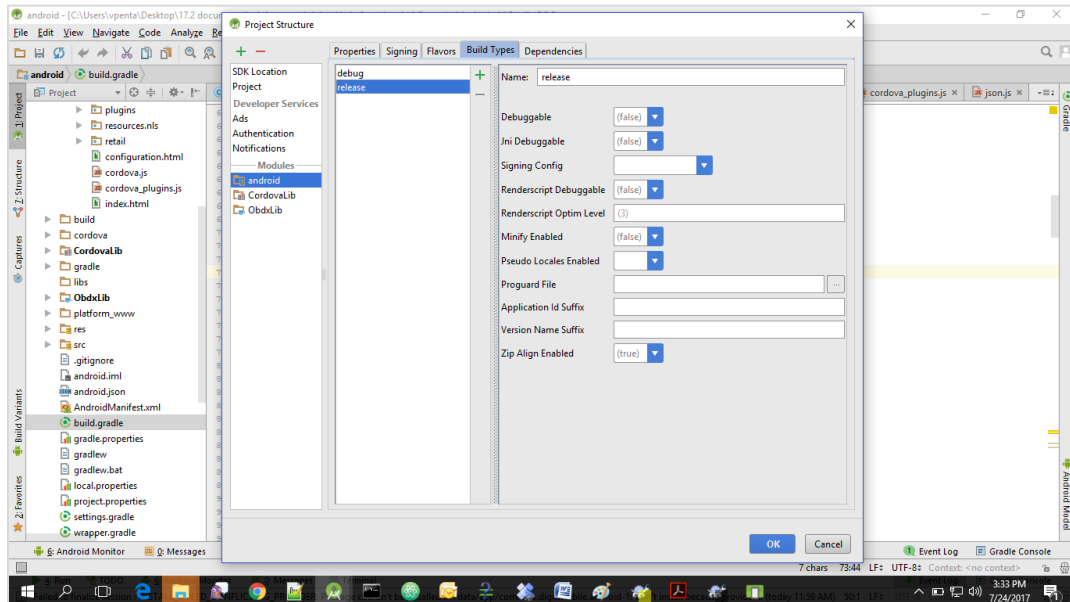
Sr. No.	Table	PROP_ID	CATEGORY_ID	PROP_VALUE	Purpose
1	DIGX_FW_CONFIG_VARS_B	FCM	DispatchDetails	<Server_Key>	Service account json file content captured in above step
2	DIGX_FW_CONFIG_ALL_B	FCMKeyStore	DispatchDetails	DATABASE or CONNECTOR	Specifies whether to pick server key from database or from connector. Default DB (No change)
3	DIGX_FW_CONFIG_ALL_B	Proxy	DispatchDetails	<protocol,proxy_address>	Provides proxy address, if any, to be provided while connecting to APNS server. Delete row if proxy not required. Example: HTTP,148.50.60.8

If CONNECTOR is selected in Step 2 update password as below

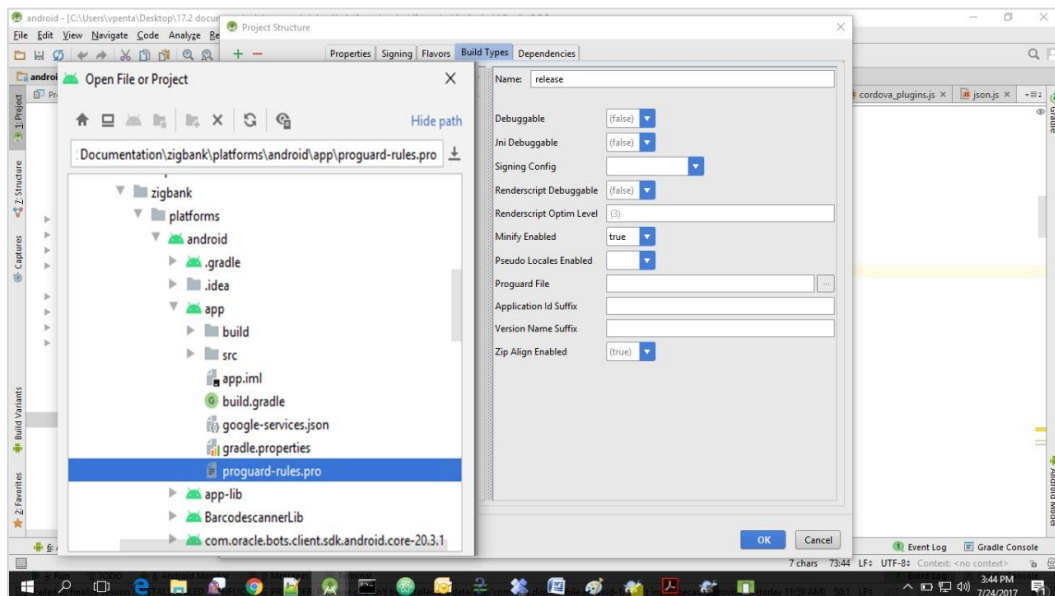
The screenshot shows the Oracle WebLogic Server Administration Console. The main window displays the 'Create a New Security Credential Mapping' dialog. The dialog has a title bar 'Create a New Security Credential Mapping' and buttons 'Back', 'Next', 'Finish', and 'Cancel'. Below the title bar, there is a section 'ETS User Name and Password' with the instruction 'Configure the ETS User Name and Password that you would like to map the WebLogic Server User to:'. Below this, there are three input fields: 'Enter the ETS User Name:' with the value 'FCM', 'Enter the ETS Password:' with a masked password, and 'Confirm Password:' with a masked password. The left sidebar shows the 'Domain Structure' with a tree view containing 'obdx_domain', 'Domain Partitions', 'Environments', 'Deployments', 'Services', 'Security Realms', 'Interoperability', and 'Diagnostics'. Below this is the 'How do I...' section with a link 'Create outbound credential mappings'. At the bottom of the sidebar is the 'System Status' section showing 'Health of Running Servers as of 10:32 AM' with a bar chart indicating 'Failed (0)', 'Critical (0)', 'Overloaded (0)', 'Warning (0)', and 'OK (2)'. The top navigation bar includes 'Home', 'Log Out', 'Preferences', 'Record', and 'Help'.

5. Build Release Artifacts

1. Clean and Rebuild your project in Android Studio.
2. In Android Studio, on the menu bar Click on **Build -> Edit Build Types -> select release**



3. Set Minify Enabled -> True & click on Proguard File selection -> Navigate to proguard-rules.pro (zigbank\platforms\android\app)



4. Click on OK -> again click on OK.
5. Adding URLs to app.properties.xml (customizations/src/main/res/values/)
 - a. NONOAM (DB Authenticator setup)

SERVER_TYPE	NONOAM
KEY_SERVER_URL	Eg. https://mumaa012.in.oracle.com:18443
WEB_URL	Eg. https://mumaa012.in.oracle.com:18443
SERVER_CERTIFICATE_KEY	Refer point 6.7

- b. OBDXTOKEN (Token based mechanism)

SERVER_TYPE	OBDXTOKEN
KEY_SERVER_URL	Eg. https://mumaa012.in.oracle.com:18443
WEB_URL	Eg. https://mumaa012.in.oracle.com:18443
SERVER_CERTIFICATE_KEY	Refer point 6.7

- c. OAM Setup (Refer to installer pre requisite documents for OAuth configurations)

SERVER_TYPE	OAM
KEY_SERVER_URL	Eg. https://mumaa012.in.oracle.com:18443 (This URL must be of OHS without webgate)
WEB_URL	Eg. https://mumaa012.in.oracle.com:18443
KEY_OAUTH_PROVIDER_URL	http://mum00aon.in.oracle.com:14100/oauth2/rest/token
APP_CLIENT_ID	<Base64 of clientid:secret> of Mobile App client
APP_DOMAIN	OBDXMobileAppDomain
WATCH_CLIENT_ID	<Base64 of clientid:secret> of wearables
WATCH_DOMAIN	OBDXWearDomain
SNAPSHOT_CLIENT_ID	<Base64 of clientid:secret> of snapshot
SNAPSHOT_DOMAIN	OBDXSnapshotDomain
LOGIN_SCOPE	OBDXMobileAppResServer.OBDXLoginScope
SERVER_CERTIFICATE_KEY	Refer point 6.7

d. IDCS Setup

SERVER_TYPE	IDCS
KEY_SERVER_URL	Eg. https://mumaa012.in.oracle.com:18443 (This URL must be of OHS without webgate)
WEB_URL	Eg. https://mumaa012.in.oracle.com:18443
KEY_OAUTH_PROVIDER_URL	http://obdx-tenant01.identity.c9dev0.oc9qadev.com/oauth2/v1/token
APP_CLIENT_ID	<Base64 of clientid:secret> of Mobile App client
WATCH_CLIENT_ID	<Base64 of clientid:secret> of wearables
SNAPSHOT_CLIENT_ID	<Base64 of clientid:secret> of snapshot
LOGIN_SCOPE	obdxLoginScope
OFFLINE_SCOPE	urn:opc:ldm:__myscopes__ offline_access
SERVER_CERTIFICATE_KEY	Refer point 6.7

6. To Enable SSL

There are 2 levels of SSL checks added in the app. One is to check SSL on app launch only and another one is to check SSL for every api calls in UI.

By default app launch SSL is enabled & UI SSL check is disabled. Bank can enable/disable SSL by using below properties.

ENABLE_SSL	true
ENABLE_SSL_FOR_UI	false

7. Enable/Disable Face biometric

Below flag is use to enable or disable Face biometric for alternate login in OBDX app.

ALLOW_FACE_BIOMETRIC	true
----------------------	------

By default product support both biometric type i.e. Face & Fingerprint for alternate login.

Note: Face biometric option is shown by google's biometric api. It may possible that if your device has face registered, but in OBDX app its not showing up while registering biometric alternate login. This is because of sensors of your device is not compatible with the google's biometric api.

If you want to reset alternate login on Add/Remove of fingerprint in your device then disable this flag.

8. Domain Based Setup (This is same for OBDX servicing App and Authenticator App)

To use domain based setup, enable below flag in app.properties file -

```
<string name="DOMAIN_BASED_CATEGORIZATION">true</string>
```

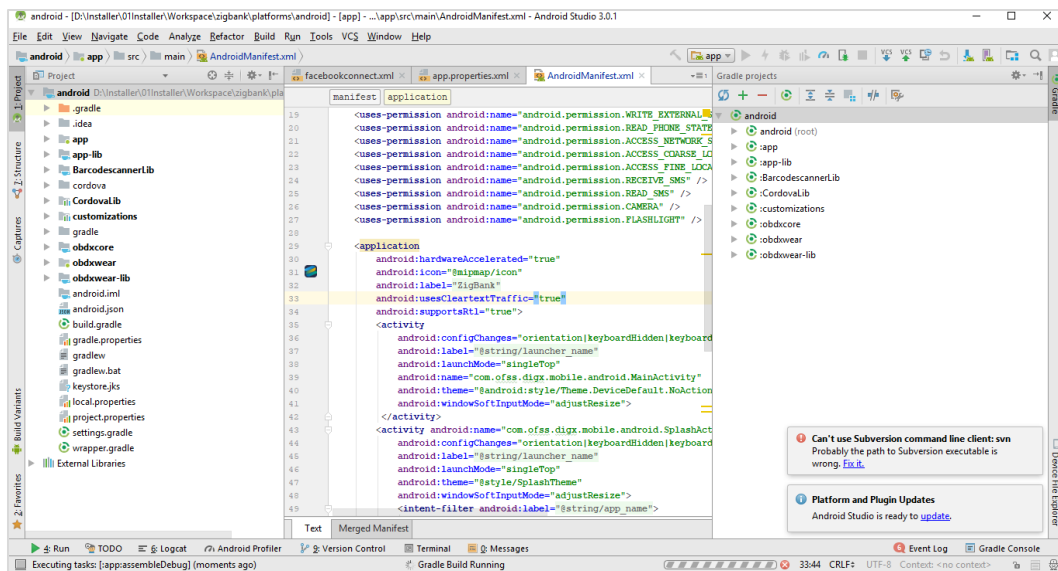
If you are using local UI then enable below flag in config.js(platforms/android/app/src/main/assets/www/framework/js/configurations/config.js) file -

```
domainDeployment: {  
  
    enabled: true  
  
}
```

9. Adding chatbot support to mobile application (Optional)

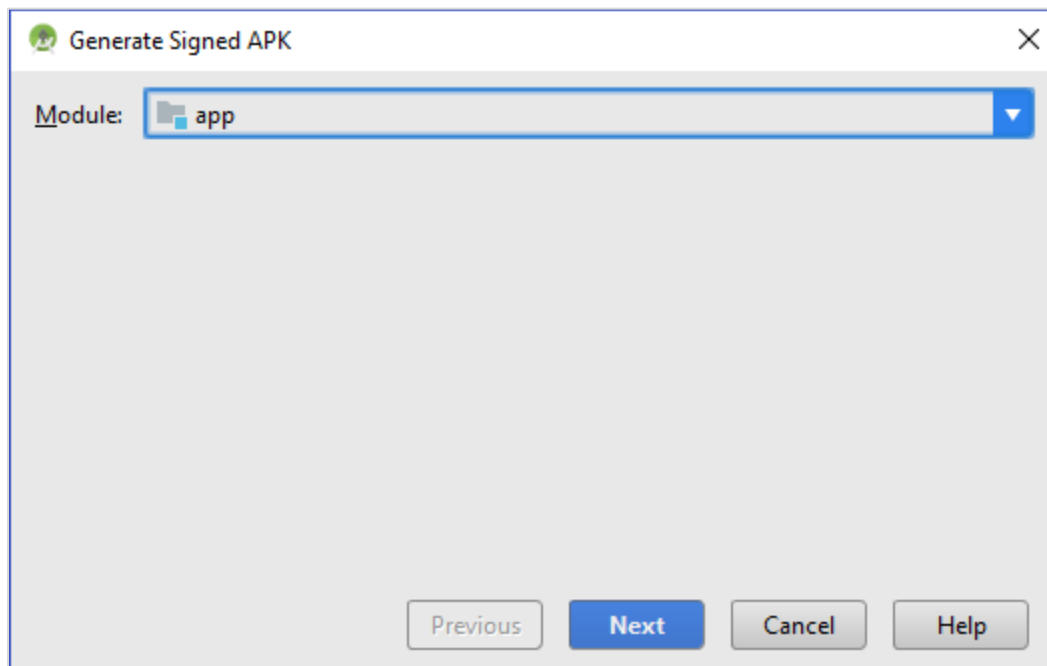
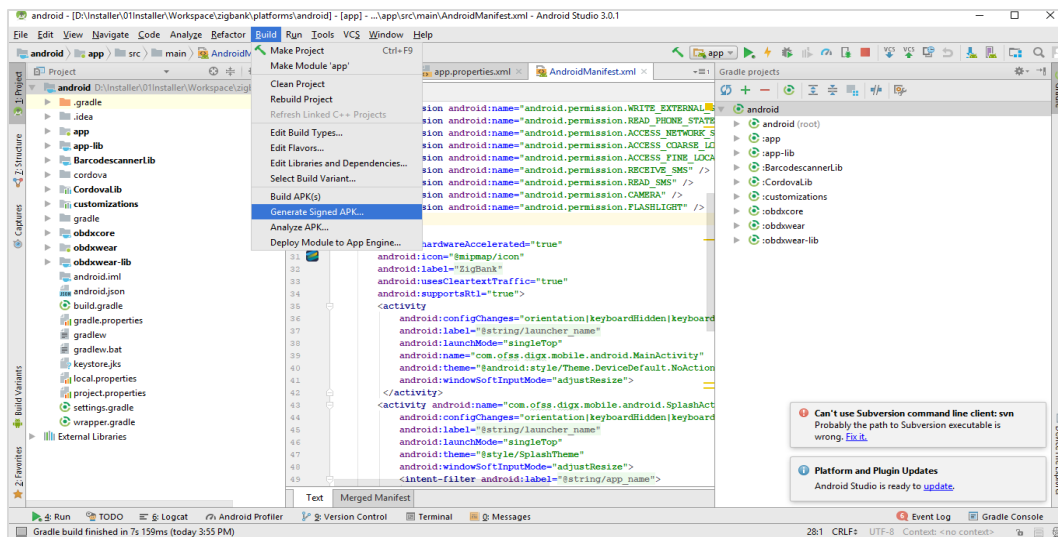
CHATBOT_ID	The tenant ID
CHATBOT_URL	The URL for the ChatApp application in ODA

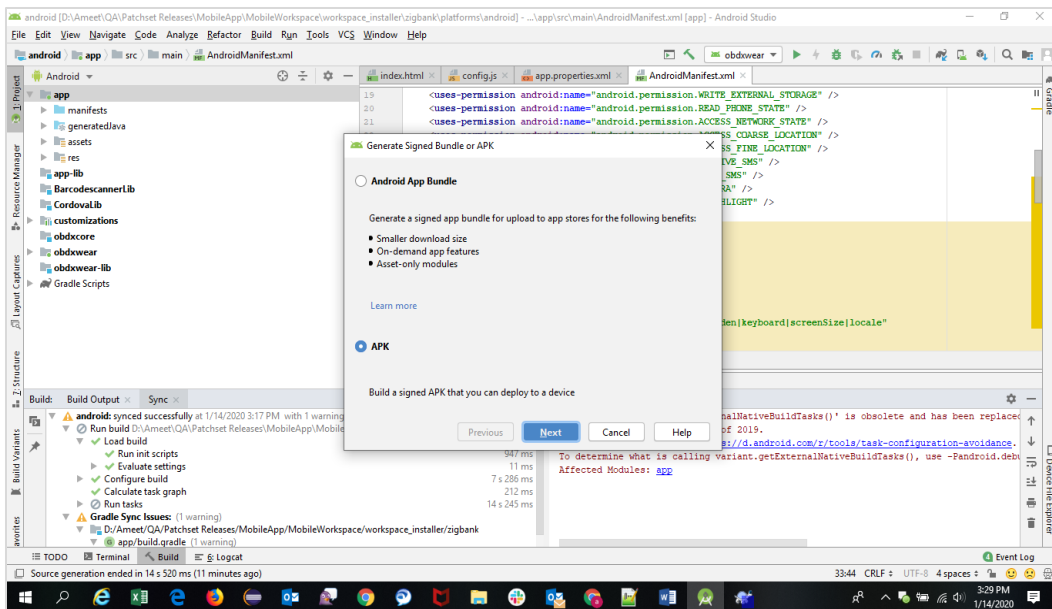
10. If using http protocol for development add (android:usesCleartextTraffic="true") to application tag of AndroidManifest.xml (on app & obdxwear target)



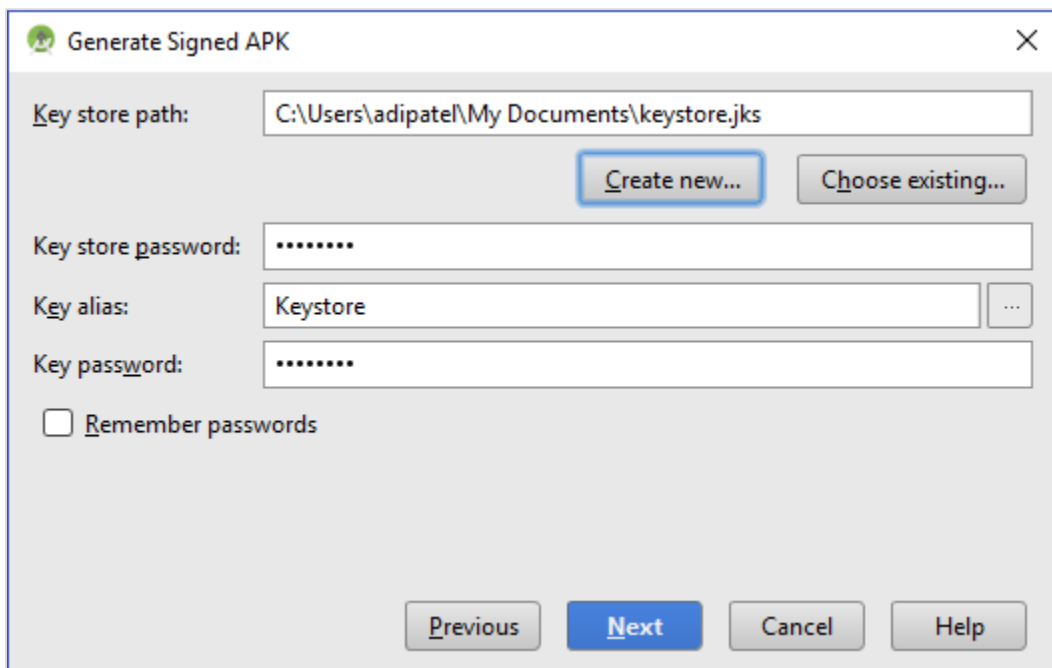
11. For Generating Signed Apk: To Generate release-signed apk as follows:

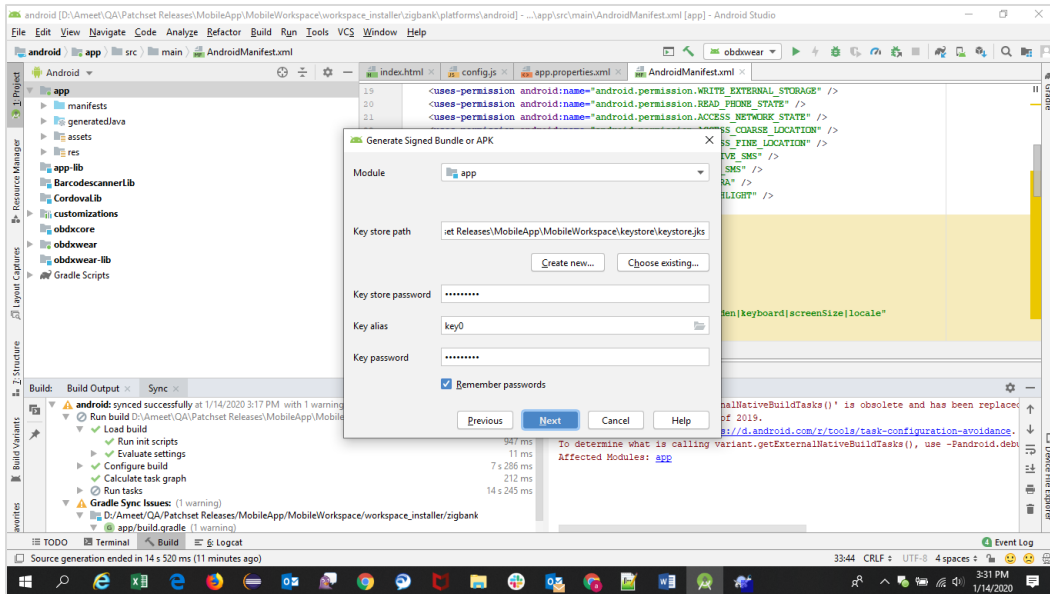
On menu bar click on Build -> Generate Signed Apk



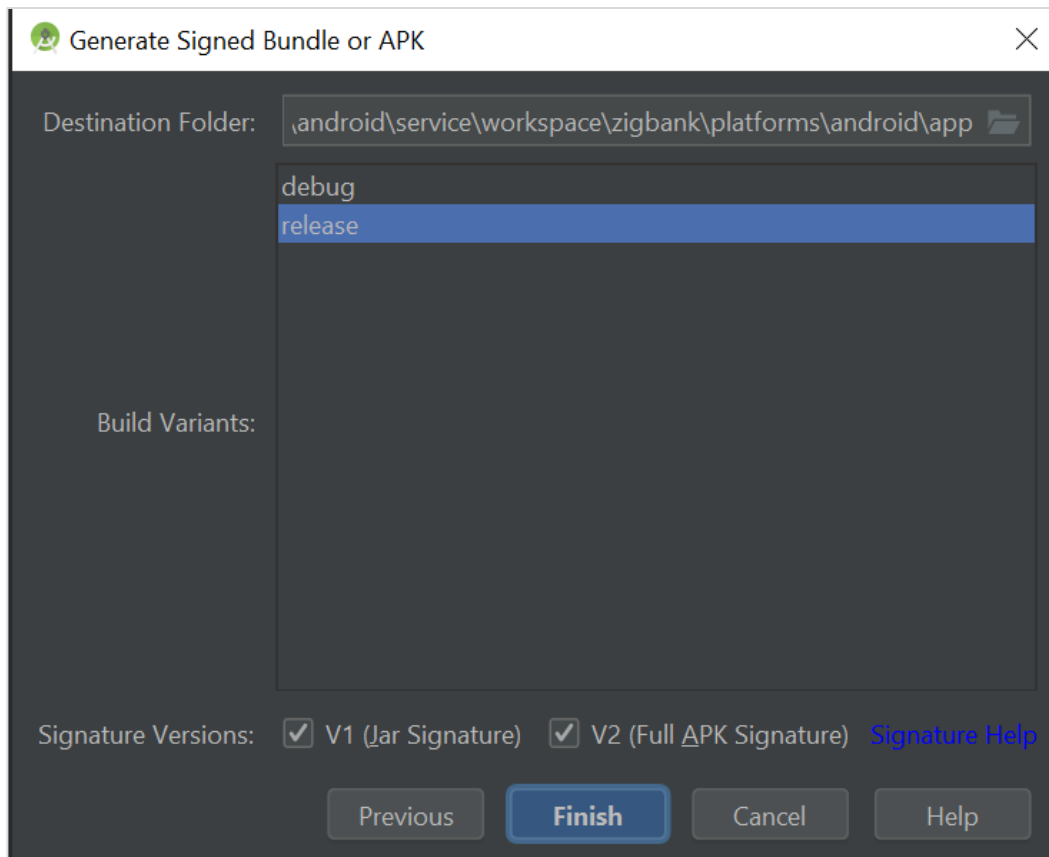


12. If you have an existing keystore.jks file then select choose Existing else click on Create New





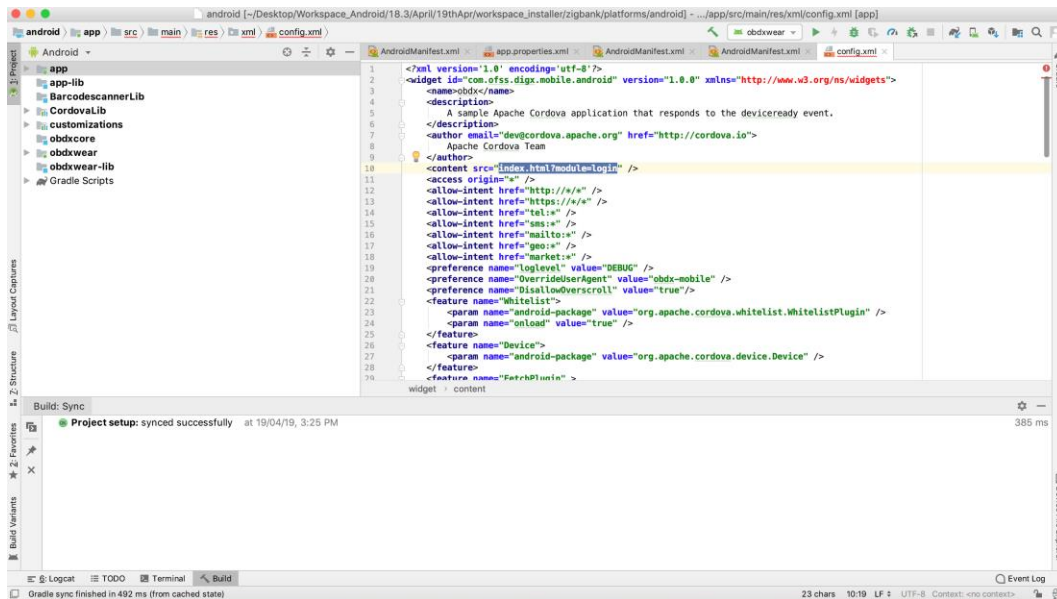
13. Select **Build Type** as **Release**, **Signature Version** as **V1(JAR Signature)** and **V2(Full APK Signature)** and Change APK Destination folder if you want and click on Finish



14. This will generate APK by the given name and destination folder. Default APK Destination folder is **zigbank\platforms\android\app\release**
15. Run the App and select Device or Simulator.

16. Repeat same steps (From step 8 and obdxwear as module) for OBDX Wear App for Release Signing. Use proguard-rules.pro from workspace_installer\zigbank\platforms\android\obdxwear using explorer. The select obdxwear as the module and follow same signing steps with same keystore.
17. The application has a config page at launch to enter the URL of the server (for development only). To remove this page, update the config.xml as shown below

The application has config page to add URL. This is for development purpose only and can be removed using below step. (Update content src tag)



18. Application will work on https only, there is no support for http url further.
19. To enable App widget, enable below flag in app.properties file:
20. <bool name="ENABLE_WIDGET">true</bool>Maintenance page configs-

Enable below flag to show maintenance page when server is under maintenance

```
<string name="SHOW_MAINTENANCE_PAGE">true</string>
```

Also add the status code returned when server is under main in below property-

```
<string-array name="MAINTENANCE_PAGE_STATUS_CODE">
```

```
<item>Your Status Code</item>
```

```
</string-array>
```

Note: You can add multiple status code.

21. To disable caching in app, make below flag to false
- <bool name="ENABLE_CACHING">true</bool>
22. To disable ssl pinning in app, make below flag to false

<bool name="ENABLE_SSL">true</bool> in app.properties.

23. To disable ssl pinning for ui in app, make below flag to false

<bool name="ENABLE_SSL_FOR_UI ">true</bool> in app.properties.

6. OBDX Authenticator Application

1. This is an Authenticator Application which is used when bank has enabled Soft Token Authentication as Authentication mechanism for any transaction. This application basically supports one of below authentication:
 - HOTP: Random based Soft Token
 - TOTP: Time based Soft Token
2. Users should have this application installed and logged in and PIN is set before initiating any transaction which needs this token.
3. Based on the configuration set, user can any time log in with PIN and check the token and use that token for completing any transaction based on "Soft Token Authentication"

6.1 Authenticator UI (Follow any one step below)

6.1.1 Using built UI

For TOKEN-BASED - Unzip dist.tar.gz directory from OBDX_Patch_Mobile\authenticator\TOKEN-BASED

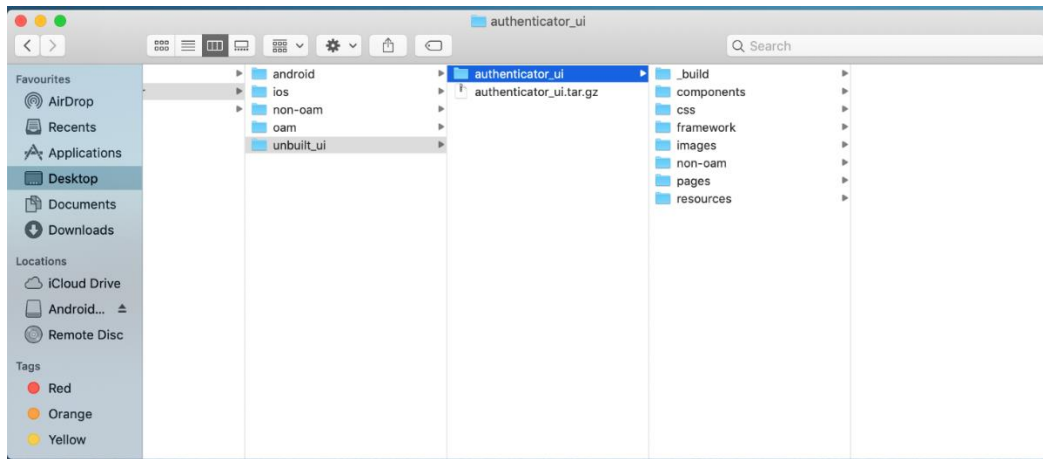
6.1.2 Using Un-built UI

1. Extract authenticator_ui.tar.gz from OBDX_Patch_Mobile\authenticator\unbuilt_ui. Copy the "*token-based/login*" folder and replace it at the "components/modules/" location. This will replace the existing the login folder.
2. Copy the contents except _build folder to Authenticator workspace->platform/ios/www folder

6.1.3 Building UI manually

Extract authenticator_ui.tar.gz from OBDX_Patch_Mobile\authenticator\unbuilt_ui.

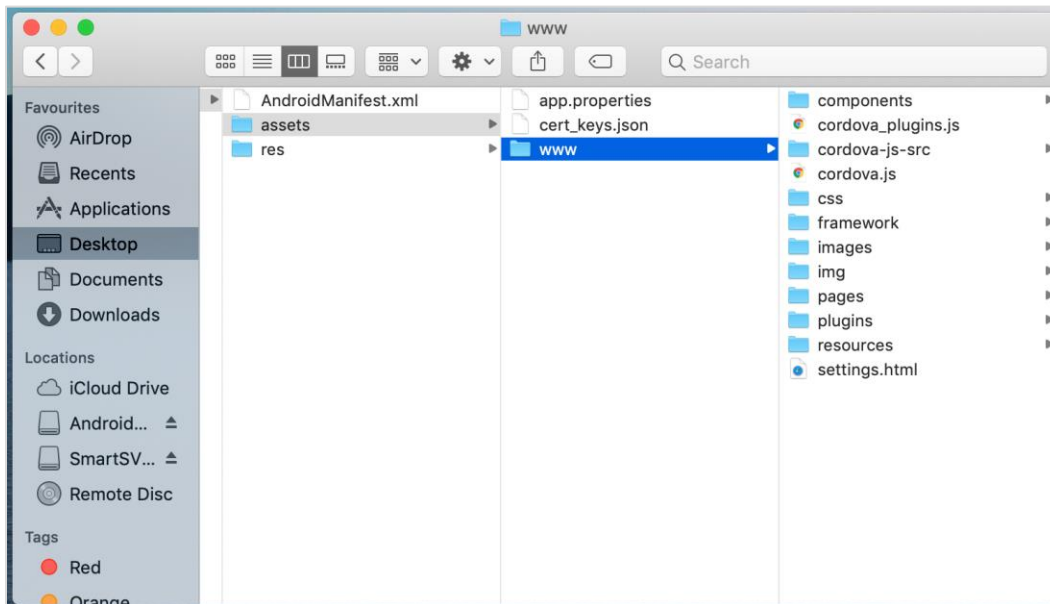
The folder structure is as shown:



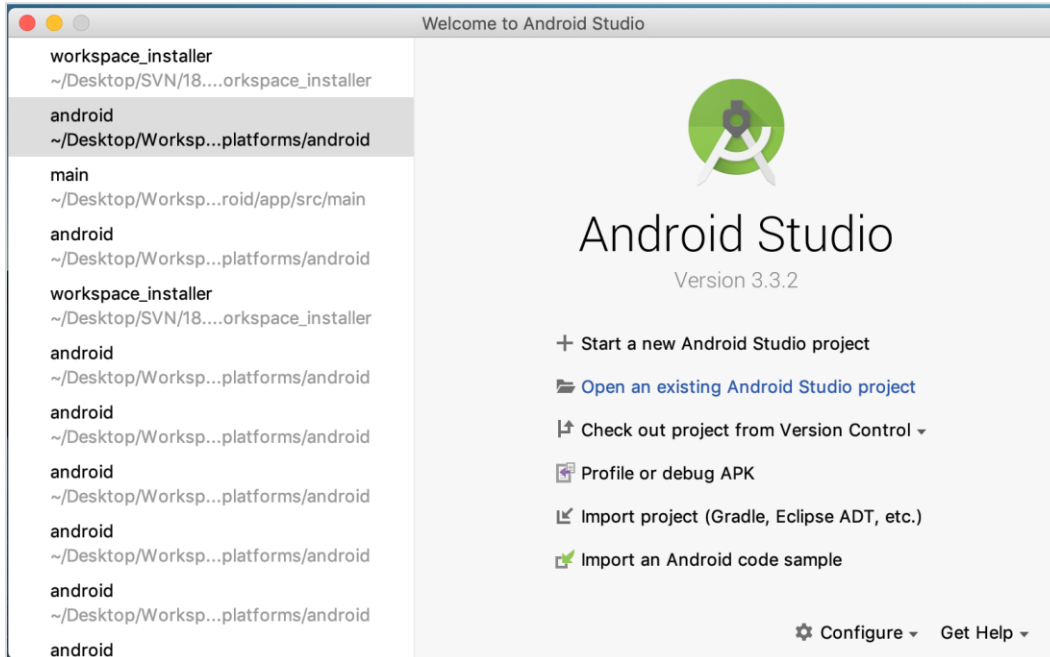
6.2 Authenticator Application Workspace Setup

1. Copy UI (Directories – components, css, framework, images, pages, resources) from /dist directory to workspace/installer/app/src/main/assets/www/

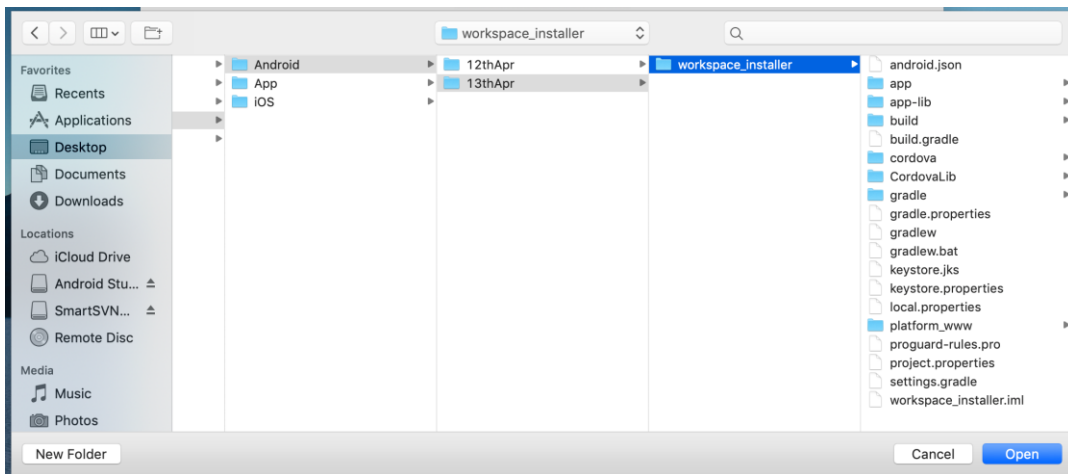
In case any popup appears, click replace



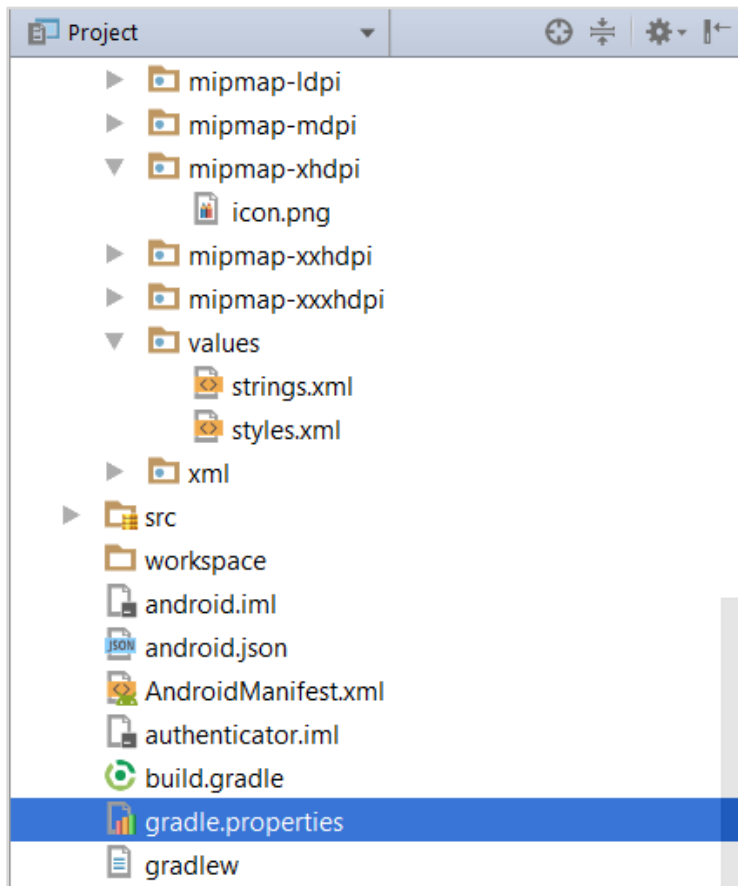
2. Launch Android Studio and open existing project



3. Open OBDX_Installer/workspace_installer folder in Android Studio.



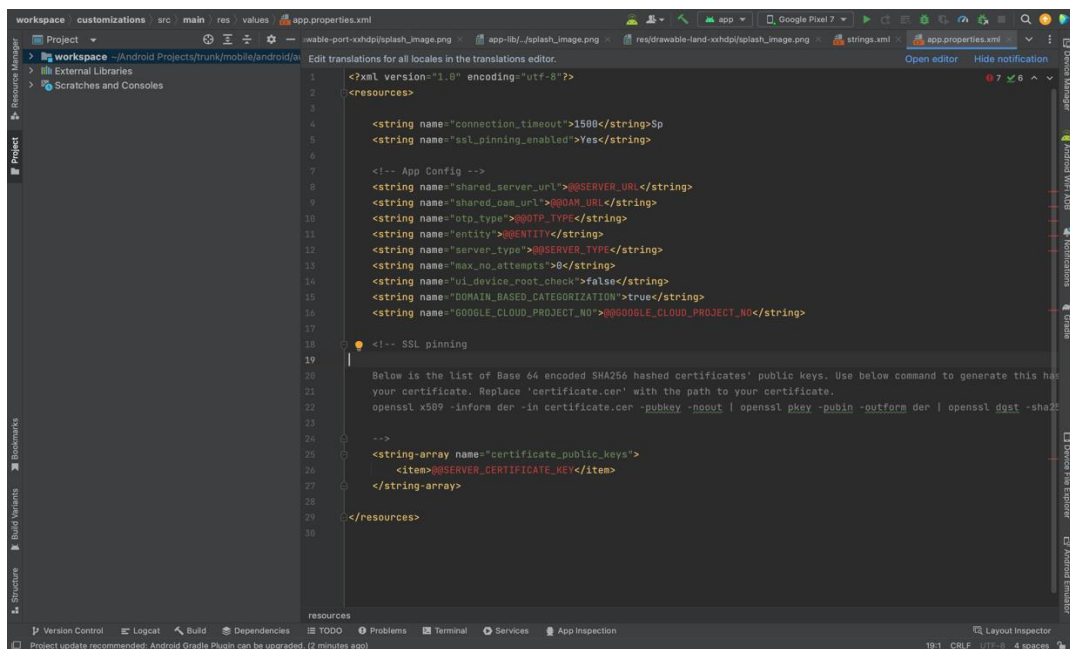
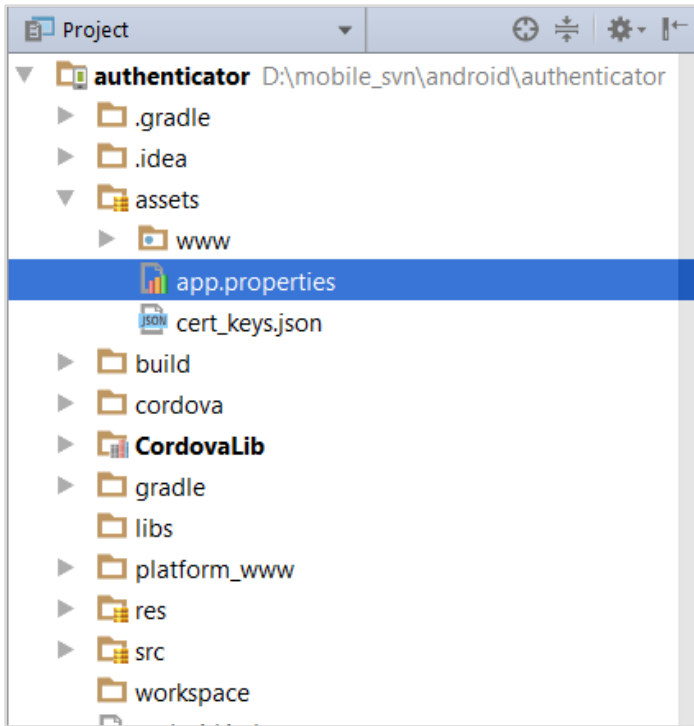
4. Open gradle.properties file and update following properties with relevant proxy address if required



```
systemProp.http.proxyHost = <proxy_address>
```

```
systemProp.https.proxyPort = <port_number>
```

5. Open “assets\app.properties” file and update following properties as per requirement



Set OTP type to HOTP/TOTP as per requirement.

Set Server Type to OBDXTOKEN

Set MAX No Attempts greater than 0

Set UI Device root check to true if you want to add check on login button.

Note: If selected authentication mechanism is not OAM based then remove "*shared_oam_url*" property.

6. Click Build → Clean & Build → Rebuild project in Android Studio.

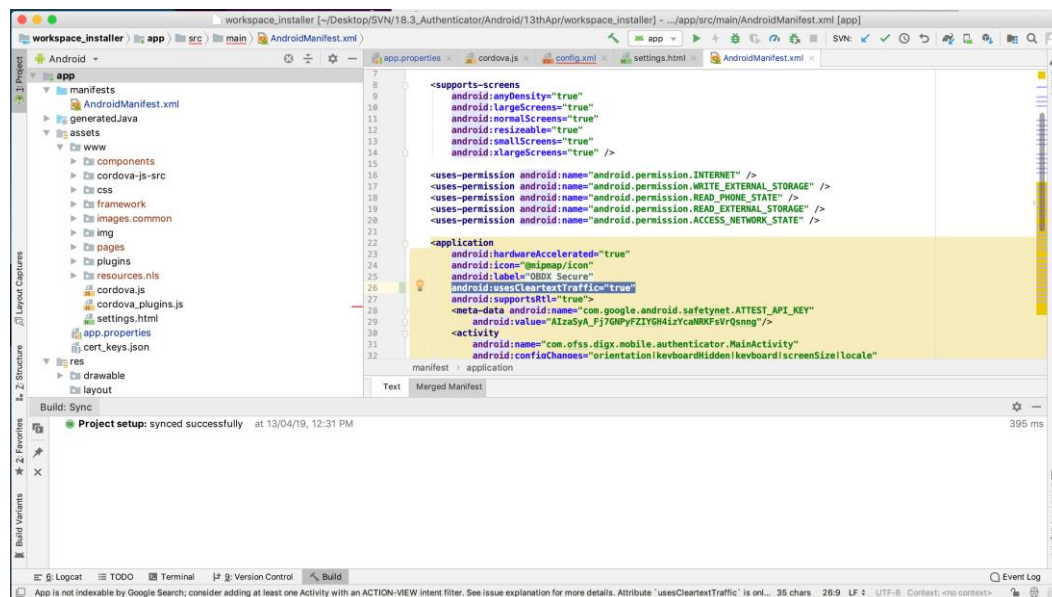
7. Click on Build → Edit Build Type → app → release

Enable minify → true

Add proguard file from workspace_installer/proguard-rules.pro

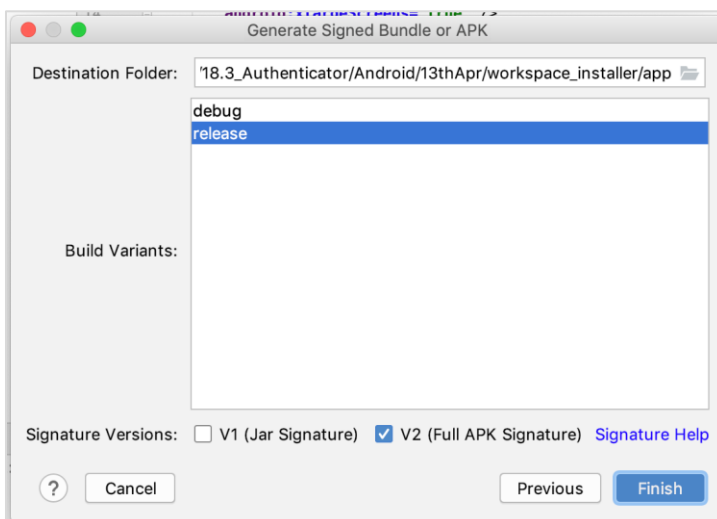
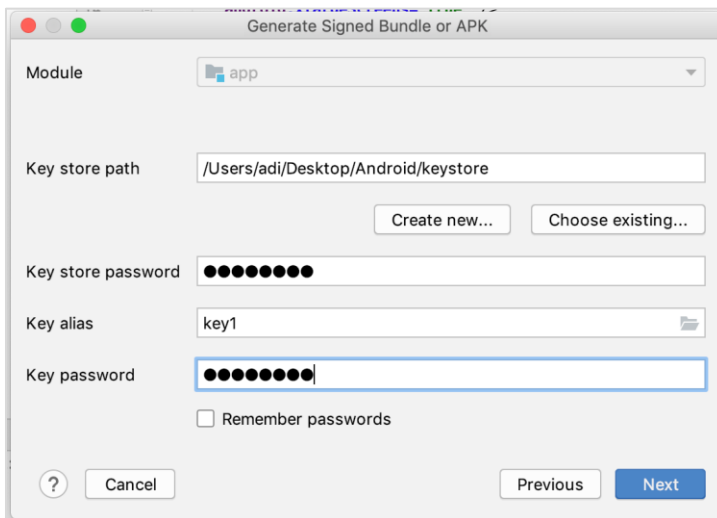
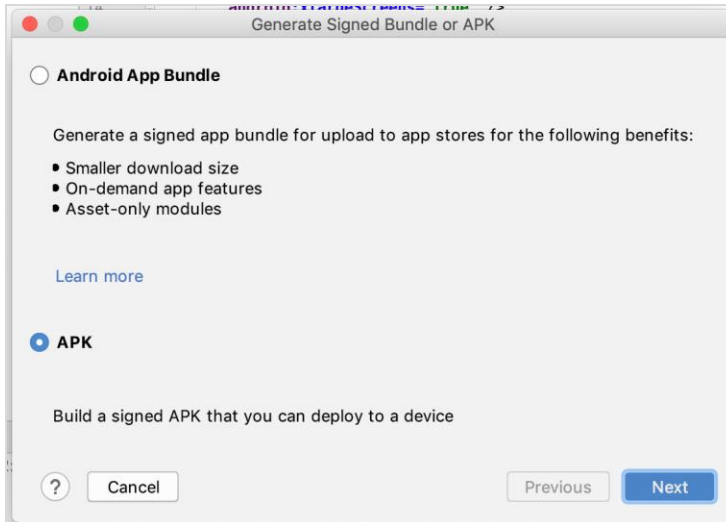
Click OK

8. If using http protocol for development add (android:usesCleartextTraffic="true") to application tag of AndroidManifest.xml



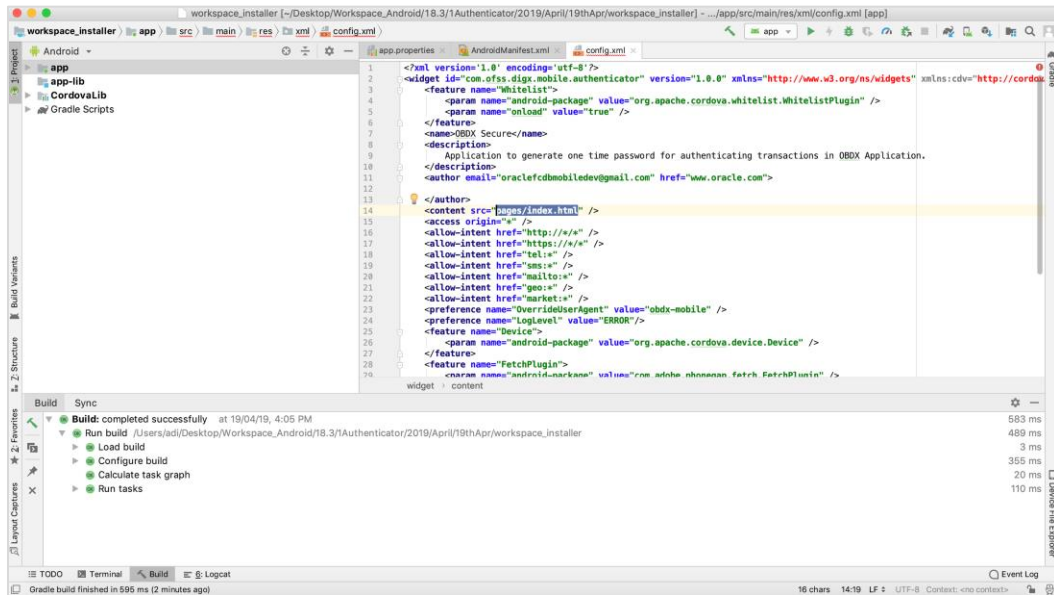
9. **For Generating Signed Apk:** To Generate release-signed apk as follows:

10. On menu bar click on Build -> Generate Signed Apk



Click Finish to generate .apk

The application has config page to add URL. This is for development purpose only and can be removed using below step. (Update content src tag)



7. Application Security Configuration

Root Check → Ensure Step 3 is completed.

1. We also have to maintain package names of Servicing and Authenticator app in the same table, i.e. **DIGX_FW_CONFIG_ALL_B** corresponding to the following keys respectively:

ANDROID_SERVICING_PACKAGE and **ANDROID_AUTHENTICATOR_PACKAGE**

An example query will be:

```
insert into digx_fw_config_all_b (PROP_ID, CATEGORY_ID, PROP_VALUE,
FACTORY_SHIPPED_FLAG, PROP_COMMENTS, SUMMARY_TEXT, CREATED_BY,
CREATION_DATE, LAST_UPDATED_BY, LAST_UPDATED_DATE, OBJECT_STATUS,
OBJECT_VERSION_NUMBER) values ('ANDROID_SERVICING_PACKAGE',
'mobileconfig', 'com.ofss.zigbank', 'N', '', 'Stores device id in OUD', 'ofssuser', sysdate,
'ofssuser', sysdate, 'Y', 1,);
```

SSL Pinning

2. Get the list of Base 64 encoded SHA256 hashed certificates' public keys of server's valid certificates. Use below command to generate this hash for your certificate. Replace '<certificate.der>' with the path to your certificate.

```
openssl x509 -inform der -in <certificate.der> -pubkey -noout | openssl pkey -pubin -outform der | openssl dgst -sha256 -binary | openssl enc -base64
```

3. Add the hashed keys generated in point 6 to **zigbank\platforms\android\customizations\src\main\res\values\app.properties.xml** file in 'certificate_public_keys' array. Append this key to 'sha256/' in an <item> tag as shown below. Multiple certificate keys can be added to 'certificate_public_keys' array by adding them in <item> tags.

Eg.:

```
<string-array name="certificate_public_keys">
  <item>sha256/5kJvNEMw0KjrCAu7eXY5HZdvyCS13BbA0VJG1RSP91w=</item>
</string-array>
```

Eg. for multiple certificates (In case OAM/IDCS is used):

```
<string-array name="certificate_public_keys">
  <item>sha256/5kJvNEMw0KjrCAu7eXY5HZdvyCS13BbA0VJG1RSP91w=</item>

  <item>sha256/3rgsgghoqrDegekpkkgk92Fgw1w7exyYCS1okef9Oo1w=</item>
</string-array>
```

8. Adding Custom Cordova Plugin

Step 1 -

Create java folder and add your package under app(zigbank\platforms\android\app)

Create java file under your package which will extend CordovaPlugin

Override execute method with JSONArray as a parameter

Retrieve JSONObject from JSONArray and get the data which is passed from js file

Example:

```
public class GetDirectionMapPlugin extends CordovaPlugin {

    @Override

    public boolean execute(String action, JSONArray args, CallbackContext callbackContext)

        throws JSONException {

        try{

            JSONObject object = args.getJSONObject(0);

            String yourKey = object.getString("your_key");

        }catch (Exception e){

            Log.e(TAG,e.getMessage());

        }

        return true;

    }

}
```

Step 2 –

Create plugin file under plugins folder of

www(zigbank\platforms\android\service\workspace\app\src\main\assets\www\plugins)

Example:

```
cordova.define("cordova-plugin-getdirection", function(require, exports, module) {

    var exec = cordova.require('cordova/exec');
```

```

exports.navigate = function(args, successCallback, errorCallback) {
    cordova.exec(successCallback, errorCallback, "GetDirectionMapPlugin", "direction",
        [args]);
};
});

```

cordova-plugin-getdirection.GetDirectionPlugin -> user defined id from
 cordova_plugin.js(zigbank\platforms\android\service\workspace\app\src\main\assets\ww
 w\cordova_plugin.js)

GetDirectionMapPlugin-> name of java plugin class

direction -> action

navigate -> this can be use in js file to this function

Step 3 –

Make entry of plugin in
 cordova_plugin.js(zigbank\platforms\android\service\workspace\zigbank\platforms\android\app\sr
 c\main\assets\www) as below ->

Example:

```

{
    "id": "cordova-plugin-getdirection.GetDirectionPlugin", -> user defined id
    "file": "plugins/cordova-plugin-getdirection/www/mapgetdirection.js", -> path of plugin js
    file
    "pluginId": "cordova-plugin-getdirection",
    "clobbers": [
        "window.GetDirection" -> this can be used in js file to call plugin
    ]
}

```

Step 4 -

Make entry of java plugin class in
config.xml(zigbank\platforms\android\service\workspace\zigbank\platforms\android\app\src\main\res\xml) file of app as below -

Example:

```
<feature name="GetDirectionMapPlugin">  
<param name="android-package" value="Your_Plugin_Java_Class_Path" />  
</feature>
```

GetDirectionMapPlugin -> Name of java plugin class

Step 5 -

Plugin calling in js file ->

Example:

```
window.getDirection.navigate({  
  originLatLng: origin,  
  destinationLatLng: location  
})
```

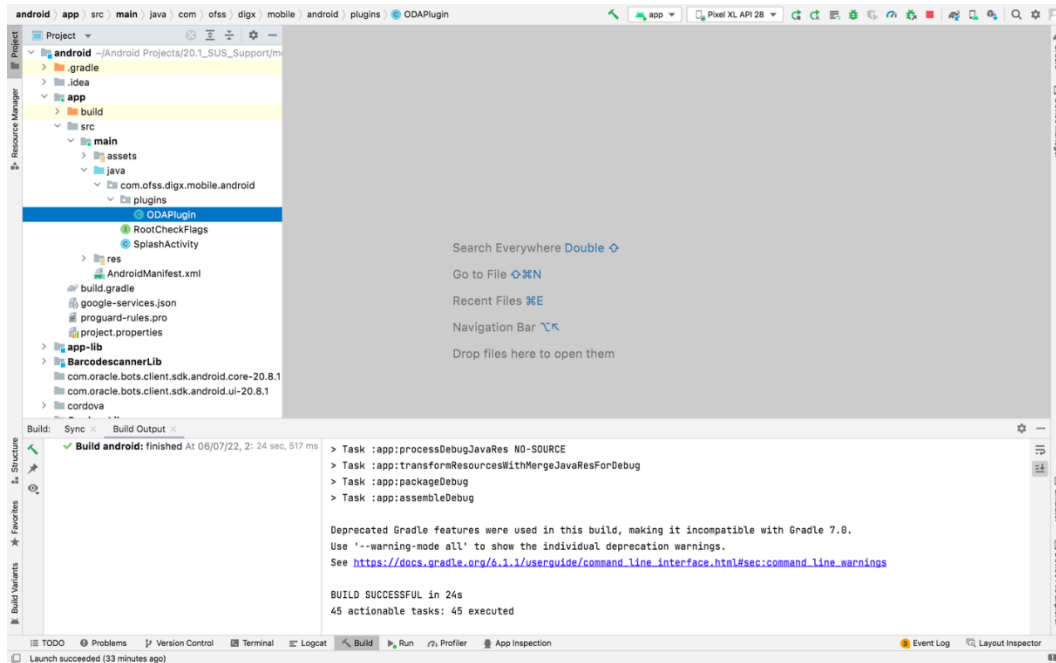
window.getDirection -> clobber define in the cordova_plugin.js file

navigate -> name of the function defined in plugin js file

9. ODA Chatbot Inclusion

To enable ODA Chatbot services in the mobile app, the following changes needs to be made:

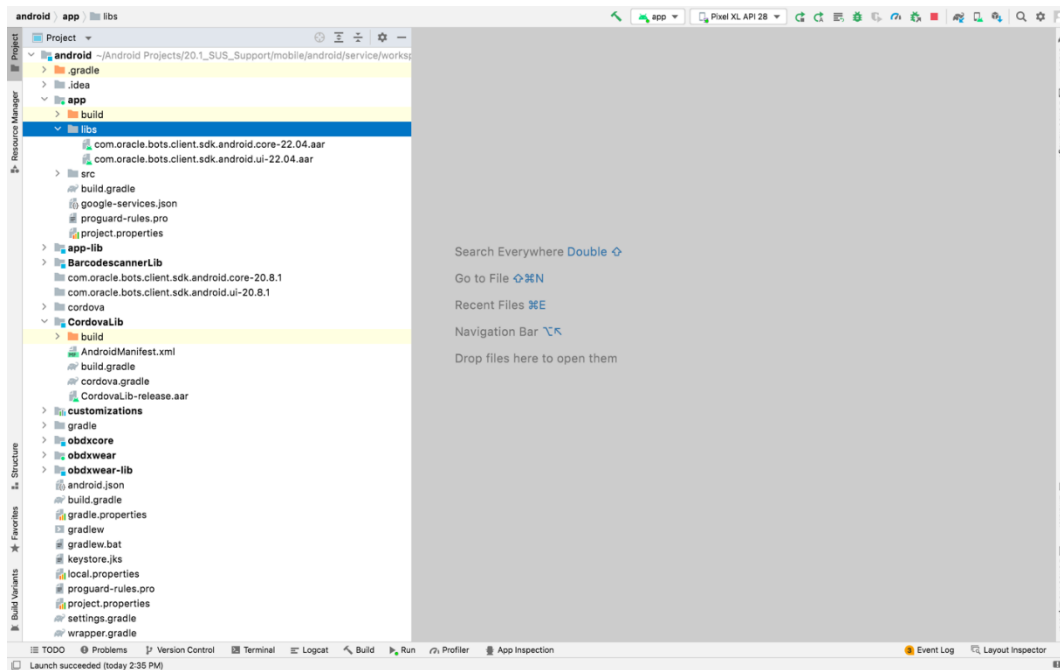
1. Copy ODAPugin.java from workspace_installer/AppExtension/oda to workspace_installer/zigbank/platforms/android/app/src/main/java/com/ofss/digx/mobile/android/plugins/



2. Download ODA Android sdk from below link-

<https://www.oracle.com/downloads/cloud/amce-downloads.html>

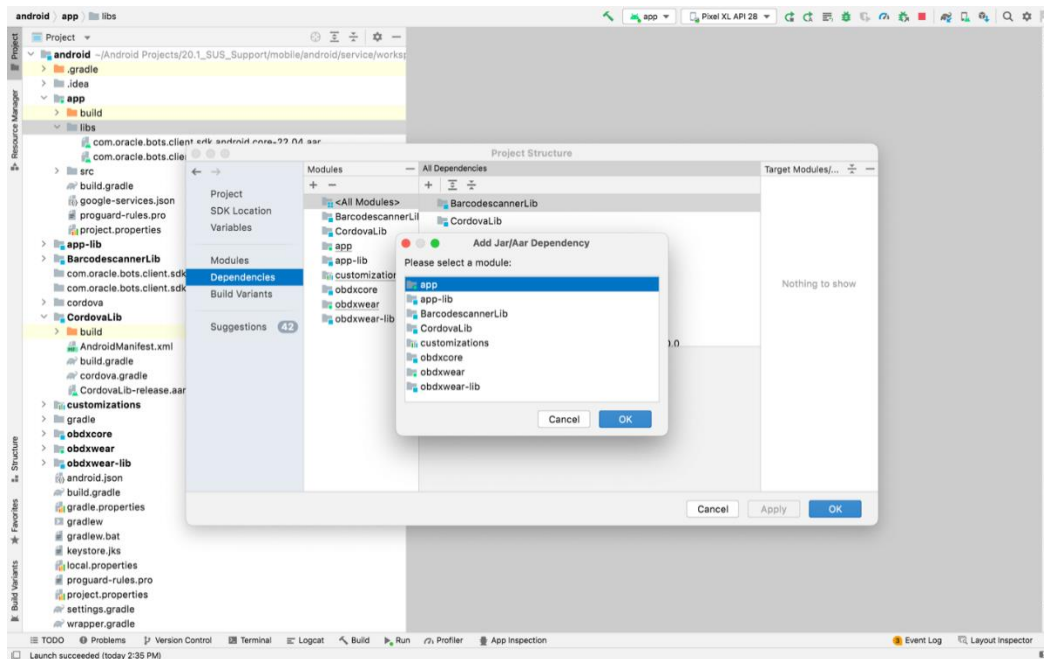
3. Add libs folder at zigbank\platforms\android\app and copy below files from downloaded sdk folder in it.
 - a. com.oracle.bots.client.sdk.android.core-xx.aar
 - b. com.oracle.bots.client.sdk.android.ui-xx.aar



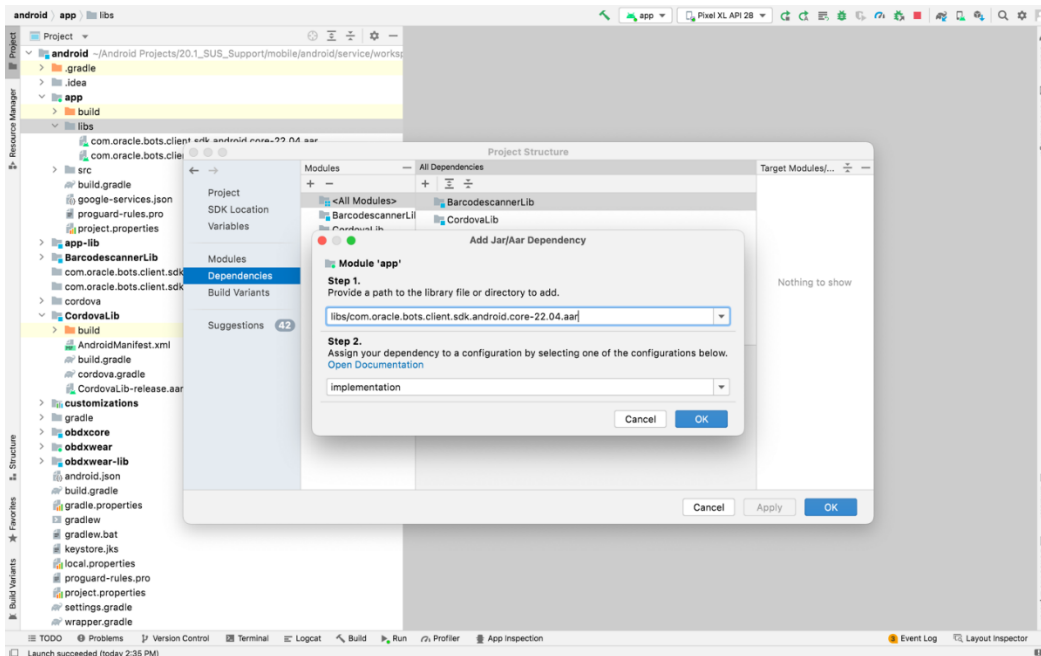
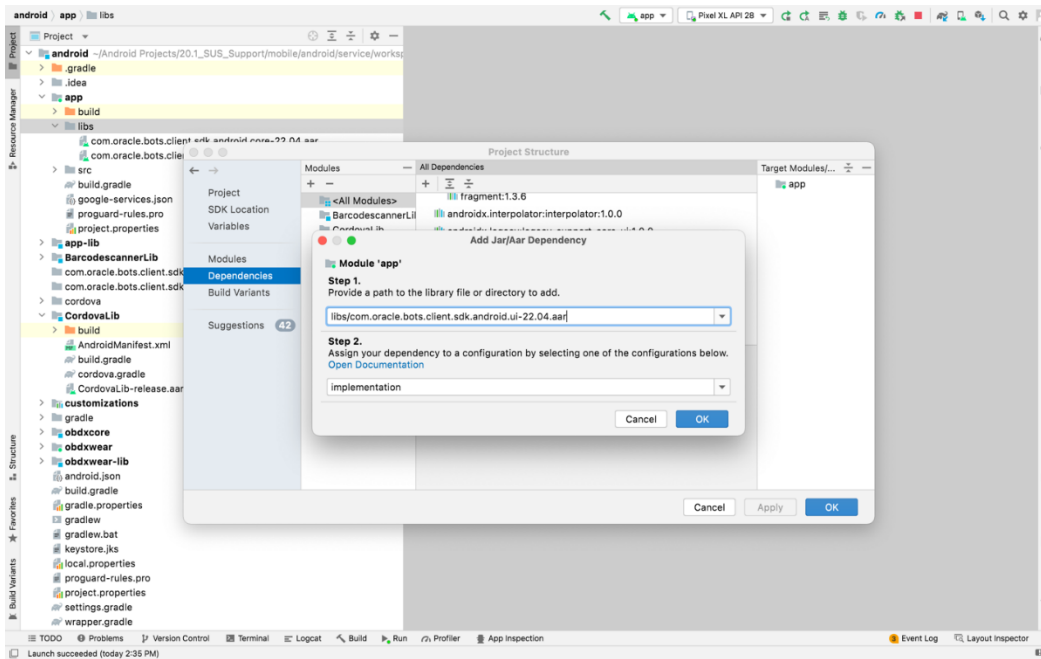
4. In Android Studio follow below steps-

File -> Project Structure -> Dependencies

5. Click on "+" icon and select JR/AAR Dependency and select app module and click Ok.



6. Add both .aar file paths from step3. Then click Apply and Ok.



7. Add Chatbot ID and Chatbot URL in app.properties.xml(zigbank\platforms\android\customizations\src\main\res\values)

```
<string name="CHATBOT_ID">@@CHATBOT_ID</string>
```

```
<string name="CHATBOT_URL">@@CHATBOT_URL</string>
```

10. Push Notification 2FA configuration

If Push notification 2fa is enabled at bank side for any transaction then, the screen displays message to wait for the push notification to accept/reject the transaction authentication. The message as well contains a timer of 5 minutes displayed on the UI. This value is set in the UI code. If bank needs to change this value, bank needs to update the value in UI code:

File path: channel/metadata/user-components/push-out-of-band/push-out-of-band/hook.js

Code to be changed: const mins = <<value>>;

Update the value to what bank needs to set it. This value is in minutes.

So, ideally 5 minutes (existing value in base UI code) is an ideal time. Any changes made in this value should satisfy below pre-condition.

1. There is an OTP expiration time set in “digx_fw_config_ALL_b” table.
2. Also, there is business policy check set to 10 minutes for validation of the generated 2fa token. Bank can write their own business policy where they can modify the 10 minutes time.

So, the time in UI code should not exceed 10 minutes and OTP expiration time in “digx_fw_config_ALL_b” table.