Chatbot Configuration Guide
Oracle Banking Digital Experience
Release 21.1.0.0.0

Part No. F40800-01

May 2021

ORACLE

Chatbot Configuration Guide

May 2021

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/

Table of Contents

1. Preface

1.1 Intended Audience

This document is intended for the following audience:

- Customers
- Partners

1.2 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.

1.3 Access to Oracle Support

Oracle customers 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 Structure

This manual is organized into the following categories:

Preface gives information on the intended audience. It also describes the overall structure of the User Manual.

The subsequent chapters describes following details:

- Introduction
- Preferences & Database
- Configuration / Installation.

1.5 Related Information Sources

For more information on Oracle Banking Digital Experience Release 21.1.0.0.0, refer to the following documents:

- Oracle Banking Digital Experience Installation Manuals

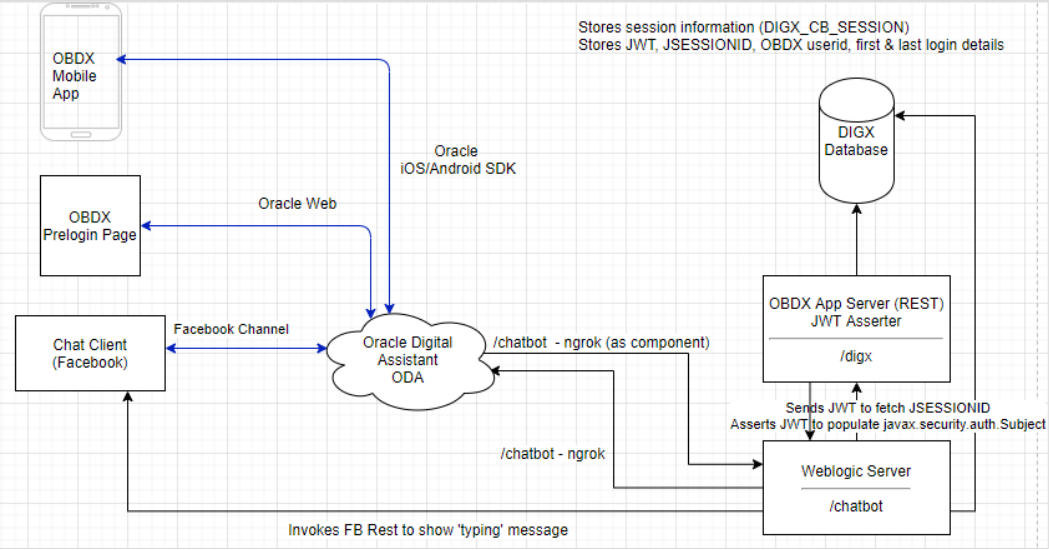
2. Purpose

OBDX provides interface for Chatbot module, integrated with Oracle Digital Assistant (ODA) out of the box. It provides end users a chat interface to interact with the bank. Transactions like balance enquiry, fund transfers to payees, enquiring about banking products and details of ATM/Branches can be achieved through chat. This document provides steps to setup OBDX chatbot module with ODA. The prerequisites include:

- ODA setup
- Facebook credentials (optional)

Home

3. Topology

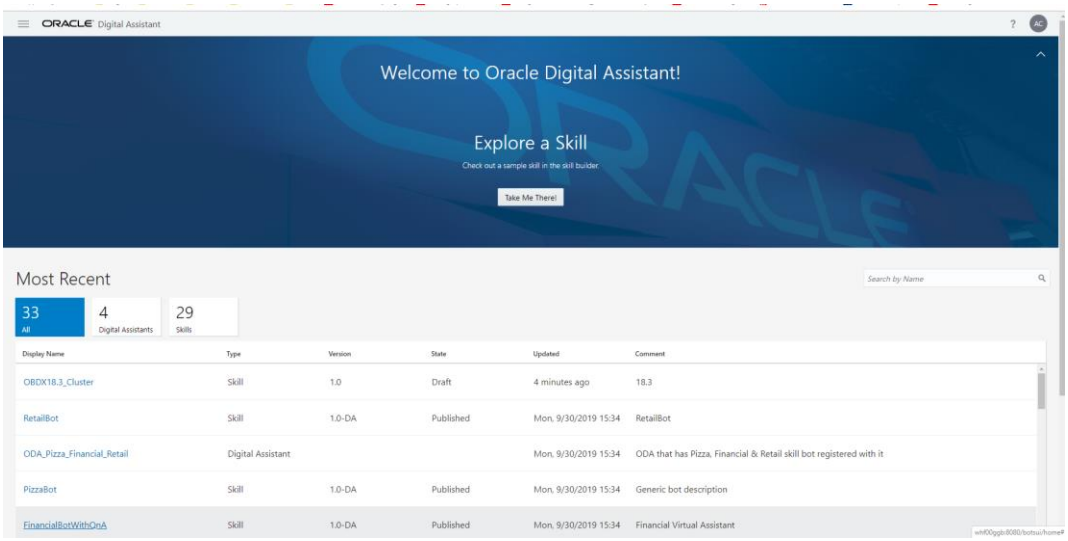


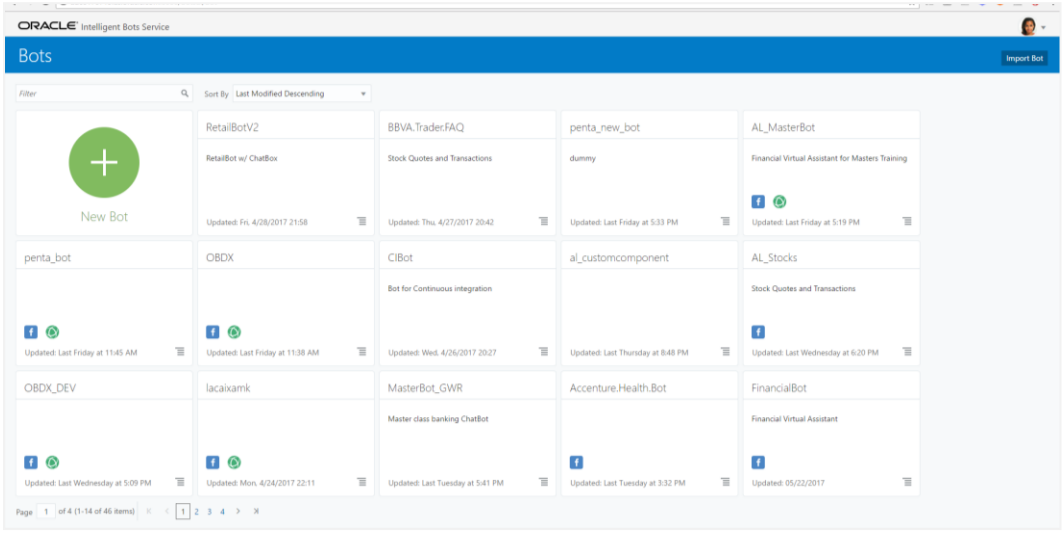
Home

4. Common Configurations

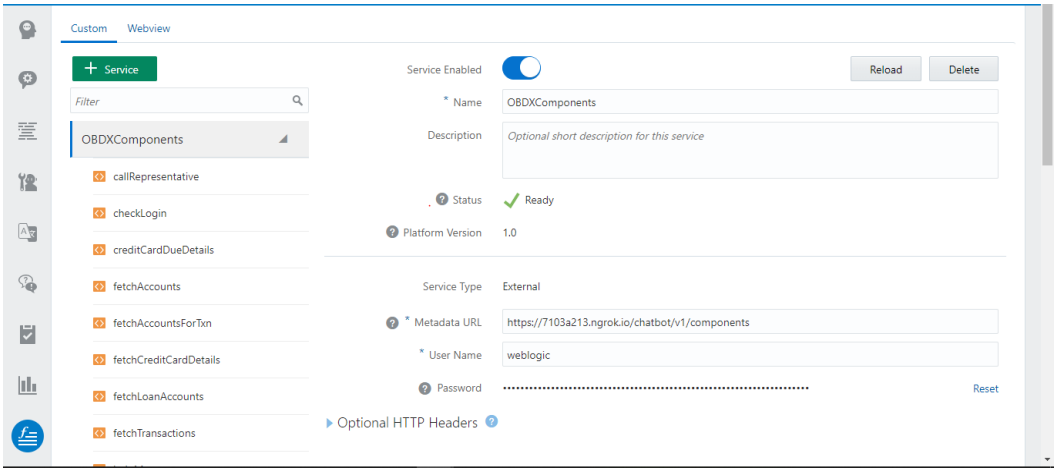
4.1 ODA Configurations

1. Login to ODA and import the OBDX bot shipped with OBDX installer. This is a zip (OBDX201.zip) file obtained in the installer in OBDX_Installer/installables/chatbot/config directory. Import this by clicking the “Import Bot” on ODA dashboard.



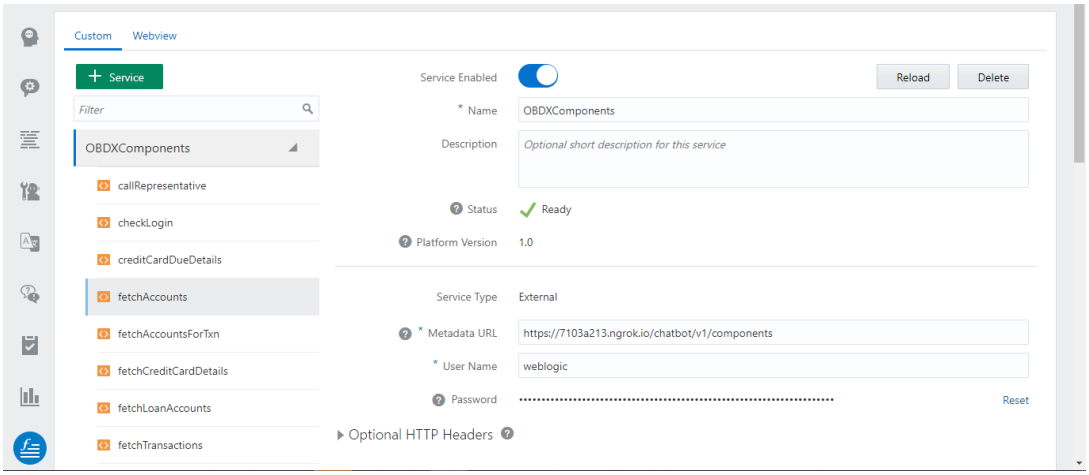


2. Click on the OBDX Bot and click on the components to add the custom components.

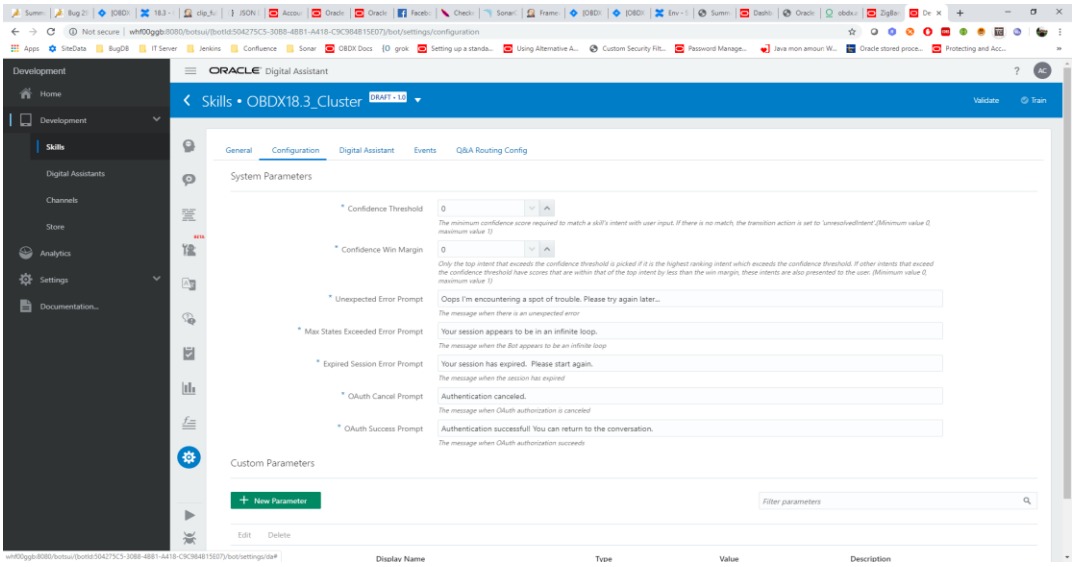


3. Put the OBDX URL here. The OBDX setup and the ODA setup must be accessible over Internet.

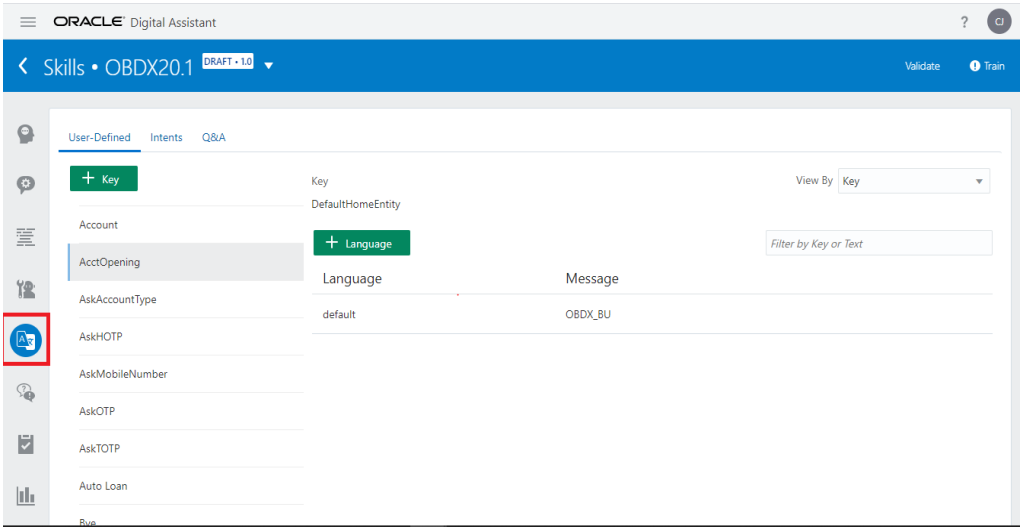
4. Add username/password (in HTTP Basic authorization) of any user with Administrators role which can be used to login in OBDX Weblogic server.



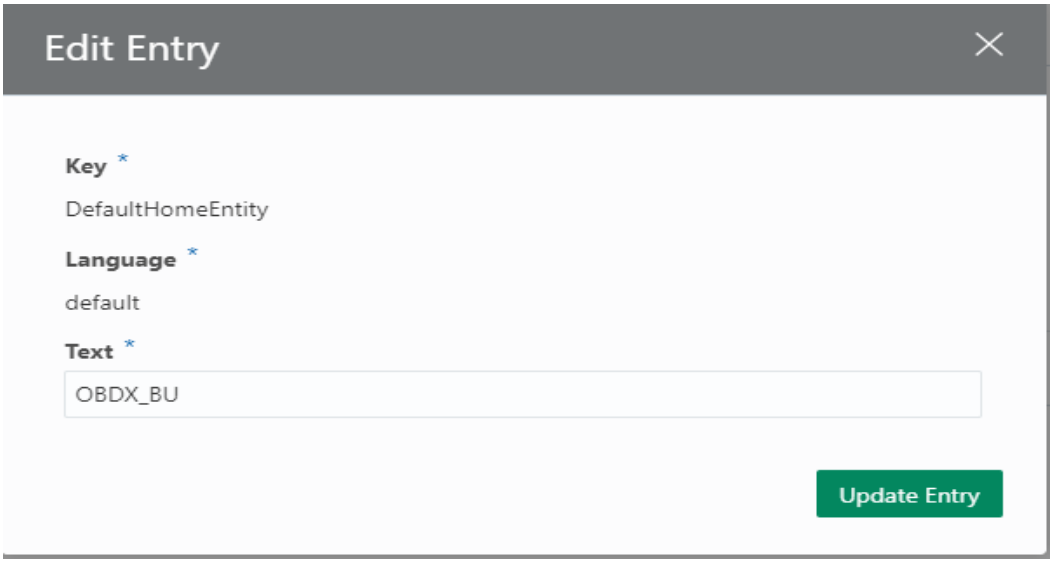
5. In order to configure intent threshold for the skill go to settings tab in a bot and click configurations to configure threshold frequency (default 0.5) as shown below-



6. Configure/add default home entity in a resource bundles (one created/configured in OBDX) on ODA as same will be used as the required home entity while consuming OBDX API's from custom components. Go to Resource Bundles as shown below-



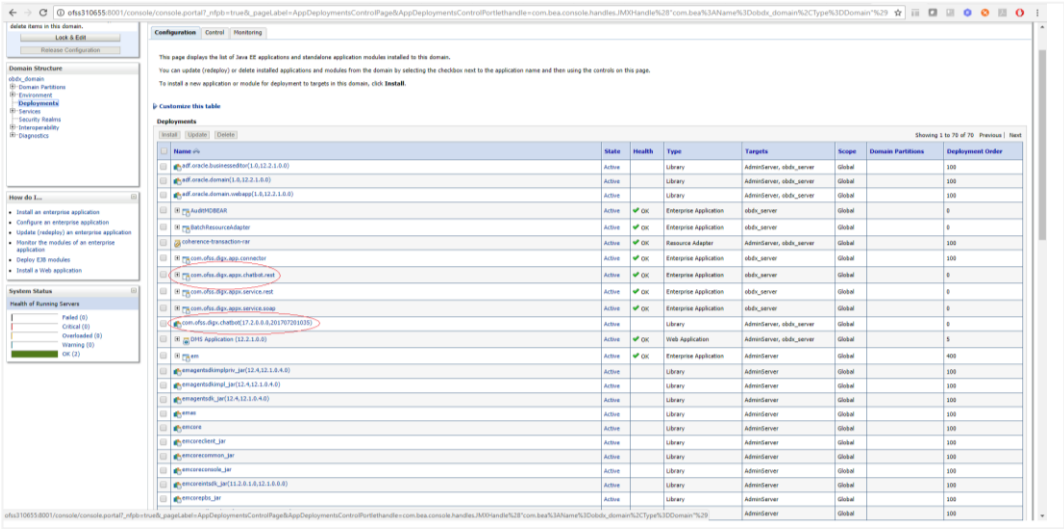
7. Search for key “DefaultHomeEntity”. Default value will be “OBDX_BU”.Chatbot works with single entity only. The default value can be changed by selecting edit option/icon in the key as shown below –



4.2 OBDX Server Configurations

Ensure that below applications are running on OBDX server

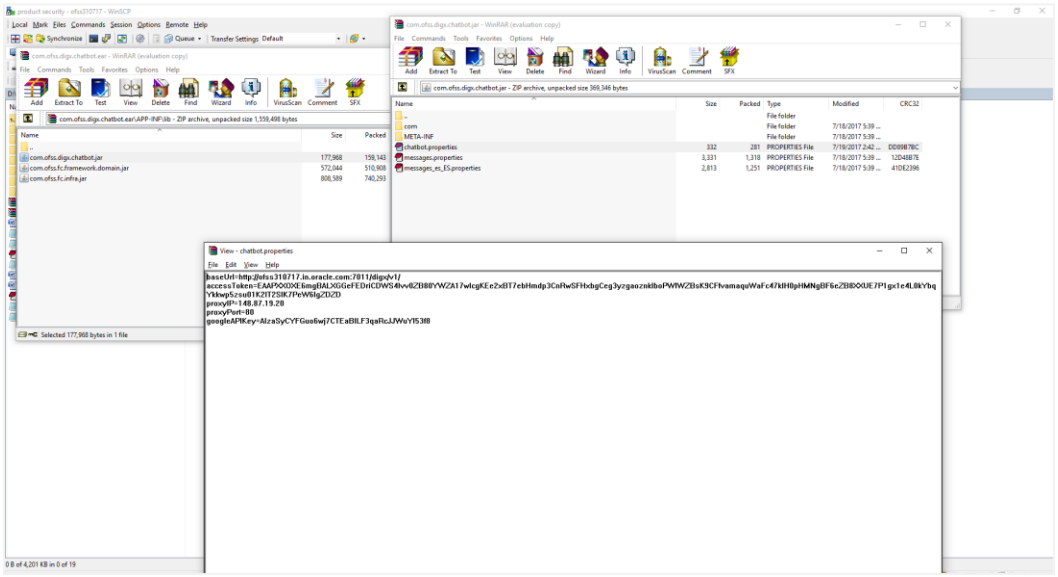
- com.ofss.digx.infra.channel
- com.ofss.digx.chatbot



In chatbot.properties in com.ofss.digx.chatbot.ear > com.ofss.digx.chatbot.jar, enter the base URL of the OBDX server where DIGX application is running.

Note: Enter the Weblogic port. If using OHS, that should not be patched with Webgate

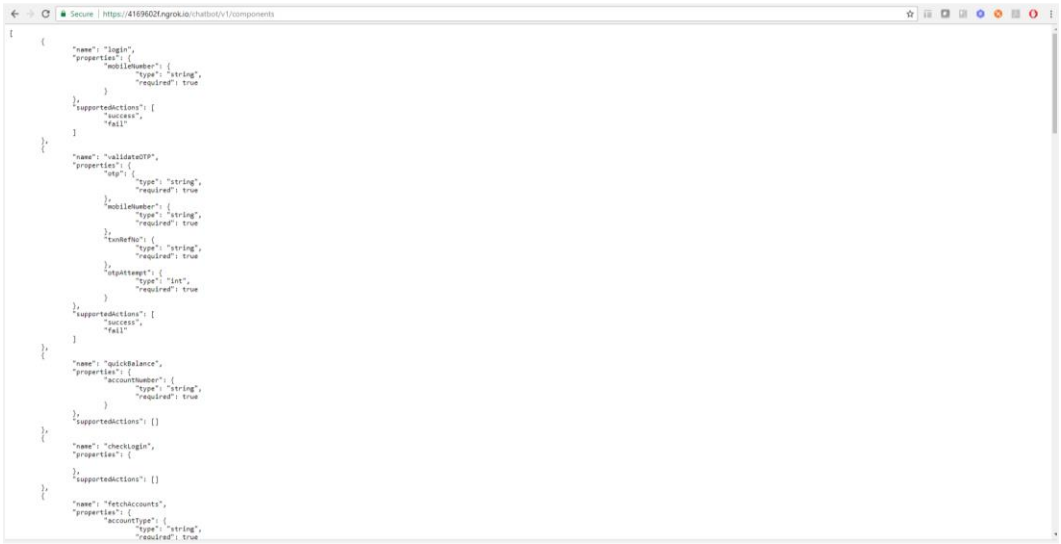
If the server needs proxy to connect to internet, enter proxy details here else leave them blank. This call is required for the chatbot to display the typing.. icon in chat. The connection is directly from OBDX Chatbot application to Facebook. The access token of the Facebook page is also required here.



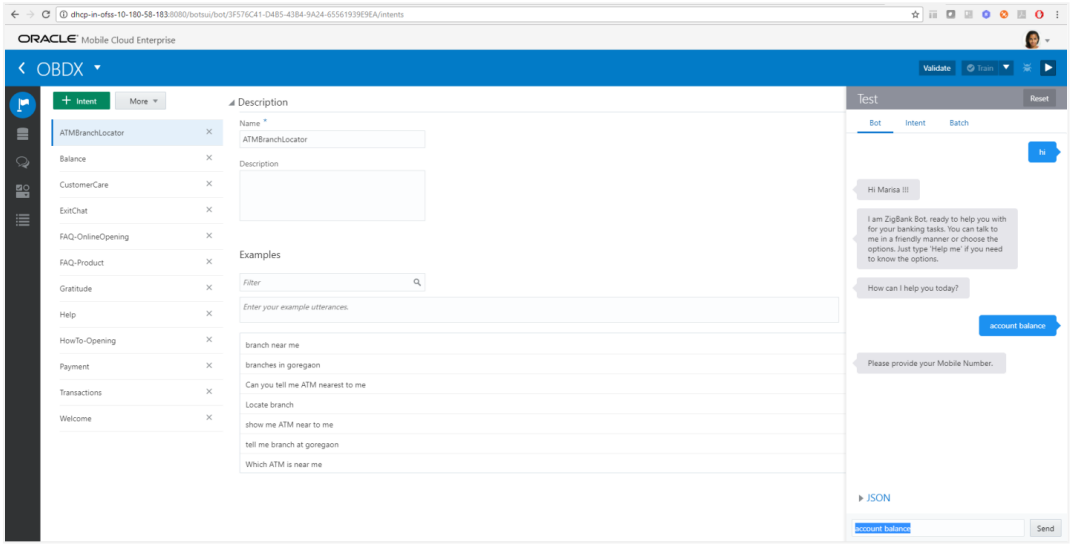
Redeploy the ear after above changes.

Verification Steps

8. Check if OBDX server Chatbot application is running and accessible over the component URL given to ODA. Open a browser and enter the URL as entered in ODA component services. (as configured in Step No-4.1.3)



9. Login to ODA and click OBDXBot > Test

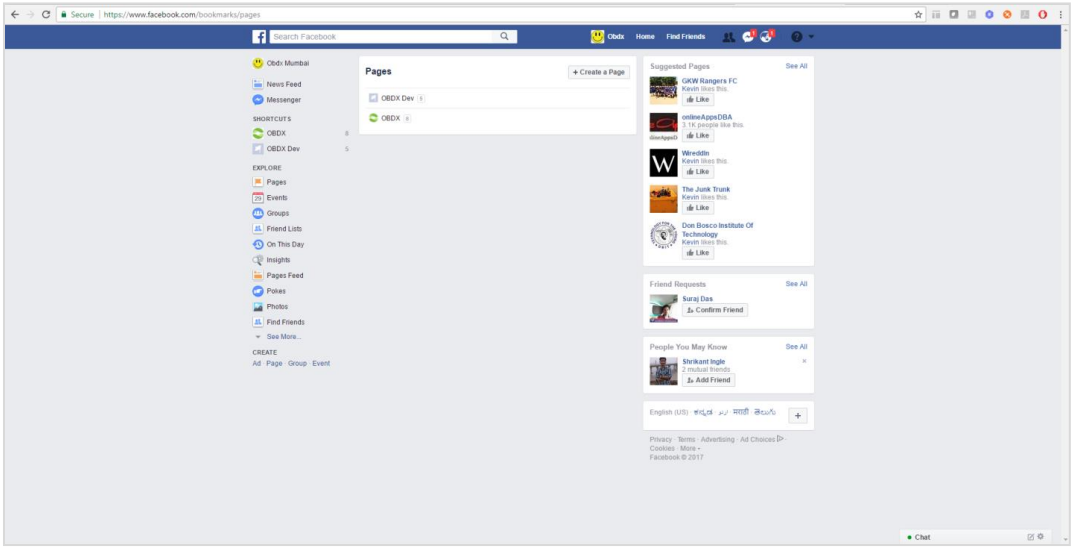


Home

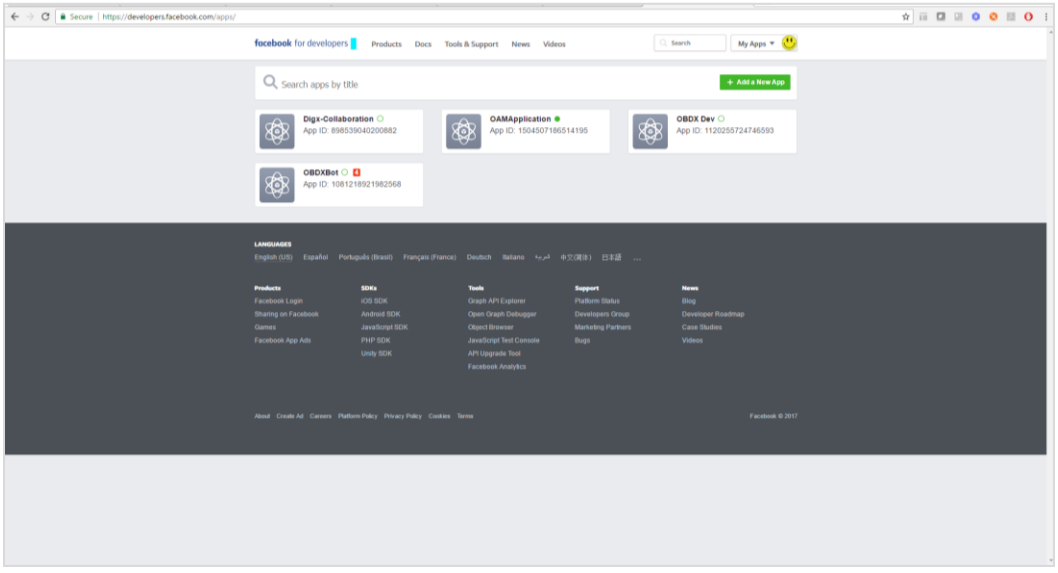
5. Facebook Configurations

Create a Facebook account for the Bank-

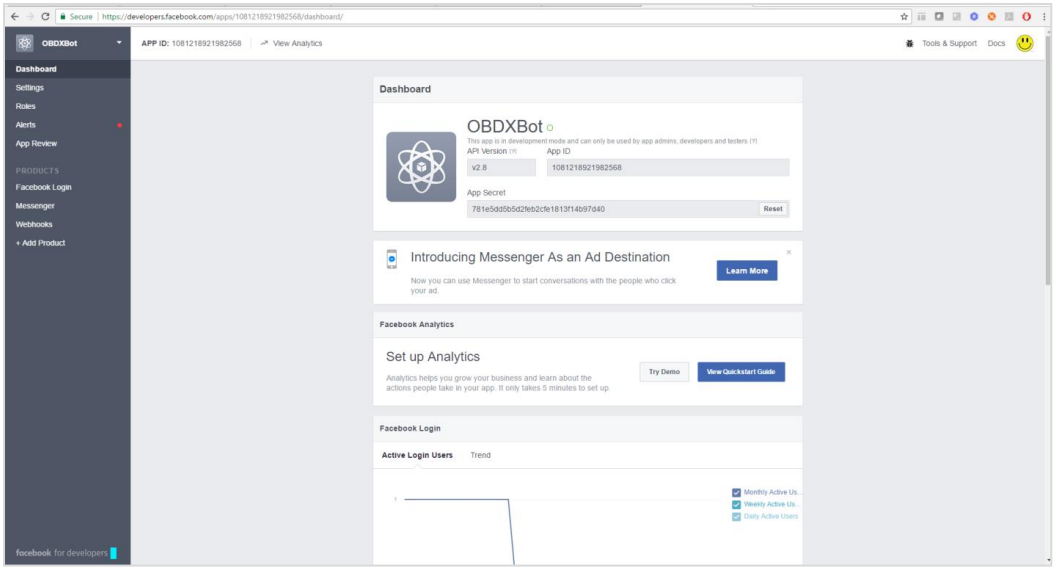
1. Login to Facebook with credentials.
2. Create a new page



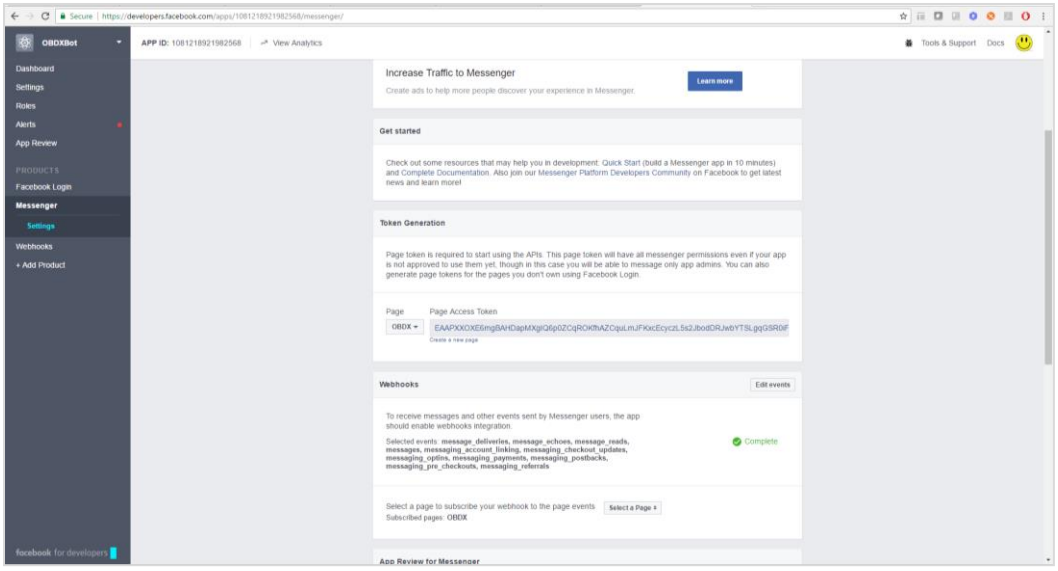
3. Navigate to developer link and create an application as shown below



4. Navigate to dashboard page and note the app secret as it will require in future steps.

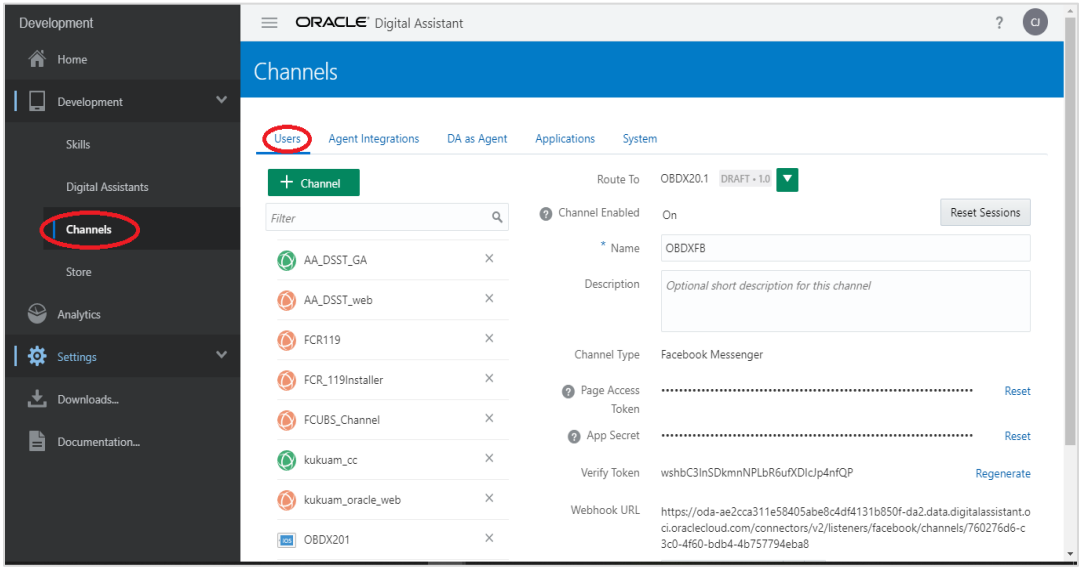


5. Navigate to Messenger > Settings page from left panel and in token generation section select the page created previously. Note the page access token.



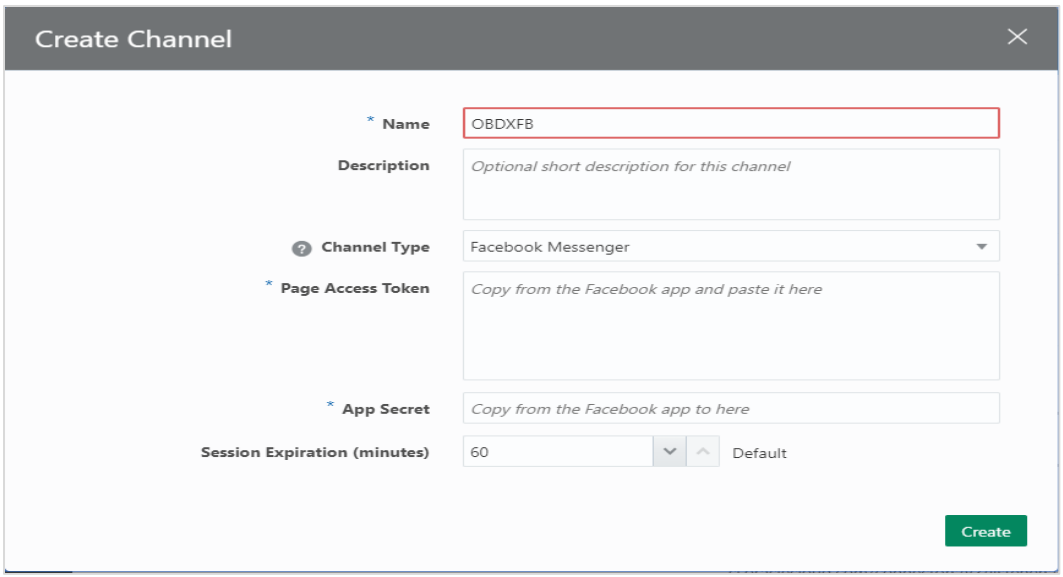
Create Facebook Channel in ODA

1. In ODA ,click channels in the left panel/menu and then click on users.



2. Next, click Add Channel to open the Create Channel dialog.

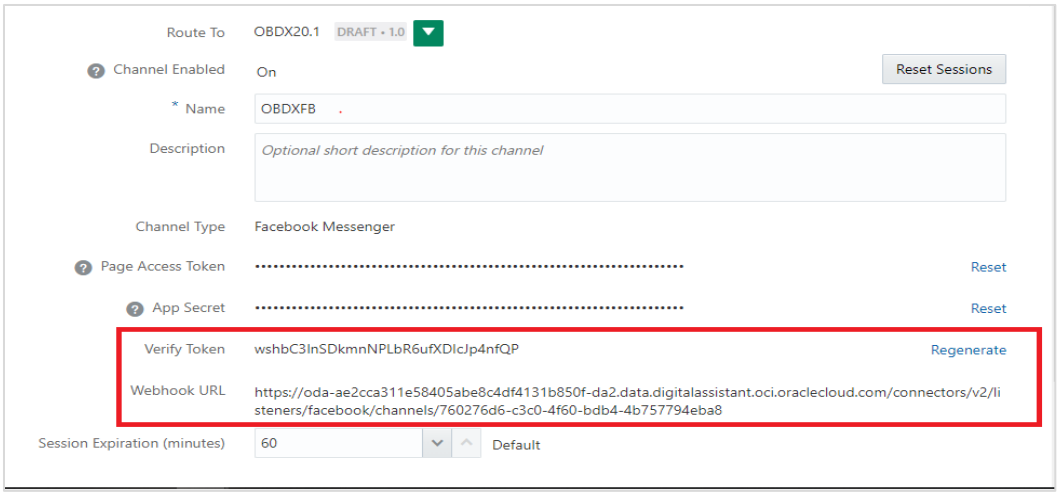
3. Choose Facebook Messenger as the channel type.



4. In the Page Access Token field, paste the page access token that you generated previously in the Set Up Facebook Messenger procedure.

5. In the App Secret field, paste the app secret that you copied previously in the Set Up Facebook Messenger procedure and click Create.

6. In the Channels page, copy both the Verify Token and WebHook URL and paste them somewhere convenient on your system. You'll need these to configure the Facebook webhook.



7. Select the digital assistant or skill that you want to associate with the channel. Switch on the Channel Enabled control to enable it.

Configure the Facebook Messenger Webhook

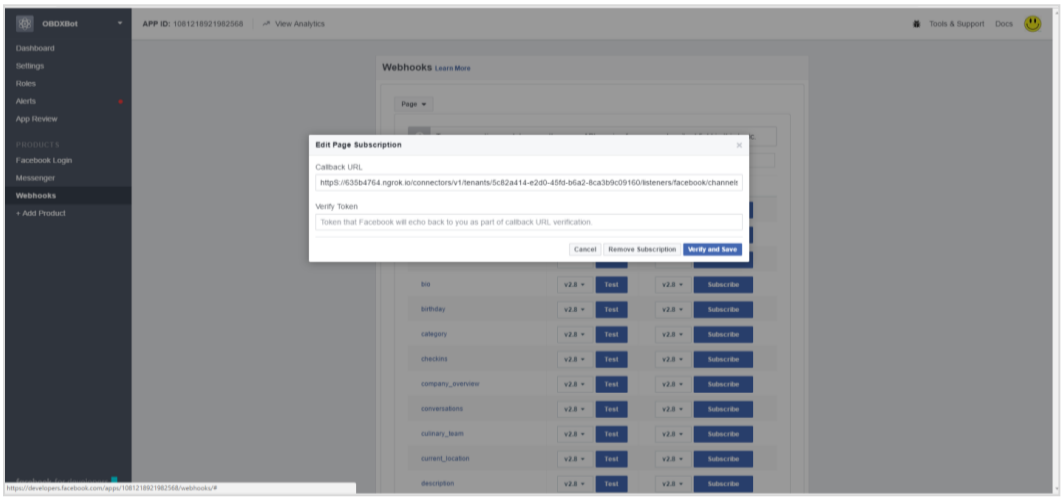
8. In Facebook Messenger, be sure that you've selected the project that you initially created for the webhook.

9. Click Messenger and then choose Settings .

10. Click Subscribe to Events to open the New Page Subscription dialog.

11. Copy the Webhook URL that you got from the Digital Assistant Channels page and paste it in the CallBack URL field in the New Page Subscription dialog.

12. Copy the Verify Token generated by Digital Assistant and paste it into the Verify Token field.

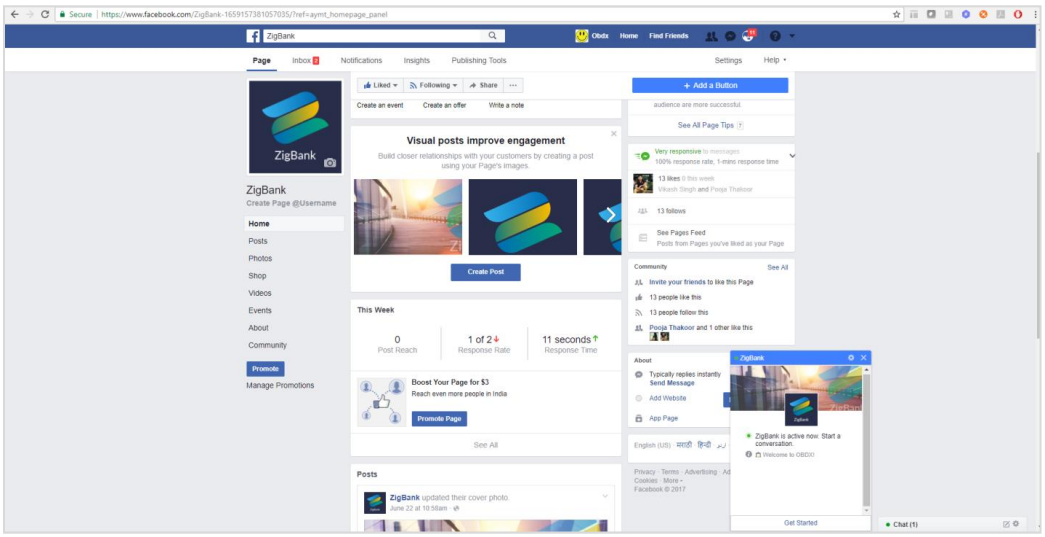


13. Under Subscription Fields, select the messages and messaging_postbacks callback events.The messages event is triggered whenever someone sends a message to your Facebook page.

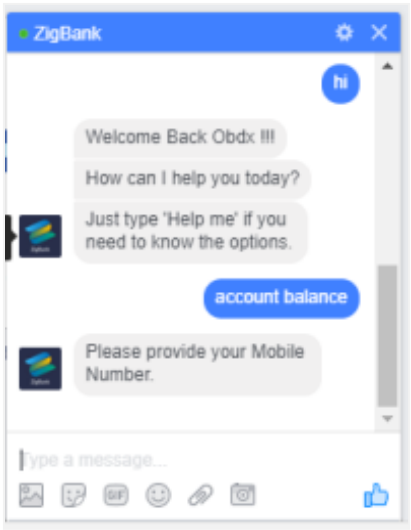
14. Click Verify and Save.
15. In the Webhooks section of the Messenger settings, select the Facebook page for your digital assistant (or standalone skill).Click Subscribe.

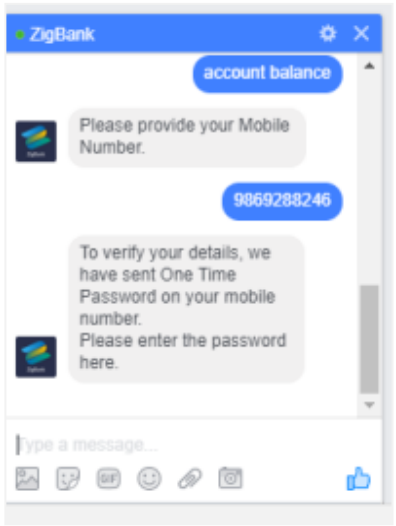
Validating configurations

Login to Facebook > Navigate to the page and click > Send message



1. Click Get Started in the chat window > You should receive welcome message from ODA
2. Enquire about account balance > OTP should be received on the registered email address of the party in core banking





Home

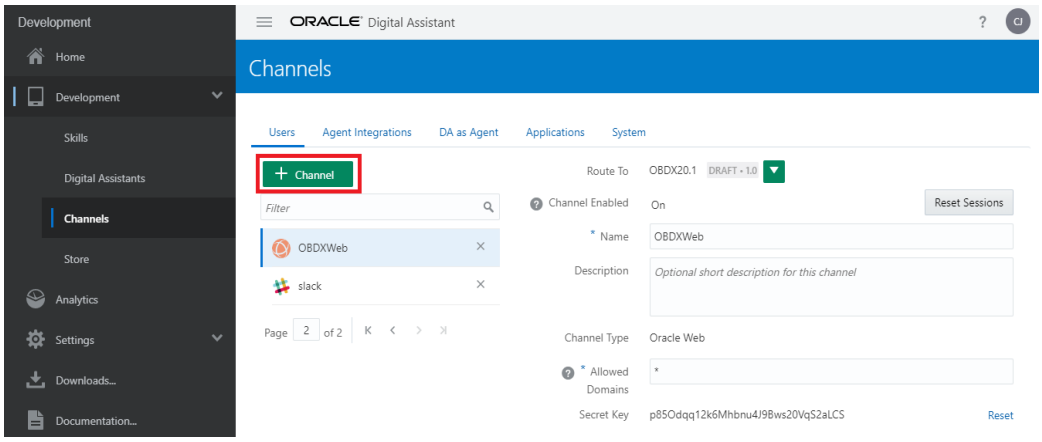
6. Configuring Channels in ODA

In addition to Facebook Messenger Channel, OBDX bot can be configured with Oracle Web, Oracle Android and Oracle IOS channel types to integrate with OBDX web/internet banking , OBDX Android, OBDX IOS applications respectively.

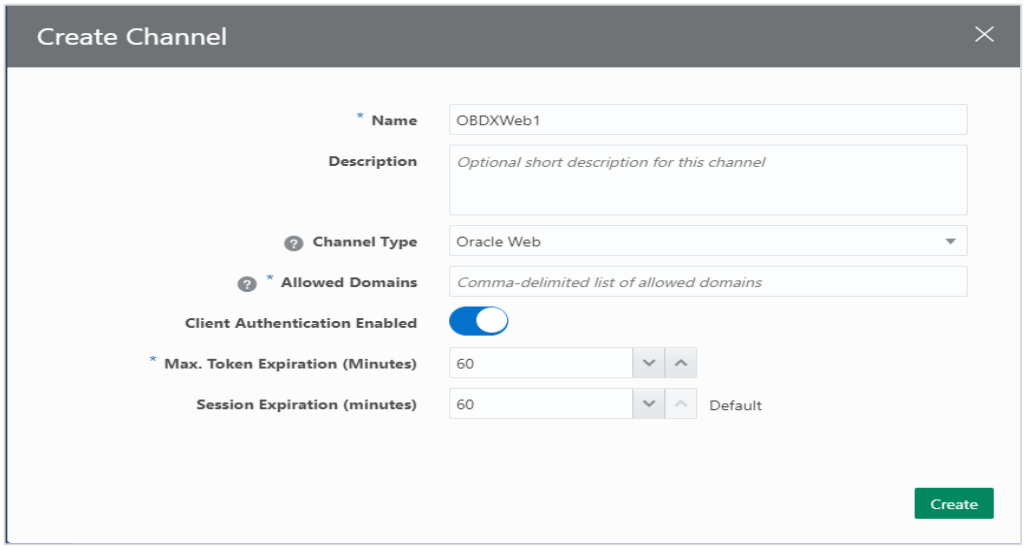
Oracle Web Channel

In order to configure this channel:

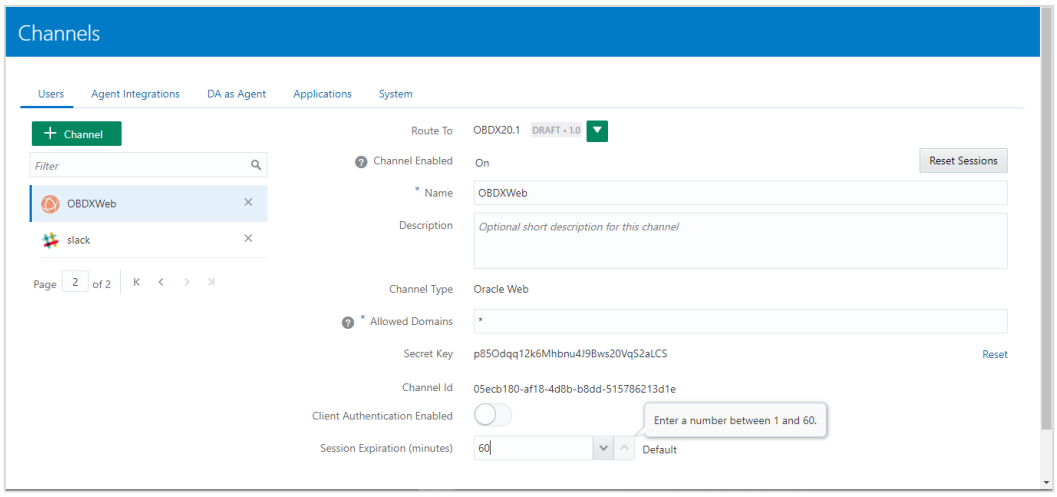
1. Choose Development ->Channels-> Users from the menu. Click **Add Channel.**



2. Create Oracle Web as the channel type.



3. Route the channel to your skill or digital assistant.
4. Switch Channel Enabled to On.



Chatbot widget appears only on prelogin page in OBDX UI

Add above Chatbot URL and channelId in /ui/framework/js/configurations/config.js in oda tag

To remove this chatbot widget

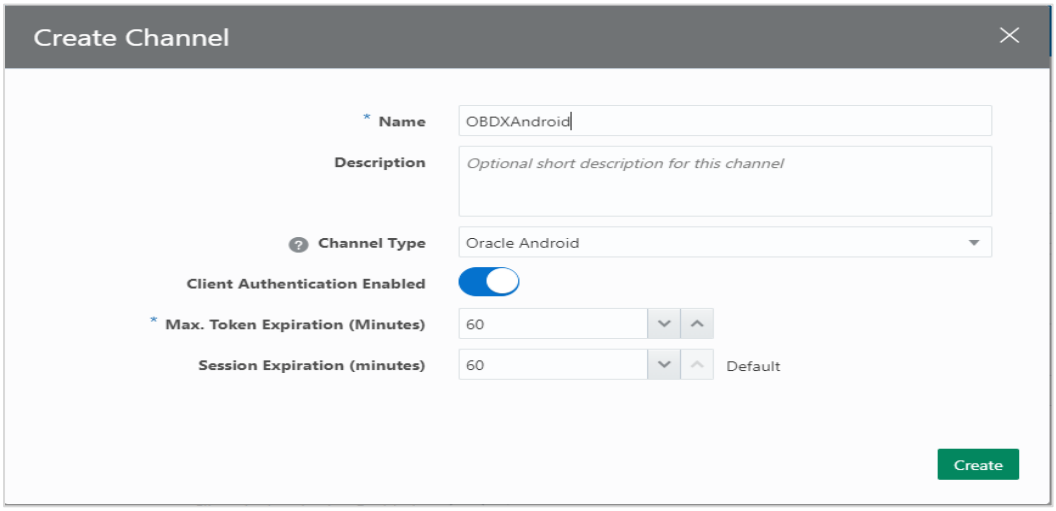
- Remove URL & channelId in above file
- Remove the “chat-bot” tag in prelogin dashboard → DIGX_DB_DASHBOARD → DASHBOARDDESIGN blob for anonymous dashboard

Oracle Android Channel

In order to configure this channel:

1. Choose Development->Channels->Users from the menu. Click **Add Channel.**

2. Then add Oracle Android as the channel type.

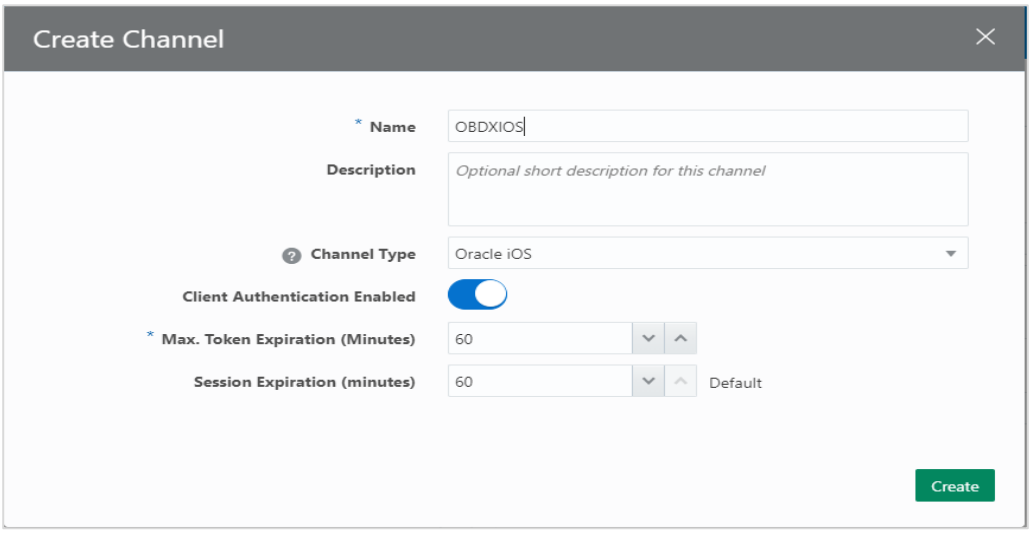


3. Route the channel to your skill or digital assistant.
4. Switch Channel Enabled to On.

Oracle iOS Channel

In order to configure this channel:

1. Choose Development->Channels->Users from the menu. Click Add Channel.
2. Add Oracle iOS as the channel type.



3. Route the channel to your skill or digital assistant.
4. Switch Channel Enabled to On.

Home

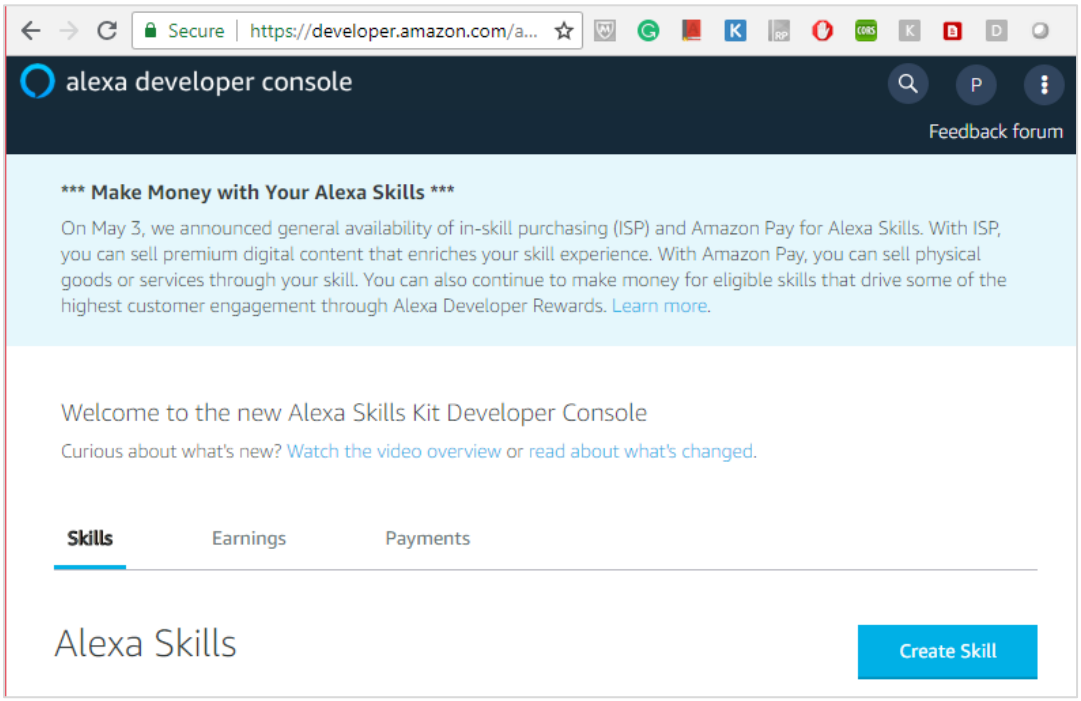
7. Alexa Skill (Zig Bank) Configuration

Creating an Alexa skill called *ZigBank* along with a Webhook channel lets you chat with a specific bot.

Add the skill information

Set up a developer account in the Amazon Developer Portal.

1. Open the **Amazon Developer Console.**
2. Click on **'Create Skill'**



3. Enter ZigBot (or any name that you want to use to invoke this skill) as the Invocation Name.

7.1 Define the Interaction Model

1. Next, add the CommandBot intent, which sends a voice text to the configured bot. **Copy and Paste** following intent schema into the Developer Console's JSON Editor and then click on **'Save Model'**

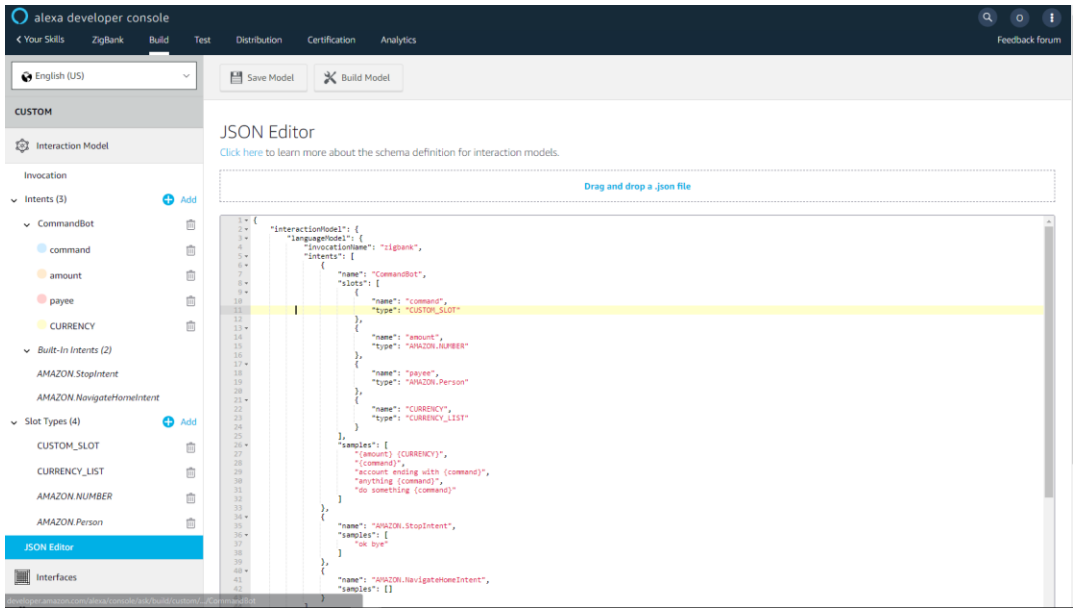
```
{
  "interactionModel": {
    "languageModel": {
      "invocationName": "zigbank",
      "intents": [
        {
          "name": "CommandBot",
          "slots": [
            {
              "name": "command",
              "type": "CUSTOM_SLOT"
            },
            {
              "name": "amount",
              "type": "AMAZON.NUMBER"
            },
            {
              "name": "payee",
              "type": "AMAZON.Person"
            },
            {
              "name": "CURRENCY",
              "type": "CURRENCY_LIST"
            }
          ],
```

```
      "samples": [
        "{amount} {CURRENCY}",
        "{command}",
        "account ending with {command}",
        "anything {command}",
        "do something {command}"
      ]
    },
    {
      "name": "AMAZON.StopIntent",
      "samples": [
        "ok bye"
      ]
    },
    {
      "name": "AMAZON.NavigateHomeIntent",
      "samples": []
    }
  ],
  "types": [
    {
      "name": "CUSTOM_SLOT",
      "values": [
        {
          "name": {
            "value": "0012",
            "synonyms": [
              "zero zero one two"
```

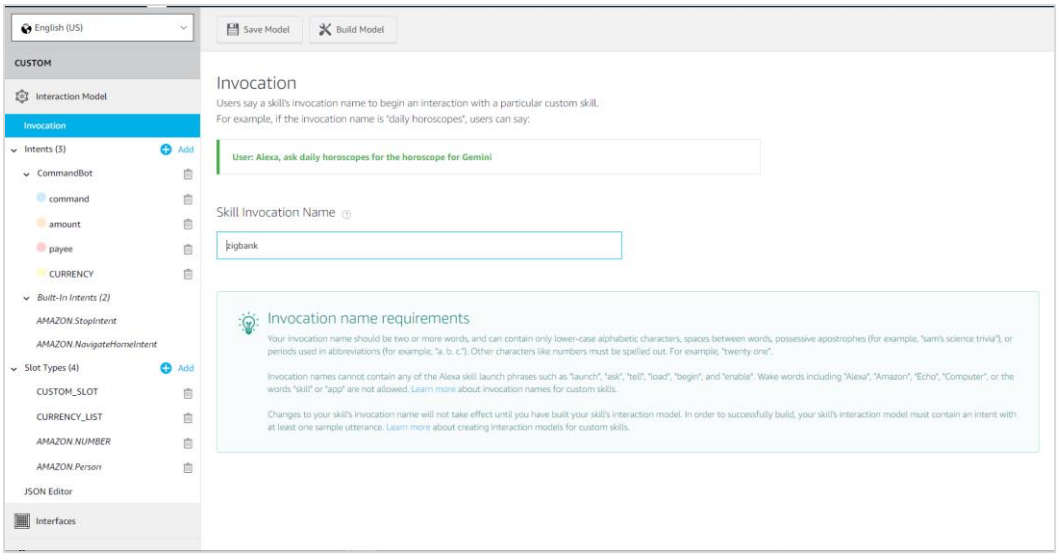
```
        ]
      }
    },
    {
      "name": {
        "value": "0045",
        "synonyms": [
          "zero zero four five"
        ]
      }
    },
    {
      "name": {
        "value": "VODAFONE 4G",
        "synonyms": [
          "vodafone"
        ]
      }
    },
    {
      "name": {
        "value": "AIRTEL",
        "synonyms": [
          "AIRTEL BROADBAND",
          "bharti airtel"
        ]
      }
    },
```

```
            {
               "name": {
                  "value": "RELIANCE",
                  "synonyms": [
                     "REL"
                  ]
               }
            }
         ]
      },
      {
         "name": "CURRENCY_LIST",
         "values": [
            {
               "name": {
                  "value": "GBP",
                  "synonyms": [
                     "gbp"
                  ]
               }
            },
            {
               "name": {
                  "value": "EURO",
                  "synonyms": [
                     "euro"
                  ]
               }
```

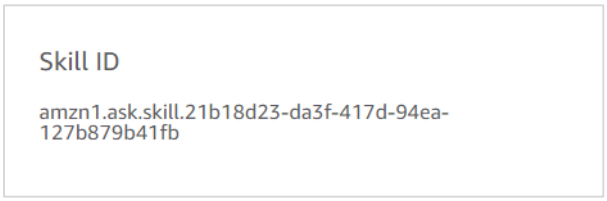
```
            }
          ]
        }
      ]
    }
  }
}
```



2. Click on **'Build Model'**

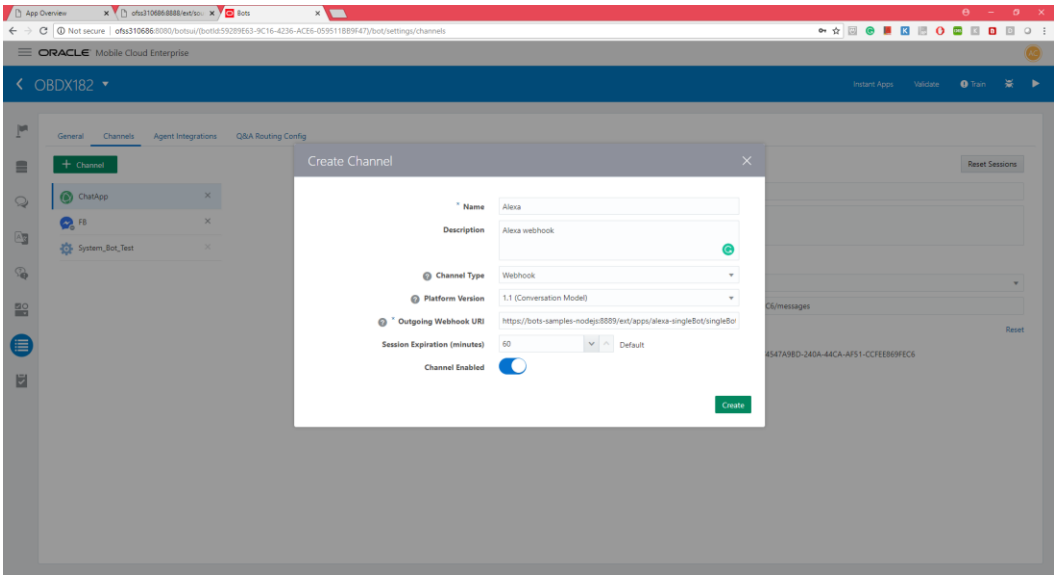


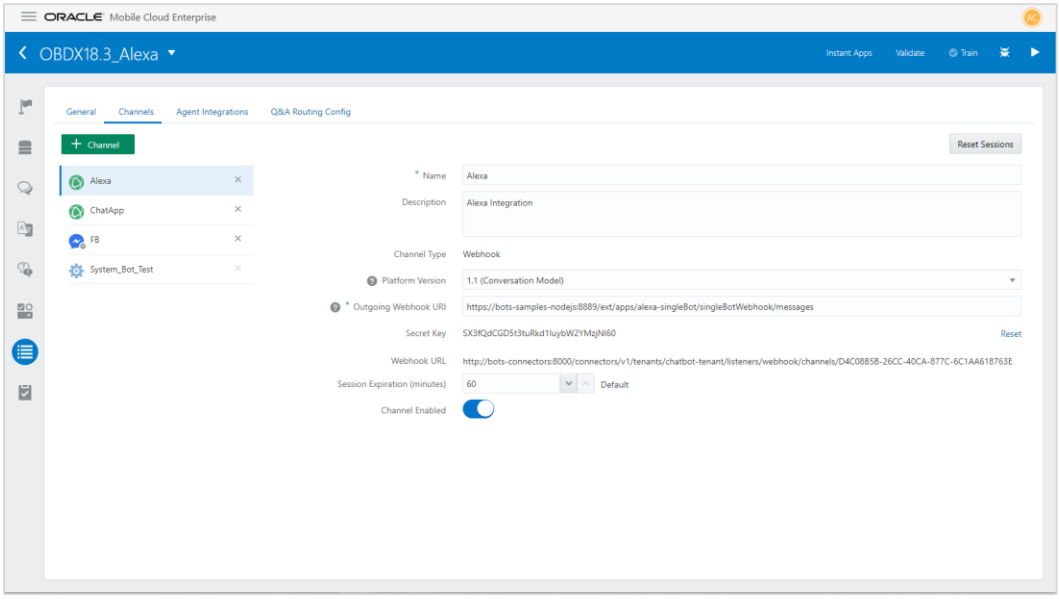
3. Also note down the alexa skill id, we will need it in next step.



7.2 Create a Webhook channel

1. In the Bot Builder, create a webhook channel for your bot. In the Create Channel dialog, enter the outgoing Webhook URL as **https://bots-samples-nodejs:8889/ext/apps/alexa-singleBot/singleBotWebhook/messages.** This URL is where your bot will send its responses back to the Alexa ZigBot skill.





2. Keep the Secret Key and Webhook URL close by because you need to add them to the app.js file. Also, remember to set the amazon skill id (created in previous steps). For example:

```
var metadata = {

      allowConfigUpdate: true,

      waitForMoreResponsesMs: 200,

      amzn_appId: "amzn1.ask.skill.21b18d23-da3f-417d-94ea-127b879b41fb",

      channelSecretKey: 'SX3fQdCGD5t3tuRkd1IuybW2YMzjNl60',

      channelUrl: 'http://bots-connectors:8000/connectors/v1/tenants/chatbot-tenant/listeners/webhook/channels/D4C08B5B-26CC-40CA-877C-6C1AA618763E'

  };
```

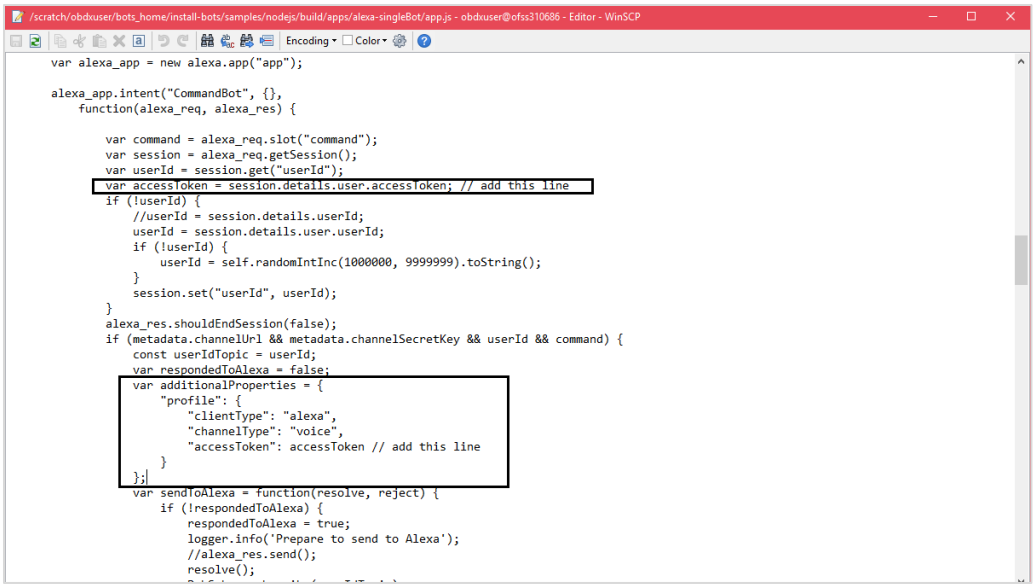
app.js file is located at BOTS_HOME/samples/nodejs/build/apps/alexa-singleBot/app.js

```
const alexa = require("alexa-app");
const _ = require("underscore");
const express = require('express');
const bodyParser = require('body-parser');
const PubSub = require('pubsub-js');
const Joi = require('joi');
const MessageModel = require('../bots-js-utils/lib/messageModel/MessageModel.js')(Joi);
const messageModelUtil = require('../bots-js-utils/lib/messageModel/messageModelUtil.js');
const botUtil = require('../bots-js-utils/lib/util/botUtil.js');
const webhookUtil = require('../bots-js-utils/lib/webhook/webhookUtil.js');

PubSub.immediateExceptions = true;

module.exports = function() {
    var self = this;

    //replace these settings to point to your webhook channel
    var metadata = {
        allowConfigUpdate: true, //set to false to turn off REST endpoint of allowing update of metadata
        waitForMoreResponsesMs: 200, //milliseconds to wait for additional webhook responses
        amzn_appId: "amzn1.ask.skill.21b18d23-da3f-417d-94ea-127b879b41fb",
        channelSecretKey: 'SX3fQdCGD5t3tuRkd1IuybW2YMzjNl60',
        channelUrl: 'http://bots-connectors:8000/connectors/v1/tenants/chatbot-tenant/listeners/webhook/channels/D4C08B5B-26CC-40CA-877C-6C1AA618763E'
    };

    this.randomIntInc = function(low, high) {
        return Math.floor(Math.random() * (high - low + 1) + low);
    };

    this.setConfig = function(config) {
        metadata = _.extend(metadata, _.pick(config, _.keys(metadata)));
    }
```

3. You can also edit the welcome message in the app.js.

```
alexa_app.launch(function (alexa_req, alexa_res) {
  var session = alexa_req.getSession();
  session.set("startTime", Date.now());
  alexa_res.say("Welcome to Zig Bank. How may I help you?");
    alexa_res.shouldEndSession(false);
});
```

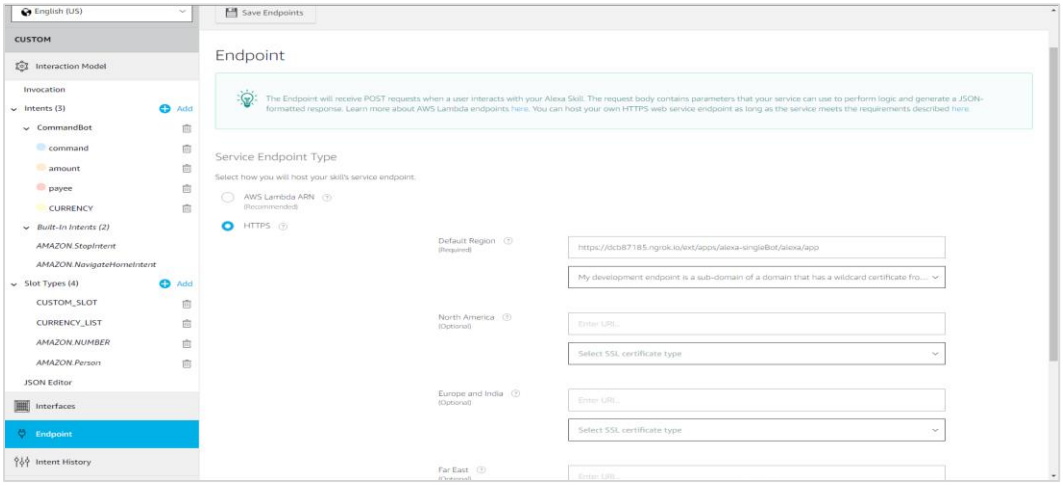
4. To pass accessToken to the OBDX Chatbot endpoint, add these additional properties in app.js (Optional)



5. Restart the `bots-samples-nodejs` container.

7.3 Configure the Endpoint

1. Choose **HTTPS**
2. Enter the HTTPS ngrok URL for port 8888 that's appended with `ext/apps/alexa-singleBot/alexa/app`. For example: `https://<ngrok URL for port 8888>/ext/apps/alexa-singleBot/alexa/app`
3. Select SSL Certificate. Choose '**My development endpoint is a sub-domain of a domain that has a wildcard certificate from a certificate authority'**.
4. Save Endpoint

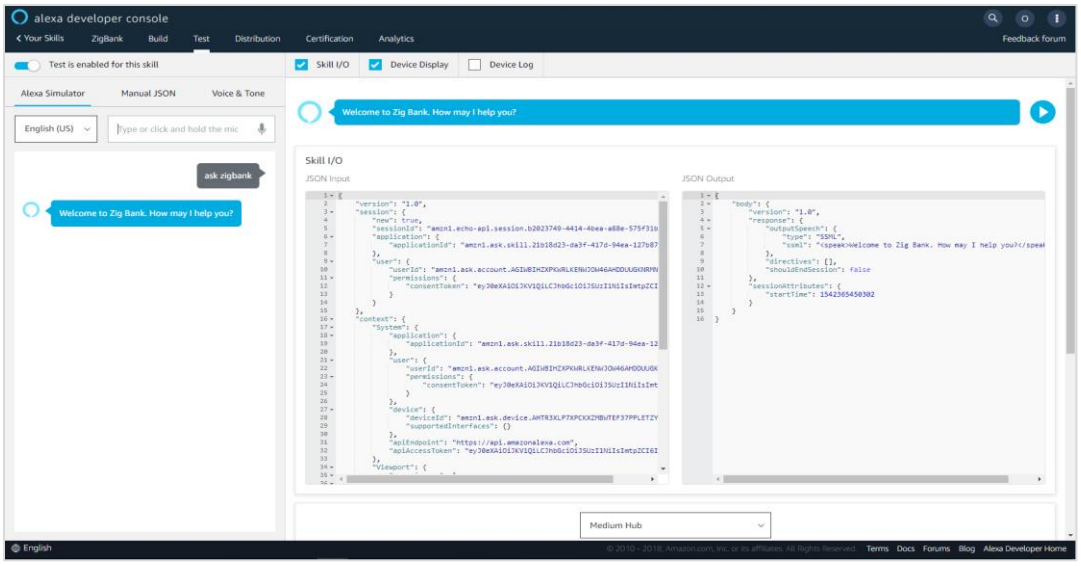


7.3.1 Testing the singleBot Skill in the Amazon Developer Console

To test the skill, enter the following utterance in the Service Simulator

- ask zigbank

For each utterance, the Service Simulator window displays the response.



7.3.2 Testing ZigBot on an Echo Device

If your Echo device is logged to the same user account that accesses the Developer Console, then the ZigBot skill will enabled in your Amazon Echo. Try out the same utterances, but start each one with "Alexa ask zibo…" If Alexa can't understand you, or your Echo's light ring is turned off, start over by saying, "Alexa ask zibo…" For example:

"Alexa ask zibo to show my balances"

==> "ssml": "<speak>For which account do you want your balance… </speak>"

"Alexa stop" or "stop" to end the interaction

If Alexa continually misunderstands you, take a look at the bots-samples-nodejs log to see which of your commands were picked up by Alexa and sent to the bot and which weren't. (docker logs <container id of bots-samples-nodejs>)

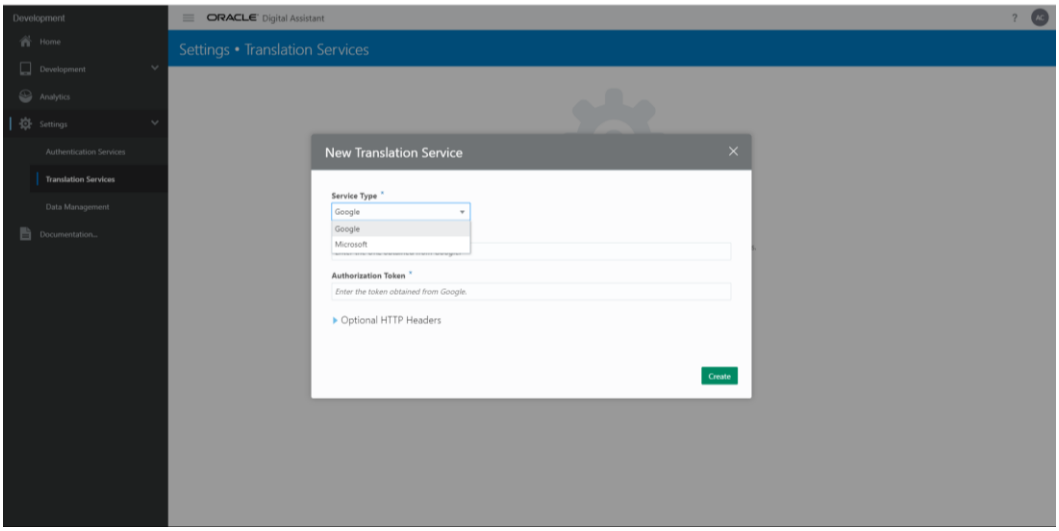
Note: Alexa might have trouble with some interactions, like following a web link or saying a number. Alexa spells out numbers (22 becomes twenty-two), which might be problematic if your bot is expecting a cardinal number. Also, Alexa won't wait for a result when web services are slow. When this happens, Alexa will state that your skill takes too long to respond.

7.3.3 Configure Account Linking

Account linking lets you connect the identity of the user with a user's account in OBDX system.

1. Click on **Account Linking**
2. Select **Auth Code Grant**
3. Enter Authorization URL https://<ngrok URL of OHS server port>/ digx-auth/oauth2/authz
4. Enter Access Token URL https://<ngrok URL of OHS for port>/ digx-auth/v1/token
5. Enter client ID and client secret
6. Select Credentials in request body
7. Add Scope OBDXVoiceAstServer.SC02 and OBDXVoiceAstServer.SC05(SC02 is default scope in case if no scopes are externally added.)
8. Click **Save**

7.3.4 Translation Services



In case input language to chatbot is other than english, configure translation services as shown above.

Home