Oracle® Financial Services Lending and Leasing

OFSLL TransactionBot Overview and Developer guide

Release 14.12.0.0.0
F82323-01
August 2024

Oracle Financial Services Lending and Leasing OFSLL TransactionBot Overview and Developer guide, Release 14.12.0.0.0

F82323-01

Contents

1

OFSLL Transaction BOT Overview and Developer Guide

OFSLL has an extended out of the box support for CHATBOT integration. This provides a new framework for direct user interaction with the system. However, since OFSLL is a back-office system there are additional external components required to be integrated to host and utilize the CHATBOT functionality.

For latest version of this document, refer to Oracle Help Center

This section consists of the following topics:

- OFSLL Transaction BOT Overview and Developer Guide
- Developer Guide for BOT Customization

Following topics are discussed in OFSLL Transition BOT Overview chapter:

- Introduction
- Architecture
- Third Party Licenses
- Features of BOT
- Sample Workflow
- Launch OFSLL Transaction BOT
- BOT UI Elements
- BOT Usability Workflow

1.1 Introduction

Currently, OFSLL integration with CHATBOT is supported with some of the functionalities such that end users can search for documentation and / or query and fetch the account related information and/or perform other actions on an account with options presented in CHATBOT menu.

This document outlines the integrated framework and procedures required to implement certain features, but it is not a general-purpose configuration manual.

- Transaction Bot Overview
- Purpose
- Audience
- Accessibility
- Access
- Prerequisites

1.1.1 Transaction Bot Overview

OFSLL integrated Transaction bot (Transaction posting chatbot) is a functionality for product end-users to query account related details, outstanding dues and post simple account related updates as a transaction. In addition, there is also dynamic content search capability provided within the Transaction bot. For information on Documentation search using chatbot, refer to **OFSLL Docubot Overview and Developer Guide**.

The Transaction ChatBot is hereafter is referred to as **BOT** in the document.

1.1.2 Purpose

The purpose of this document is to demonstrate the capability of OFSLL BOT in handling transactional updates to accounts maintained in the system by integrating with Oracle Digital Assistant (ODA). This document is intended to detail the usability features and also to serve as a developer guide to understand the configuration procedures. However, the features and options presented are provided only as a sample and needs further customization based on requirements.

1.1.3 Audience

In general, this document is intended to all those parties and decision makers who are interested to know about OFSLL BOT integrated framework. The configuration sections are intended for system administrators, consulting and implementation teams who deploy customized solutions for customer.

1.1.4 Accessibility

The OFSLL BOT integrated framework is supported from OFSLL 14.12.0.0.0 release.

OFSLL being a back office system, only the data in the system is can be exposed using REST services and the interface for BOT facility is recommended to be configured on any 3rd party web application or customer self-service portal or lenders/financial services website for the benefit of end-users.

However, the account related services provided in this framework is just a sample and needs to be customized based on requirement. BOT is agnostic of which self-service site / portal is used to provide access and interface to the users for help documentation.

1.1.5 Access

Currently the framework supports basic authentication (not OAUTH). User Management and authentication needs to be handled as part of the implementation.

1.1.6 Prerequisites

Following are the prerequisites:

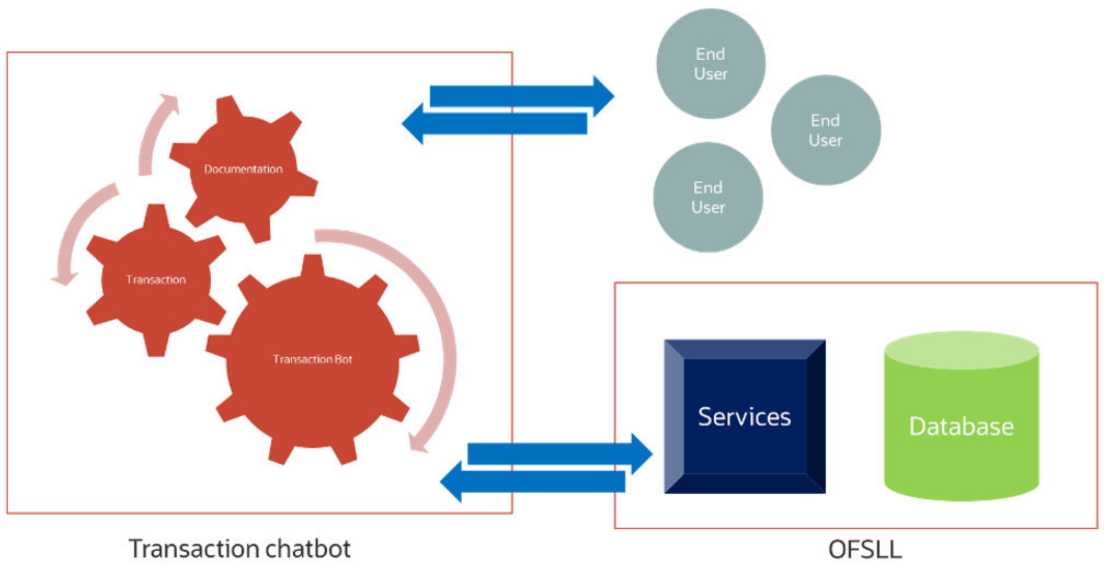
- The BOT is designed to work in ODA framework (platform version 21.02). The configuration is to be done as detailed in Developer Guide for BOT Customization section.
- Also the ODA Server Environment has to be licensed separately. For more information, refer to https://www.oracle.com/in/chatbots/digital-assistant-platform/
- Need to have release specific pre-indexed file for elastic search to work.

- Adequate space to store the indexed file directories in the respective folders.
- WebLogic server for deployment of war file (`OracleFSLLChatBot.war`).
- The parameters in `Channel.Properties` file are to be configured before creating and deploying the `.war` file (`OracleFSLLChatBot.war`). For details, refer to BOT Configuration section.

1.2 Architecture

The BOT connects to OFSLL and provides an interface to give results for the below mentioned table items. With the current structure BOT seamlessly integrates with Services and documentation of the current release of the product.

Figure 1-1 Transaction Bot_Architecture



The documentation elastic search for OFSLL BOT requires pre-indexing of content. Hence, indexing is done for 14.12.0.0.0 release. The indexing process is done automatically using the third-party plugins such as Apache Lucene and Jsoup to identify unique keywords in HTML files. This generates indexed files which serves as common directory for searched keyword and the file instance where it exists.

For more information on third-party plugins used, refer to Third Party Licenses section.

1.3 Third Party Licenses

OFSLL BOT uses the following third party licenses:

- Apache Lucene, Version: 8.10.1
  The Apache Software Foundation, Technology: Lucene, Version: 8.10.1

  Files used (below are part of Apache Lucene 8.10.1)

  Lucene Core (8.10.1)

  Lucene query parser (8.10.1)

- JSOUP 1.14.3
  Jsoup is a Java library for working with real-world HTML.

  It provides a very convenient API for fetching URLs and extracting and manipulating data, using the best of HTML5 DOM methods and CSS selectors.

  jsoup implements the WHATWG HTML5 specification, and parses HTML to the same DOM as modern browsers do.

  scrape and parse HTML from a URL, file, or string

  find and extract data, using DOM traversal or CSS selectors

  manipulate the HTML elements, attributes, and text

  The purpose of using Jsoup in chatbot is to read the html elements <tags> <href> and use it as a added part of indexing

  Link : https://jsoup.org

For detailed information, refer to product licensing guide.

1.4 Features of BOT

Following are the unique features of OFSLL BOT:

- Account details view using Account # query
  - View Account Details Summary
  - View Payment Details
  - Check the Next Payment Date
  - View and Update default Communication Preference
  - View Credit Limit Details
- Readily available navigation links to the following:
  - Link to all Release documentation
  - Dynamic Document Search option
  - Link to currently mapped Product Release notes
  - Listing of Product Module / Classified Guides
  - Link to list of indexed Keywords
  - Link to Getting Started Video gallery
  - Link to Release Highlights
- Intuitive Menu options:
  - Option to clear chat data
  - Speech Conversion – Voice based Input
  - Personalization of BOT interface

This topic consists of the following sections:

- Support of Text and Voice Based inputs
- Release Specific Indexing

1.4.1 Support of Text and Voice Based inputs

The BOT can support both Text and Voice based inputs to find information. This attempts to comply with multiple accessibility options.

The BOT is enabled with voice based inputs where in voice commands are accepted as input equivalent to typing or clicks. This option works on clicking the Mic button.

During text based input, the response is provided in the BOT interface. In a voice based input, the response is provided in both voice based response and BOT response simultaneously.

However, note that voice based input does not support to open a URL (link) reference.

1.4.2 Release Specific Indexing

Indexing is done for the following release of OFSLL and indexed files are provided in respective folder. The mapping of Release number v/s Folder name and Part Number is indicated below:

Table 1-1 Release Specific Indexing

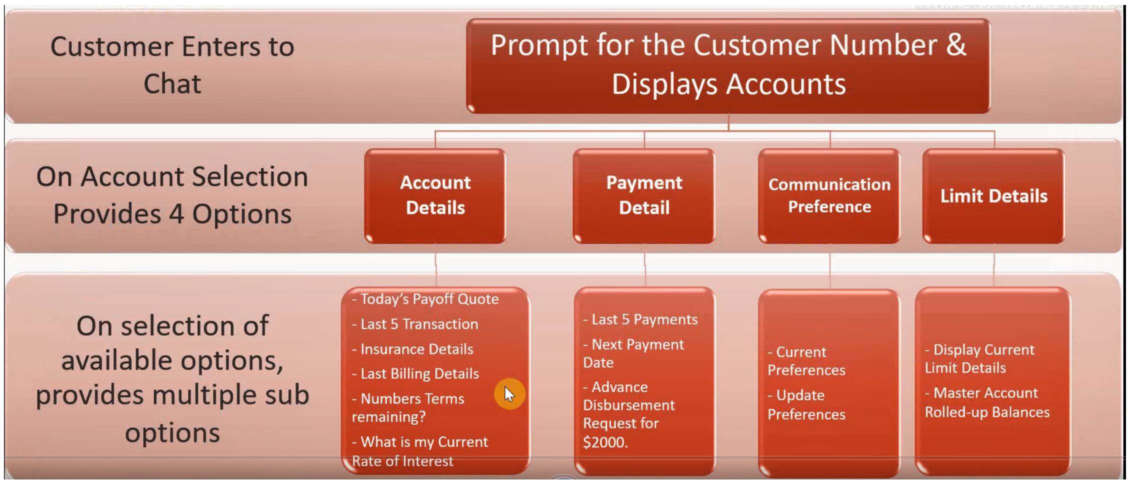
Release No	Folder Name	Part Number
14.12.0.0.0	14.12	F53373_01

1.5 Sample Workflow

While interacting with BOT, you need to input the basic details (like customer ID) to start and further drill down to explore multiple account options available.

Following image is an illustration of the workflow and also, one of the scenario is detailed as an example to indicate the BOT workflow in BOT Usability Workflow section.

Figure 1-2 Sample Workflow


Customer Enters to Chat
Prompt for the Customer Number & Displays Accounts
On Account Selection Provides 4 Options
Account Details
Payment Detail
Communication Preference
Limit Details
On selection of available options, provides multiple sub options
- Today's Payoff Quote
- Last 5 Transaction
- Insurance Details
- Last Billing Details
- Numbers Terms remaining?
- What is my Current Rate of Interest
- Last 5 Payments
- Next Payment Date
- Advance Disbursement Request for $2000.
- Current Preferences
- Update Preferences
- Display Current Limit Details
- Master Account Rolled-up Balances

To Start with, enter your name and confirm if you want to continue using the bot. Based on your intent the bot starts building the answers.

Table 1-2 Transaction Bot_Sample Workflow

Sl.No.	Menu
1	Begin with entering a Customer ID / Account number.
2	Click on required account from the list of accounts belonging to the Customer ID
3	View the Account Summary. Click Payment Details option form the list.
4	View the account details with below menu options Today's payoff Quote Last 5 Transactions Insurance Details Last Billing details Number of Terms remaining Need more help
5	View the payment details Last 5 payments Next Payment date Advance disbursement request for 2000 $ Need more help
6	View the communication preference Current Preferences Update Preference Need more help
7	View the Limit details Display current limit details Master Account rolled-up Balances Need more help
8	Click on the **OFSLL documentation tree** The available options are OFSLL release documentation Document Search Product release notes Product classified guides Find by indexed keyword Getting started videos Release Highlights Need more help
9	Click **Need more help** The user has a the option to continue with the same customer id or enter new customer id

Also, one of the scenario is detailed as an example to indicate the Chatbot workflow in OFSLL. Refer to BOT Usability Workflow section.

1.6 Launch OFSLL Transaction BOT

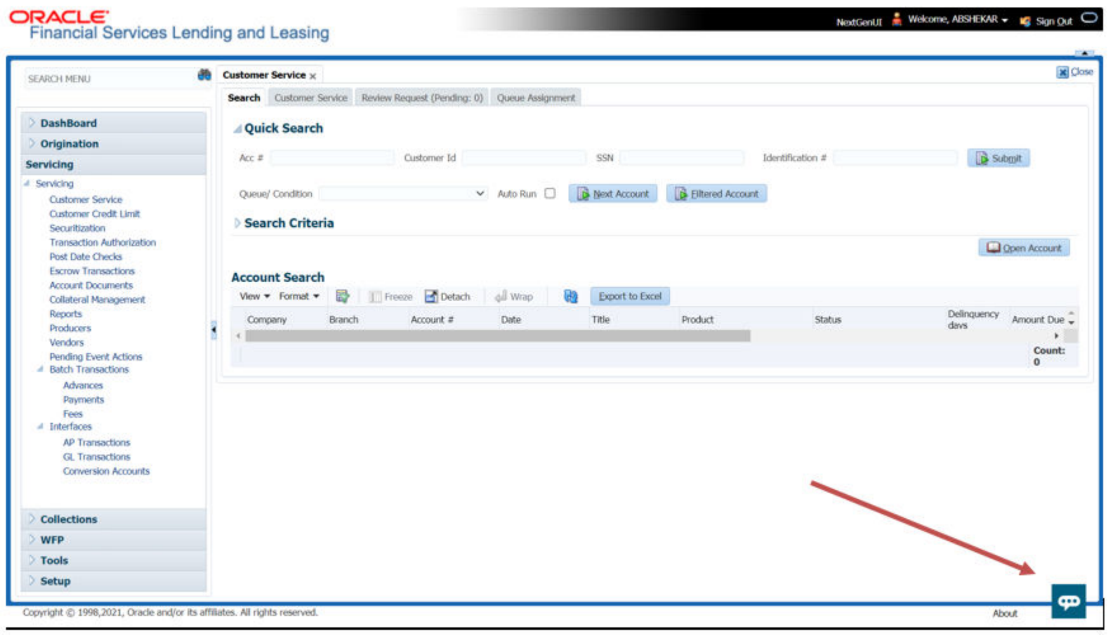
OFSLL Transaction BOT is accessible after logging in to OFSLL application. This BOT can either be in enabled or disabled status by default depending on the weblogic csf configuration (refer section 2.5 in this document). If enabled, on login of OFSLL application the BOT is available at right bottom corner.

Note:

Before you being, ensure to perform the required configuration as detailed in Developer Guide for BOT Customization chapter.

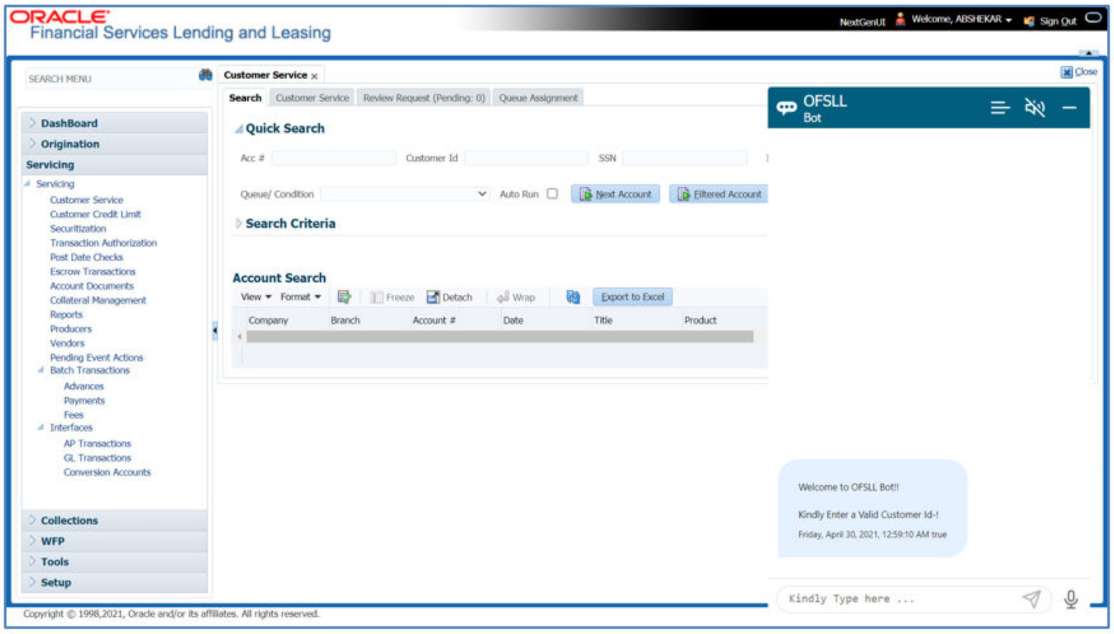
The BOT after login is as shown below:

Figure 1-3 Launch OFSLL Transaction BOT



On clicking bot icon, the interface is as displayed:

Figure 1-4 Transaction BOT_Interface



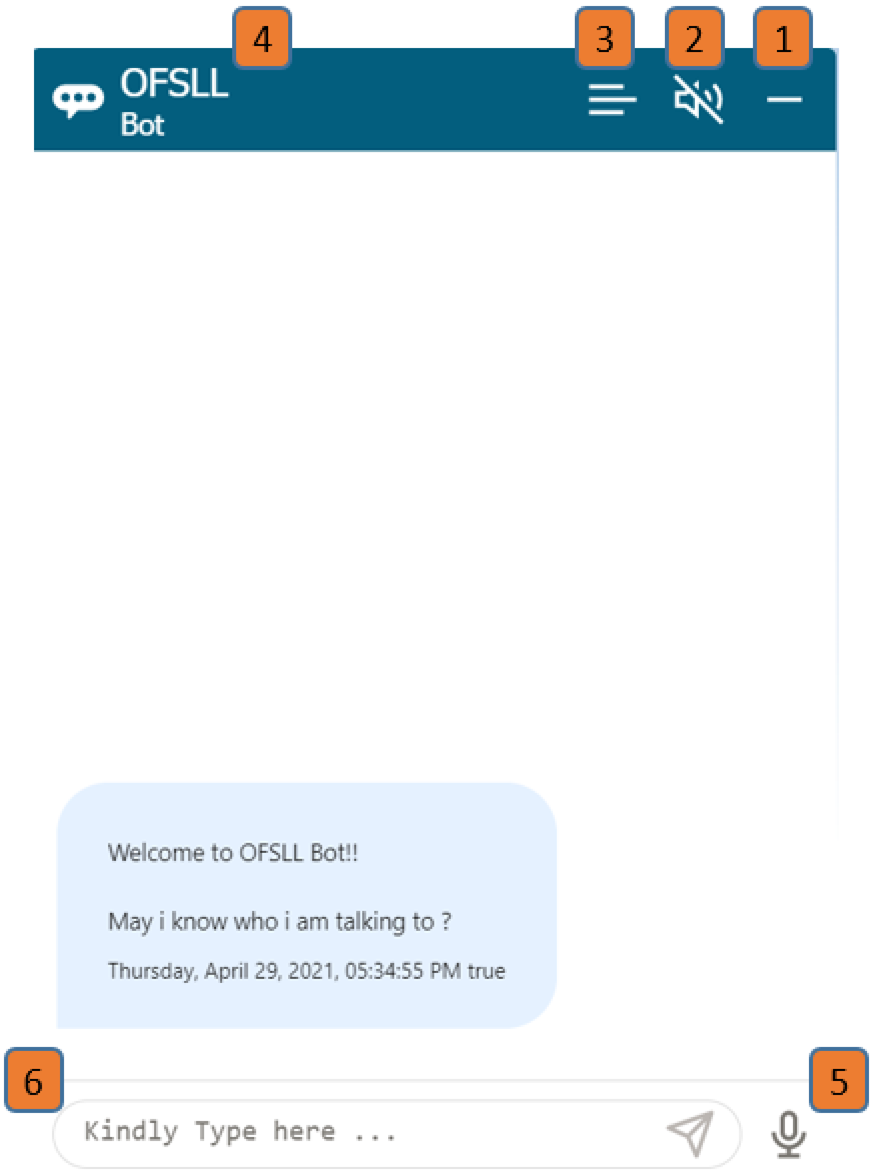
Welcome to OFSLL Bot!!

Kindly Enter a Valid Customer Id-!

Friday, April 30, 2021, 12:59:10 AM true

1.7 BOT UI Elements

Figure 1-5 BOT UI Elements



Welcome to OFSLL Bot!!

May i know who i am talking to ?

Thursday, April 29, 2021, 05:34:55 PM true

Table 1-3 BOT UI Elements

Sl.No	Option	View / Action
1	Minimize	Minimize BOT window
2	Speaker output	Enable BOT in speaker mode
3	Clear chat	Clear all messages in the BOT

Table 1-3 (Cont.) BOT UI Elements

Sl.No	Option	View / Action
4	Customized label	Customization for title label is detailed in **Bot Customization** section of 'OFSLL Docubot Overview and Developer guide'.
5	Mic Input	Enable Mic for voice based input
6	Text Input	Enter search string using keyboard

1.8 BOT Usability Workflow

Following is a sample workflow indicating the steps performed in chatbot. You can perform the following:

- View Account Details
- View Payment Details
- View and update Communication Preferences
- View Credit Limit Details
- View OFSLL documentation tree

Table 1-4 BOT Usability Workflow

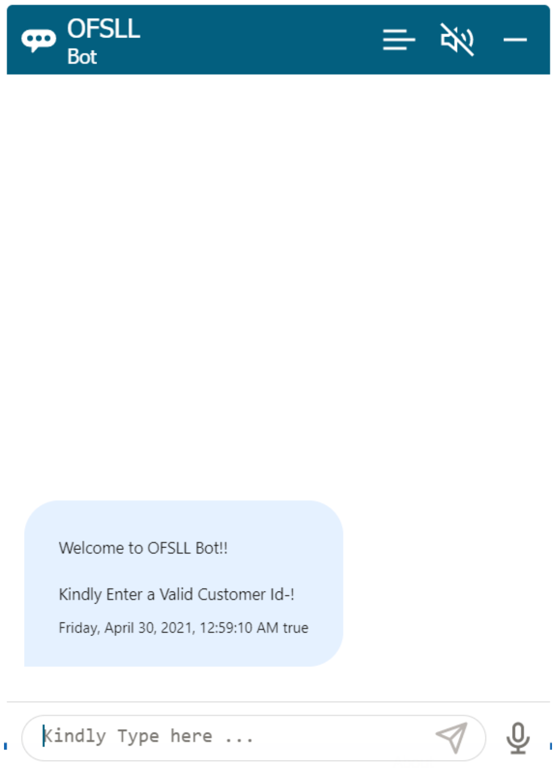
Action	BOT Response
Begin with entering a Customer ID / Account number.	 Welcome to OFSLL Bot!! Kindly Enter a Valid Customer Id-! Friday, April 30, 2021, 12:59:10 AM true

Table 1-4 (Cont.) BOT Usability Workflow

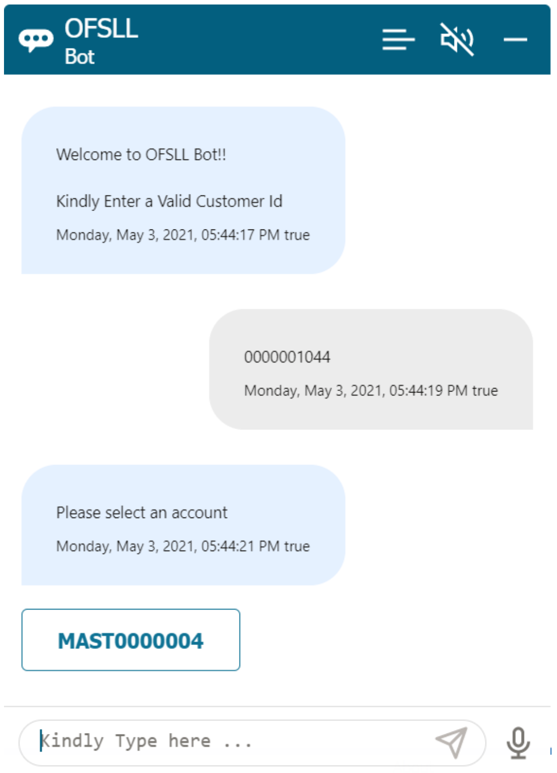
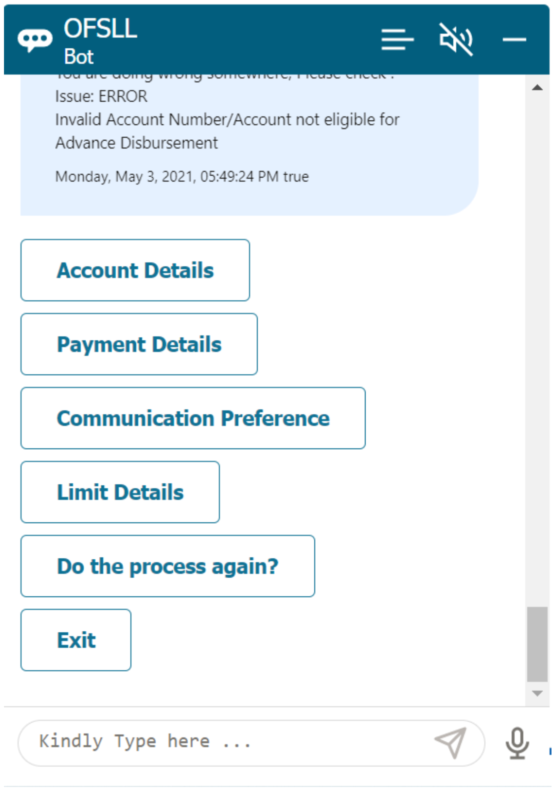
Action	BOT Response
Click on required account from the list of accounts belonging to the Customer ID	 OFSLL Bot Welcome to OFSLL Bot!! Kindly Enter a Valid Customer Id Monday, May 3, 2021, 05:44:17 PM true 0000001044 Monday, May 3, 2021, 05:44:19 PM true Please select an account Monday, May 3, 2021, 05:44:21 PM true
Click on **Account Details** to view the Account Summary which consists of the following options: • Today’s Payoff Quote • Last 5 Transaction • Insurance Details • Last Billing Details • Number of Terms remaining?	 OFSLL Bot Issue: ERROR Invalid Account Number/Account not eligible for Advance Disbursement Monday, May 3, 2021, 05:49:24 PM true Account Details Payment Details Communication Preference Limit Details Do the process again? Exit Kindly Type here ...

Table 1-4 (Cont.) BOT Usability Workflow

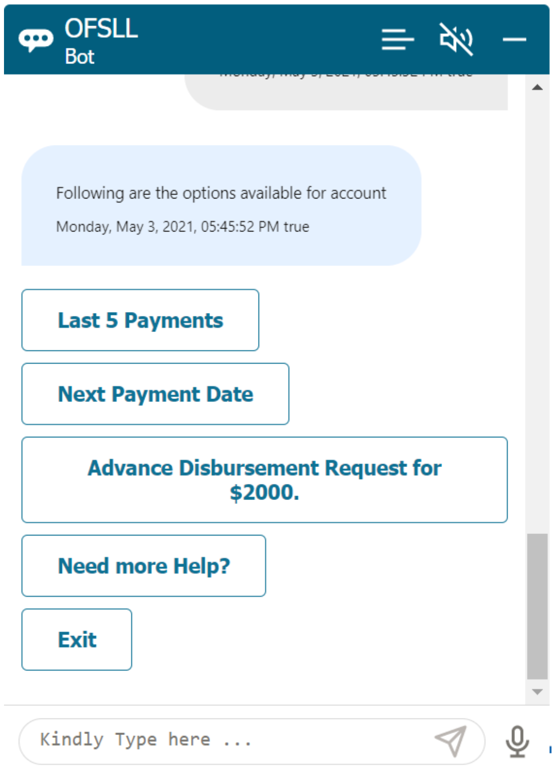
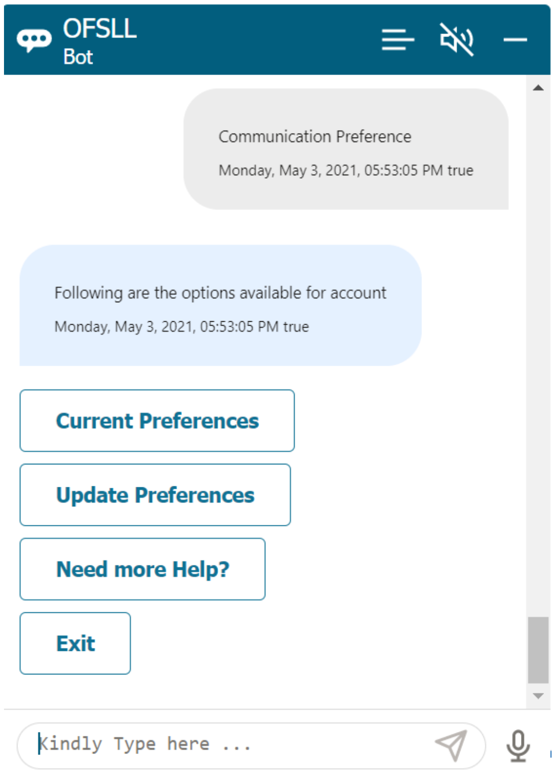
Action	BOT Response
Click on **Payment Details** and view the following information related to the account: • Last 5 Payments • Next Payment Date • Advance Disbursement Request for $2000	 OFSLL Bot Following are the options available for account Monday, May 3, 2021, 05:45:52 PM true Last 5 Payments Next Payment Date Advance Disbursement Request for $2000. Need more Help? Exit Kindly Type here ...
Click on **Communication Preferences** and view the following options: • Current Preferences • Update Preference • Need more help	 OFSLL Bot Communication Preference Monday, May 3, 2021, 05:53:05 PM true Following are the options available for account Monday, May 3, 2021, 05:53:05 PM true Current Preferences Update Preferences Need more Help? Exit Kindly Type here ...

Table 1-4 (Cont.) BOT Usability Workflow

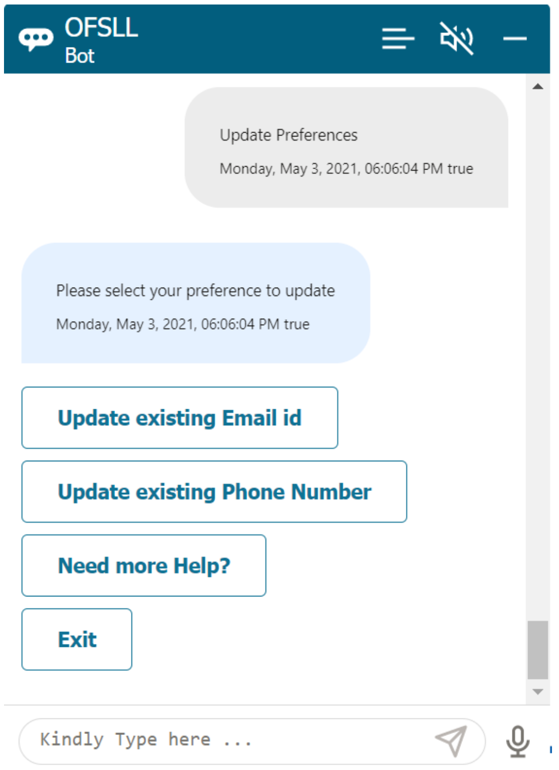
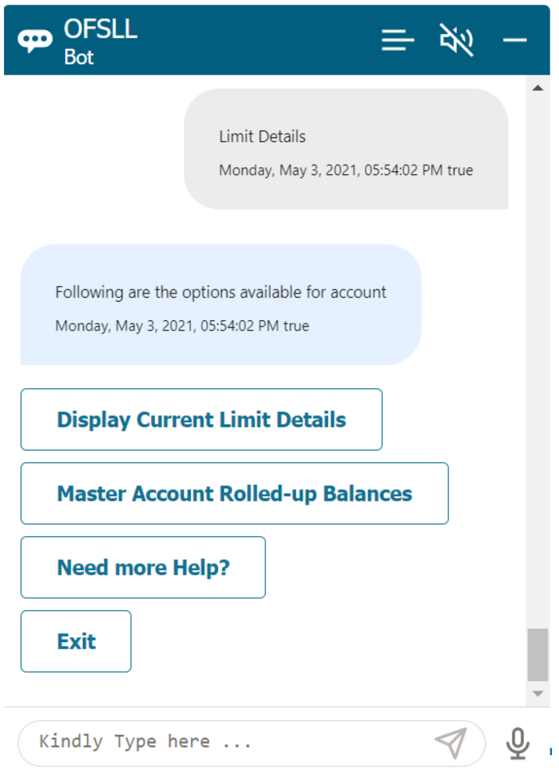
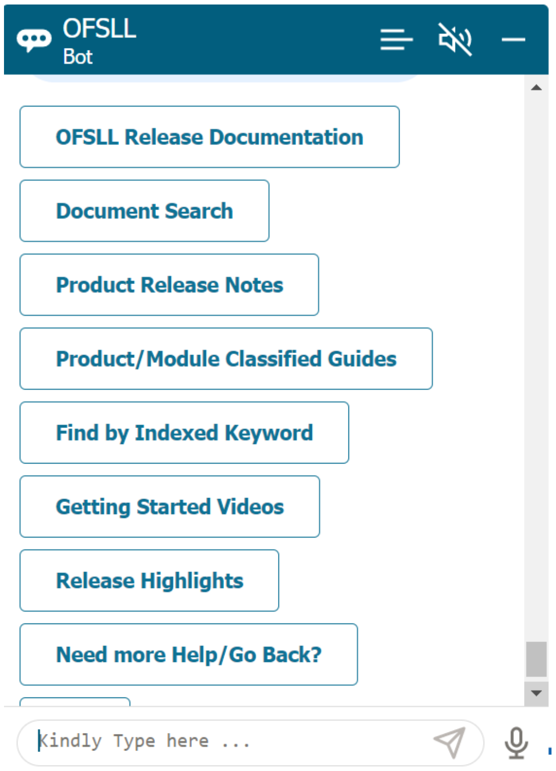
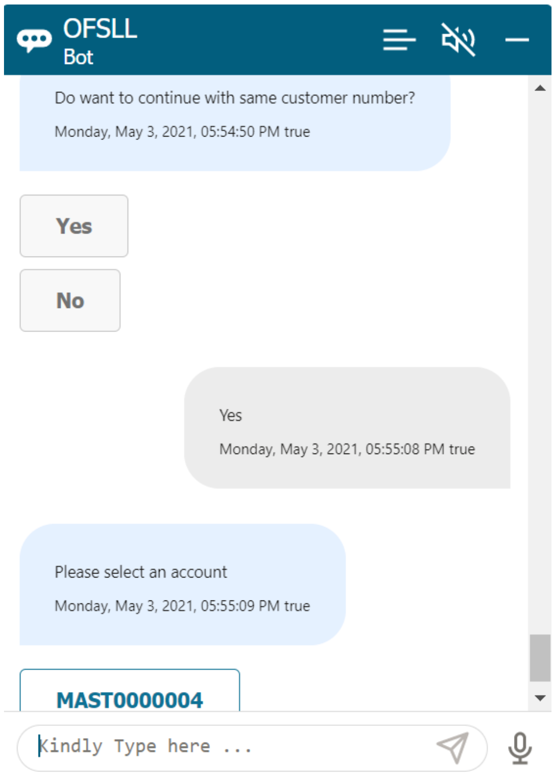
Action	BOT Response
Selecting **Update preference** options allows you to modify the following details by posting appropriate transaction: • Update existing Email id • Update existing Phone Number	 Update Preferences Monday, May 3, 2021, 06:06:04 PM true Please select your preference to update Monday, May 3, 2021, 06:06:04 PM true OFSLL Bot Update existing Email id Update existing Phone Number Need more Help? Exit Kindly Type here ...
Click on **Limit Details** and view the following options: • Display current limit details • Master Account rolled-up Balances	 Limit Details Monday, May 3, 2021, 05:54:02 PM true Following are the options available for account Monday, May 3, 2021, 05:54:02 PM true OFSLL Bot Display Current Limit Details Master Account Rolled-up Balances Need more Help? Exit Kindly Type here ...

Table 1-4 (Cont.) BOT Usability Workflow

Action	BOT Response
Click on the **OFSLL documentation tree** and view the following options: • OFSLL release documentation • Document Search • Product release notes • Product classified guides • Find by indexed keyword • Getting started videos • Release Highlights • Need more help For detailed information on Documentation Bot Usability, refer to **ofsll_docubot_overview_and_ developer_guide** document.	 OFSLL Bot OFSLL Release Documentation Document Search Product Release Notes Product/Module Classified Guides Find by Indexed Keyword Getting Started Videos Release Highlights Need more Help/Go Back? Kindly Type here ...
Click **Need more help** You have the option to continue with the same customer id or enter new customer id.	 Do want to continue with same customer number? Monday, May 3, 2021, 05:54:50 PM true Yes Monday, May 3, 2021, 05:55:08 PM true Please select an account Monday, May 3, 2021, 05:55:09 PM true

2

Developer Guide for BOT Customization

This section of the document intends to help you to set up and configure Oracle Digital Assistant (ODA) **ASK** with the sample OFSLL wrapper. However, the instructions are provided in brief and for any additional information, contact Oracle Financial Services Lending and Leasing Product Engineering team.

> **Note:**
>
> Currently this framework supports basic authentication provided by OFSLL REST service. OAUTH authentication is not supported. Additionally, OBDX (Oracle Banking Digital Experience) can be integrated for user authentication purpose. For more information, refer to documentation at https://docs.oracle.com/cd/E97825_01/webhelp/Content/obdx/core/authentn/authntctn.htm

This topic consists of the following sections:

- Pre-requisites
- OFSLL Wrapper customization
- ODA – Dialog Flow Development
- Deploying war file on WebLogic Server
- Web application UI for Accessing BOT
- App configuration for enabling chatbot
- BOT Configuration

2.1 Pre-requisites

Following are the mandatory pre-requisites:

- OFSLL being a back-office system with limited capability, the following external components are to be integrated in a single framework:
  - ODA or Oracle Digital Assistant is a platform that allows to create and deploy digital assistants, which are AI-driven interfaces that help users accomplish a variety of tasks in natural language conversations.
  - OBDX or Oracle Banking Digital Experience as a Application Launching portal and for multi-factor authentication.
    --or--
  - Any 3rd party web application or customer self-service portal or lenders/financial services website to launch OFSLL BOT. In this case user authentication related integration needs to be handled as part of the implementation activity.
- Users need to have a capability to develop customized workflows using ODA development framework. A brief introduction is explained in ODA – Dialog Flow Development section.

- User need to have a good understanding of OFSLL REST services and should be able to customize it accordingly.
- User needs to be well versed with OFSLL wrapper customization as explained in OFSLL Wrapper customization section.

2.2 OFSLL Wrapper customization

> **Note:**
>
> Note: From the current release onwards, no additional jar file needs to be added since `Maven – Pom.xml` based model has been implemented.

Follow the below steps for OFSLL wrapper customization:

1. Import project into eclipse and modify channel.Properties to update below properties.

```
ofsll.baseURL = <OFSLL REST service base URL
<http://<host>:<port>/OfsllRestWS/service/api/resources>>
ofsll.username = <OFSLL username>
ofsll.password = <OFSLL pass>
ofsll.suffix = htm
ofsll.otmHttpUrl=https://docs.oracle.com/cd/
ofsll.fIndex=/findex.htm
ofsll.index=index.htm
ofsll.video=/videos.htm
ofsll.ofsllReleaseNotes=/pdf/refdocs/ofsll_release_notes.pdf
ofsll.ofsllReleaseDoc=https://docs.oracle.com/en/industries/financial-
services/financial-lending-leasing/index.html
ofsll.splitSeperator==
ofsll.maxHitsResults=<max number of results returned>
ofsll.indexDir = <Release index directory path of server >
ofsll.releaseVersionUrl= <Release Part number>
ofsll.releaseNo=<Release No>
ofsll.releaseHighlights=/pdf/refdocs/release_highlights.htm
```

2. To add any new service modify com.ofss.ofsll.chatbot.restclient.ChatRestClient.java file.
   - Inside ChatRestClient Class add a new method with required actions
   - Add supporting JAXB files
   - Use the available supporting methods -- readInputStream,setChatBotResponse, createConnection, stringToJaxb etc.

   Example for document search functionality is indicated below:

```
@Consumes(MediaType.APPLICATION_JSON)
    @Produces(MediaType.APPLICATION_JSON)
    @POST
    @Path("/lucenesearch")
    public Response lucenesearch(ODARequestDTO ibcsRequest) throws
IOException {
    final IChatbotAssembler chatbotAssembler =
ChatbotAssemblerFactory.getInstance().getChatbotAssembler("ODA");
```

```
    final HashMap < String,
    Object > map = (HashMap < String, Object > )
ibcsRequest.getProperties();
    String searchQuery = "";
    Properties prop = new Properties();
    try (InputStream propertiesFile =
this.getClass().getClassLoader().getResourceAsStream("channel.properties"))
 {
prop.load(propertiesFile);
    }
    if (map != null && map.containsKey("query")) {
    searchQuery = (String) map.get("query");
    }
    ResponseDTO ibcsResponse = null;
    try {
    ChatbotResponseDTO chatbotResponse = new ChatbotResponseDTO();
    String indexDirPath = prop.getProperty("ofsll.indexDir")
+prop.getProperty("ofsll.releaseNo");
    String releaseVersionUrl = prop.getProperty("ofsll.releaseVersionUrl");
    String urlPrefix = prop.getProperty("ofsll.otmHttpUrl");
    String splitSeperator = prop.getProperty("ofsll.splitSeperator");
    String releaseNo = prop.getProperty("ofsll.releaseNo");
    String urlPrefixPath = urlPrefix + releaseVersionUrl;
    String findexPath = prop.getProperty("ofsll.fIndex");
    String indexPath = prop.getProperty("ofsll.index");
    String videoPath = prop.getProperty("ofsll.video");
    String ofsllReleaseNotesPath =
prop.getProperty("ofsll.ofsllReleaseNotes");
    String ofsllReleaseDocPath = prop.getProperty("ofsll.ofsllReleaseDoc");
    Integer maxHitsResults =
Integer.parseInt(prop.getProperty("ofsll.maxHitsResults"));
    File fileIndexDirPath = new File(indexDirPath);
    LuceneSearchHighlighter luceneSearchHighlighter = new
LuceneSearchHighlighter();
    List<String> fileList = new ArrayList <> ();
    if ((searchQuery.toLowerCase().trim().contains("#ofsll release
document")) ||
(searchQuery.toLowerCase().trim().contains("navigate to index page")) ||
(searchQuery.toLowerCase().trim().contains("#video gallery")) ||
(searchQuery.toLowerCase().trim().contains("#ofsll release notes")) ||
(searchQuery.toLowerCase().trim().contains("#index page"))) {
    if ((searchQuery.toLowerCase().trim().contains("#ofsll release
document"))) {
    releaseNo="All Release Version";
    fileList.add(searchQuery + splitSeperator + ofsllReleaseDocPath +
splitSeperator+searchQuery+ splitSeperator+releaseNo);
    }
    if ((searchQuery.toLowerCase().trim().contains("navigate to index
page"))) {
    fileList.add(searchQuery + splitSeperator + urlPrefixPath + findexPath
+ splitSeperator+searchQuery+ splitSeperator+releaseNo);
    }
    if ((searchQuery.toLowerCase().trim().contains("#index page"))) {
    searchQuery = indexPath;
    fileList = luceneSearchHighlighter.searchsinglepage(fileIndexDirPath,
searchQuery, maxHitsResults, splitSeperator);
```

```
    }
    if ((searchQuery.toLowerCase().trim().contains("#video gallery"))) {
    fileList.add(searchQuery + splitSeperator + urlPrefixPath + videoPath
+ splitSeperator+searchQuery+ splitSeperator+releaseNo);
    }
    if ((searchQuery.toLowerCase().trim().contains("#ofsll release
notes"))) {
    fileList.add(searchQuery + splitSeperator + urlPrefixPath +
ofsllReleaseNotesPath + splitSeperator+searchQuery+
splitSeperator+releaseNo);
    }
    } else {
    searchQuery = searchQuery.replaceAll("#", "");
    fileList = luceneSearchHighlighter.search(fileIndexDirPath,
searchQuery, maxHitsResults, splitSeperator);
    }
    String serviceOutputForChatBot = "";
    for (String obj: fileList) {
    if (serviceOutputForChatBot == "") {
    serviceOutputForChatBot = obj.replace("\\", "/");
    } else {
    serviceOutputForChatBot = serviceOutputForChatBot + "\n---\n" +
obj.replace("\\", "/");
    }
    }
    if (fileList.isEmpty()) {
    String errorOutputForChatBot = "Search is not found for : " +
searchQuery;
    setChatBotResponse("failure", errorOutputForChatBot, chatbotResponse,
"response", "request");
    } else {
    List < String > srhchoices = new ArrayList < >();
    for (String obj: fileList) {
    srhchoices.add(obj.replace("\\", "/"));
    }
    setChatBotResponse("success", srhchoices, chatbotResponse, "acc_srh",
"acc_srh");
    }
    ibcsResponse = chatbotAssembler.fromChatbotResponseDTO((RequestDTO)
ibcsRequest, chatbotResponse);
    } catch(Exception e) {
    LOGGER.log(Level.SEVERE, e.getMessage());
    }
    return Response.status(Response.Status.OK).entity((Object)
this.buildResponse((Object) ibcsResponse)).build();
}
```

Example for lastbillingdetails Service -- This uses Account Details Service

```
@Consumes(MediaType.APPLICATION_JSON)@Produces(MediaType.APPLICATION_JSON)@
POST@Path("/lastbillingdetails")
public Response lastbillingDetails(ODARequestDTO ibcsRequest) {
final IChatbotAssembler chatbotAssembler =
ChatbotAssemblerFactory.getInstance().getChatbotAssembler("ODA");
final HashMap < String,
Object > map = (HashMap < String, Object > ) ibcsRequest.getProperties();
String accountNumber = "";
if (map != null && map.containsKey("acc_nbr")) {
accountNumber = (String) map.get("acc_nbr");
}
ResponseDTO ibcsResponse = null;
try {
ChatbotResponseDTO chatbotResponse = new ChatbotResponseDTO();
String requestURL = "/servicing/account/" + accountNumber + "?
displayassociateaccounts=N";
HttpURLConnection conn = createConnection("GET", requestURL, "");
if (conn.getResponseCode() != 200 && conn.getResponseCode() != 201 &&
conn.getResponseCode() != 202) {
String errorOutput = readInputStream(conn, "error");
AccountDetailResponseType accountsDetails =
stringToJaxb(AccountDetailResponseType.class, errorOutput);
String errorOutputForChatBot = "\nBilling Details: \n " +
accountsDetails.getResult().getStatus().toString() + "\n" +
accountsDetails.getResult().getStatusDetails();
setChatBotResponse("failure", errorOutputForChatBot, chatbotResponse,
"response", "request");
} else {
String serviceOutput = readInputStream(conn, "input");
AccountDetailResponseType accountsDetails =
stringToJaxb(AccountDetailResponseType.class, serviceOutput);
String serviceOutputForChatBot = "\nBilling Details: " + "\n Generation
Date: " +
dateFormater(accountsDetails.getAccountDetailSummary().get(0).getStatementD
etails().get(0).getGenerationDate()) + "\n Closing Date: " +
dateFormater(accountsDetails.getAccountDetailSummary().get(0).getStatementD
etails().get(0).getClosingDate()) + "\n Due Date: " +
dateFormater(accountsDetails.getAccountDetailSummary().get(0).getStatementD
etails().get(0).getDueDate()) + "\n Current Due Amount: " +
accountsDetails.getAccountDetailSummary().get(0).getStatementDetails().get(
0).getCurrentDueAmount();
setChatBotResponse("success", serviceOutputForChatBot, chatbotResponse,
"response", "request");
}
ibcsResponse = chatbotAssembler.fromChatbotResponseDTO((RequestDTO)
ibcsRequest, chatbotResponse);
} catch(Exception e) {
LOGGER.log(Level.SEVERE, "Error: ", e);
}
return Response.status(Response.Status.OK).entity((Object)
this.buildResponse((Object) ibcsResponse)).build();
}
```

3. Export project as war (OracleFSLLChatBot.war) file.
4. Deploy `<WL_Home>/wlserver/common/deployable-libraries/jax-rs-2.0.war` as Library on weblogic.
5. Deploy generated WAR (`OracleFSLLChatBot.war`) in step 3 onto weblogic server.
6. Note down base service URL that is required while publishing in ODA.

   Example: http://<host>:<port>/ofsll/v1/fulfillment

2.3 ODA – Dialog Flow Development

Each menu option displayed in BOT are configured as an **Intent** which is configured to perform a specific function or otherwise call a REST service in OFSLL.

In-order to achieve a sequence of menu options, dialog flow development is required to be performed in ODA Oracle Digital Assistant. Following is a quick overview of steps involved:

- Login
- Creating Skill / Digital Assistant
- Defining Entity
- Adding Intents
- Updating Bot flow using Yaml
- Adding OFSLL REST service
- Configuring Channel for Publishing
- Publishing

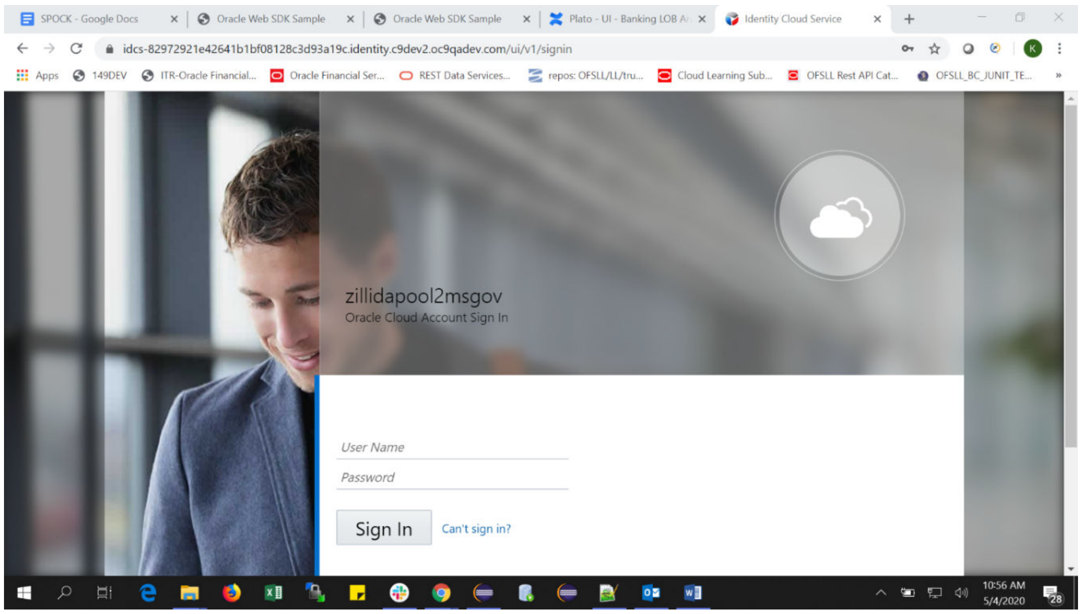
It is recommended to refer to ODA documentation for detailed information - https://docs.oracle.com/en/cloud/paas/digital-assistant/index.html

In the ODA - dialog flow development, you can either create new or import the given sample available in path – `<release.zip>\LL\release\14_x_0_0_0\ws_as\ChatBot\transaction-bot`

The sequence of flow in creating a sample BOT in ODA is indicated below with illustration:

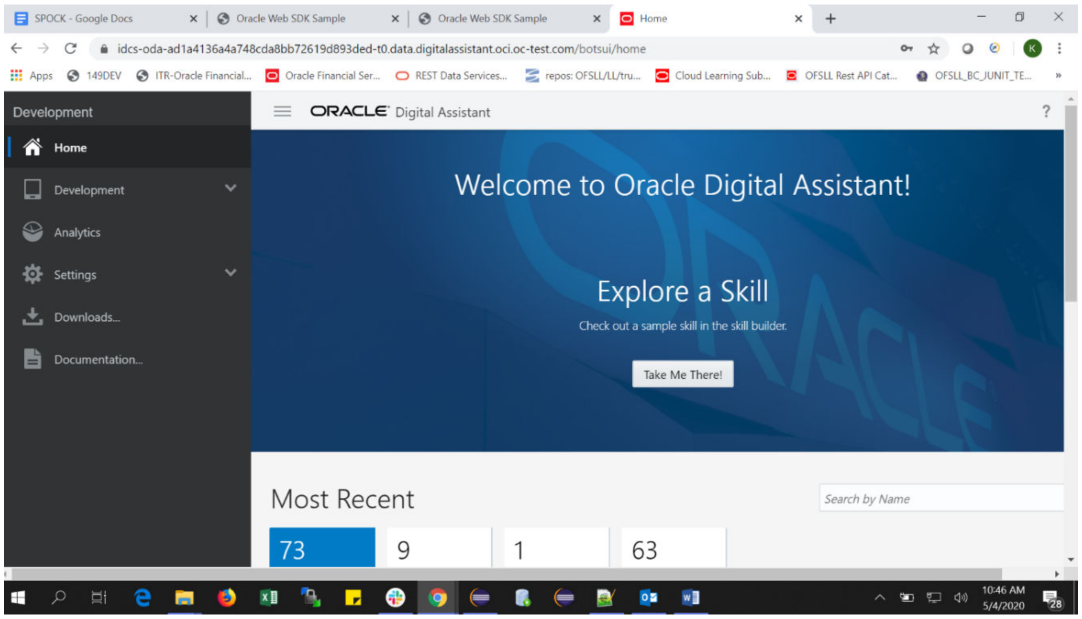
1. Login to ODA UI

Figure 2-1 ODA UI - Login page



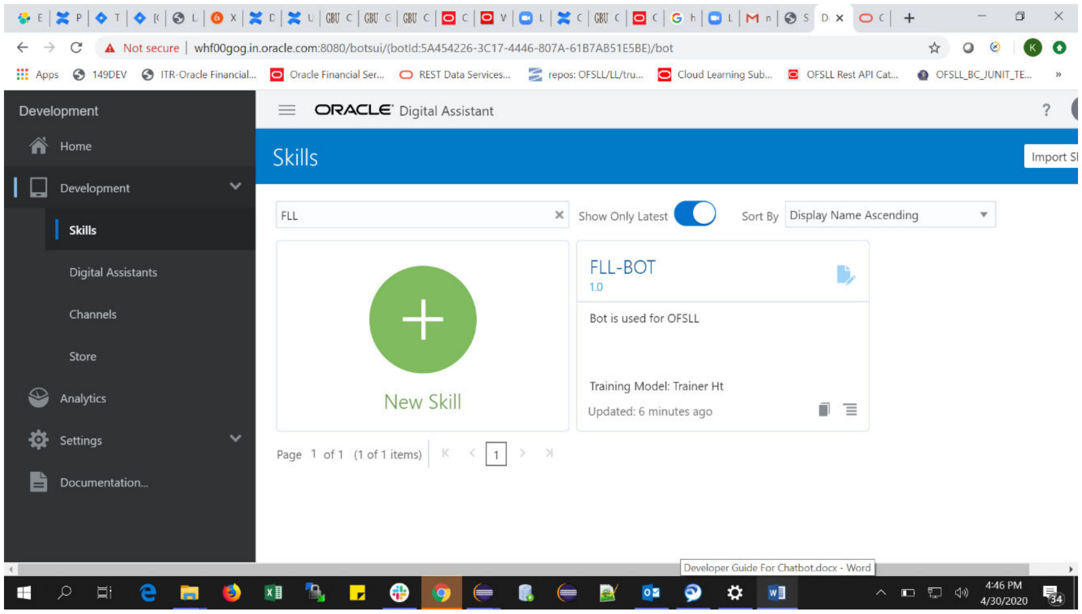
2. Go to Home

Figure 2-2 ODA UI - Home


idcs-oda-ad1a4136a4a748cda8bb72619d893ded-t0.data.digitalassistant.oci.oc-test.com/botsui/home
Development
Home
Development
Analytics
Settings
Downloads...
Documentation...
ORACLE Digital Assistant
Welcome to Oracle Digital Assistant!
Explore a Skill
Check out a sample skill in the skill builder.
Take Me There!
Most Recent
Search by Name
73
9
1
63

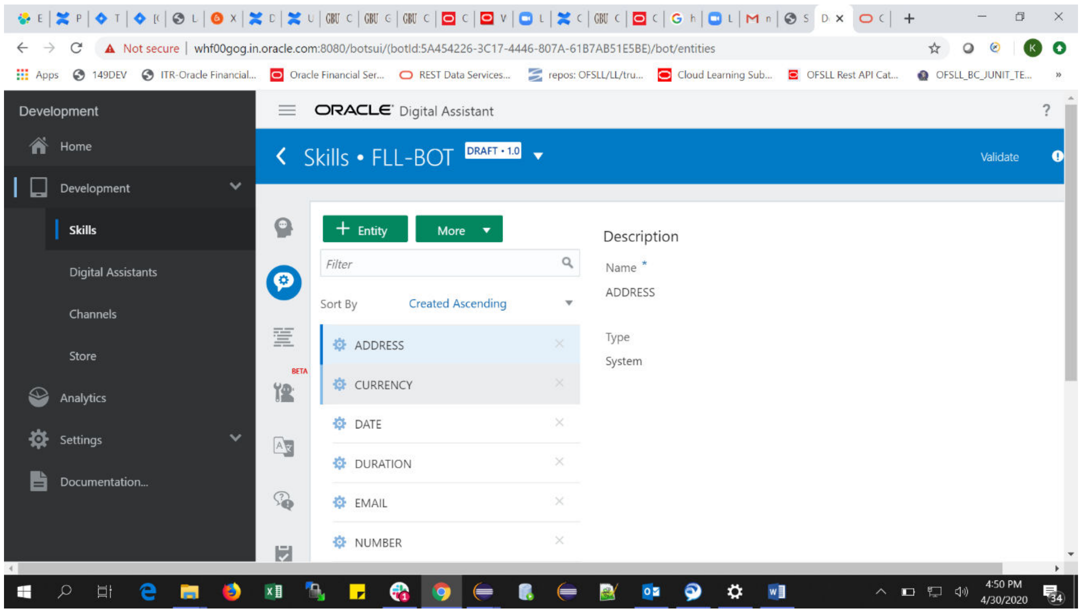
3. Create Skill/Digital Assistant.

Figure 2-3 Skill Assistant



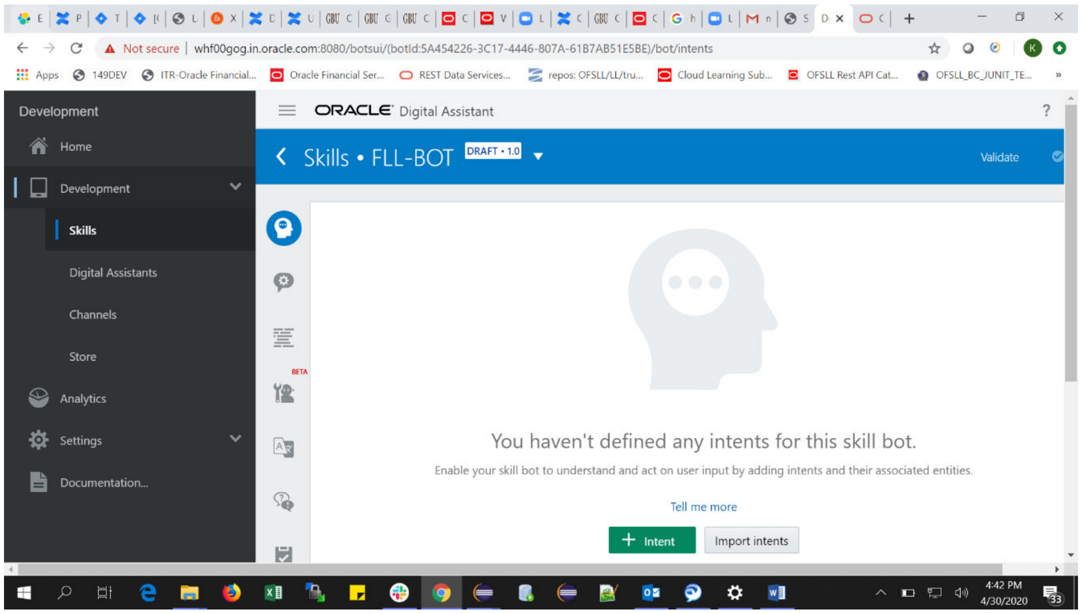
4. Add Entities

Figure 2-4 Add Entities



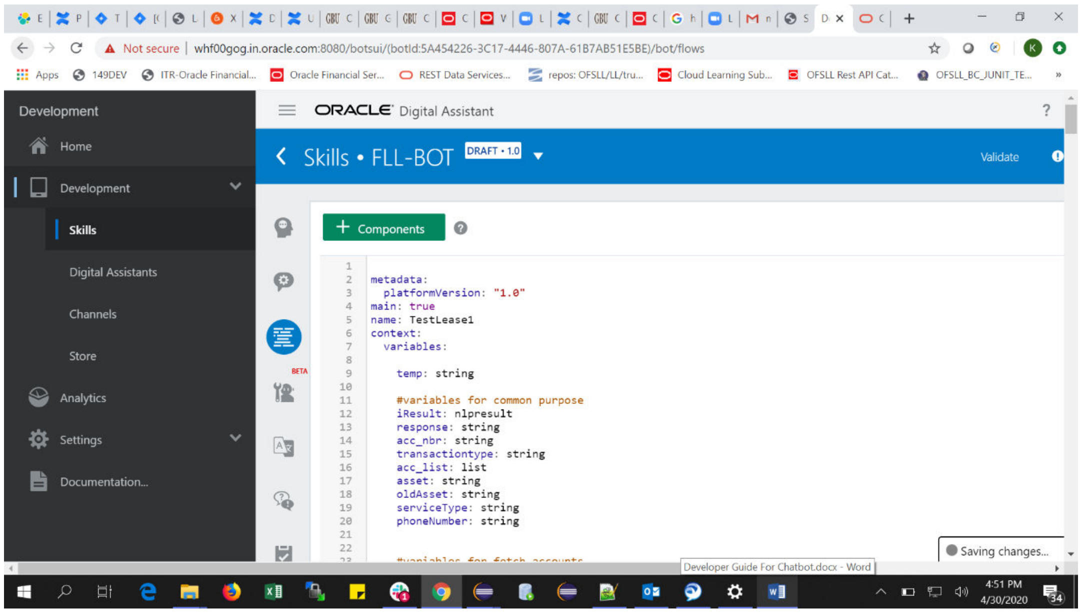
5. Add Intents. This involves defining Activity, Available option, Next level, Breakpoint, intermediate steps.

Figure 2-5 Add Intents


Not secure | whf00gog.in.oracle.com:8080/botsui/(botId:5A454226-3C17-4446-807A-61B7AB51E5BE)/bot/intents
Development
Home
Development
Skills
Digital Assistants
Channels
Store
Analytics
Settings
Documentation...
ORACLE Digital Assistant
Skills • FLL-BOT DRAFT • 1.0
Validate
You haven't defined any intents for this skill bot.
Enable your skill bot to understand and act on user input by adding intents and their associated entities.
Tell me more
Intent
Import intents

6. Add Bot flow using Yaml

Figure 2-6   Add Bot flow



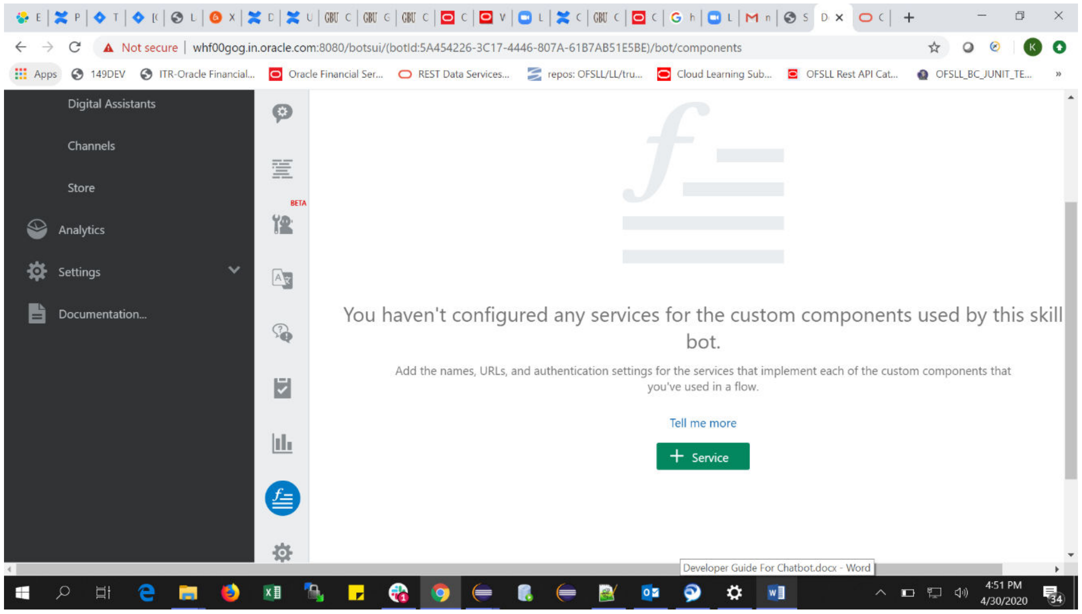
```
metadata:
  platformVersion: "1.0"
main: true
name: TestLease1
context:
  variables:

    temp: string

    #variables for common purpose
    iResult: nlpresult
    response: string
    acc_nbr: string
    transactiontype: string
    acc_list: list
    asset: string
    oldAsset: string
    serviceType: string
    phoneNumber: string
```

7. Add OFSLL REST Service

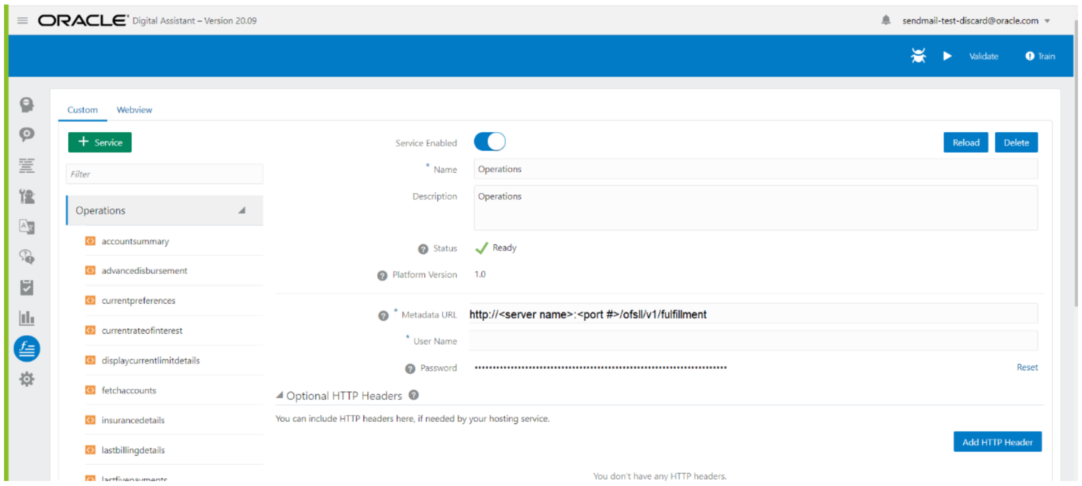
Figure 2-7   OFSLL REST Service


Not secure | whf00gog.in.oracle.com:8080/botsui/(botId:5A454226-3C17-4446-807A-61B7AB51E5BE)/bot/components
Digital Assistants
Channels
Store
Analytics
Settings
Documentation...
Tell me more
Service

You haven't configured any services for the custom components used by this skill bot.

Add the names, URLs, and authentication settings for the services that implement each of the custom components that you've used in a flow.

Figure 2-8 REST Service

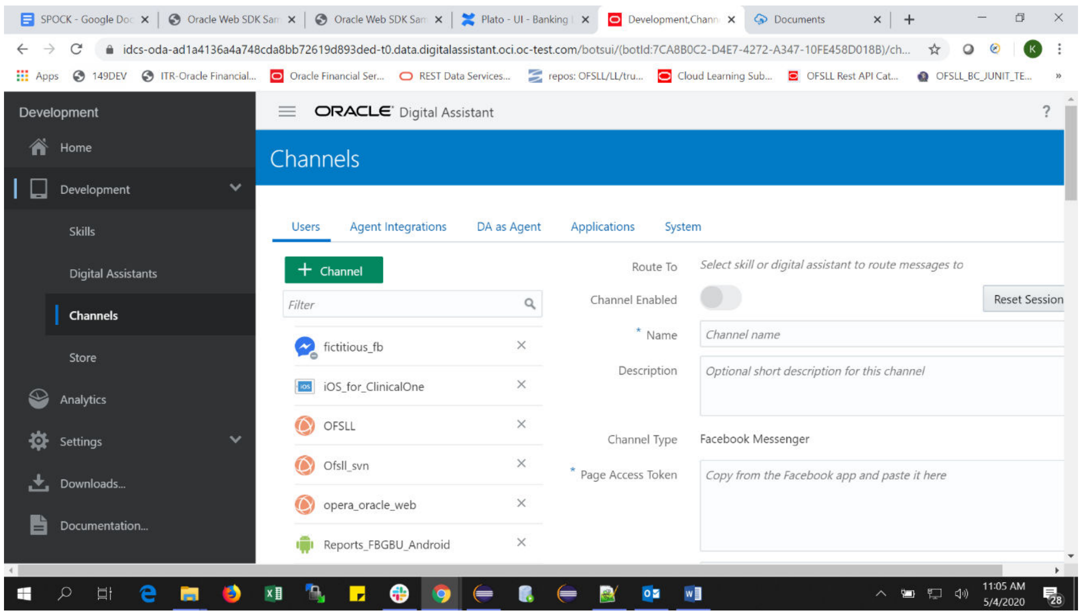

ORACLE Digital Assistant – Version 20.09
sendmail-test-discard@oracle.com
Validate
Train
Custom
Webview
Service
Filter
Reload
Delete

Service Enabled: on
* Name: Operations
Description: Operations
Status: Ready
Platform Version: 1.0
* Metadata URL: http://<server name>:<port #>/ofsll/v1/fulfillment
* User Name
Password: ••• Reset
Optional HTTP Headers
You can include HTTP headers here, if needed by your hosting service.
Add HTTP Header
You don't have any HTTP headers.

Operations: accountsummary, advancedisbursement, currentpreferences, currentrateofinterest, displaycurrentlimitdetails, fetchaccounts, insurancedetails, lastbillingdetails, lastfivepayments

8. Add Channel. This indicates where it has to be published and in this sample application, only web channel is supported.
9. Enter the published URL as generated in step 2.6

Figure 2-9 Channels



Channels: fictitious_fb, iOS_for_ClinicalOne, OFSLL, Ofsll_svn, opera_oracle_web, Reports_FBGBU_Android

Route To: Select skill or digital assistant to route messages to
Channel Enabled: off
* Name: Channel name
Description: Optional short description for this channel
Channel Type: Facebook Messenger
* Page Access Token: Copy from the Facebook app and paste it here

10. After completion of Skill, publish. On publishing, the draft is converted to final non-editable version and only final published version is accessible in bot.
11. Additional security layer is available to allow chatbot to work for specific registered domains. To do so, select the channel, navigate to **Allowed Domains** and add the domain name in the field. For example, adding *in.company1.com* allows chatbot to work only from company1 domain.
12. There is also an option to define the session time-out for chatbot which by default is set to maximum of 1440 minutes. You can enter the required time in minutes.

Note:

The **ofsll-transaction-bot** is the sample ODA FLL application designed for the demo purpose. The same can be imported in any ODA environment tested, modified for new features.

2.4 Deploying war file on WebLogic Server

Before you begin, ensure to use the war file for deployment of OFSLL BOT available in the path – `release\<14_x.0.0.0>\ws_as\ChatBot\OracleFSLLChatBot.war`.

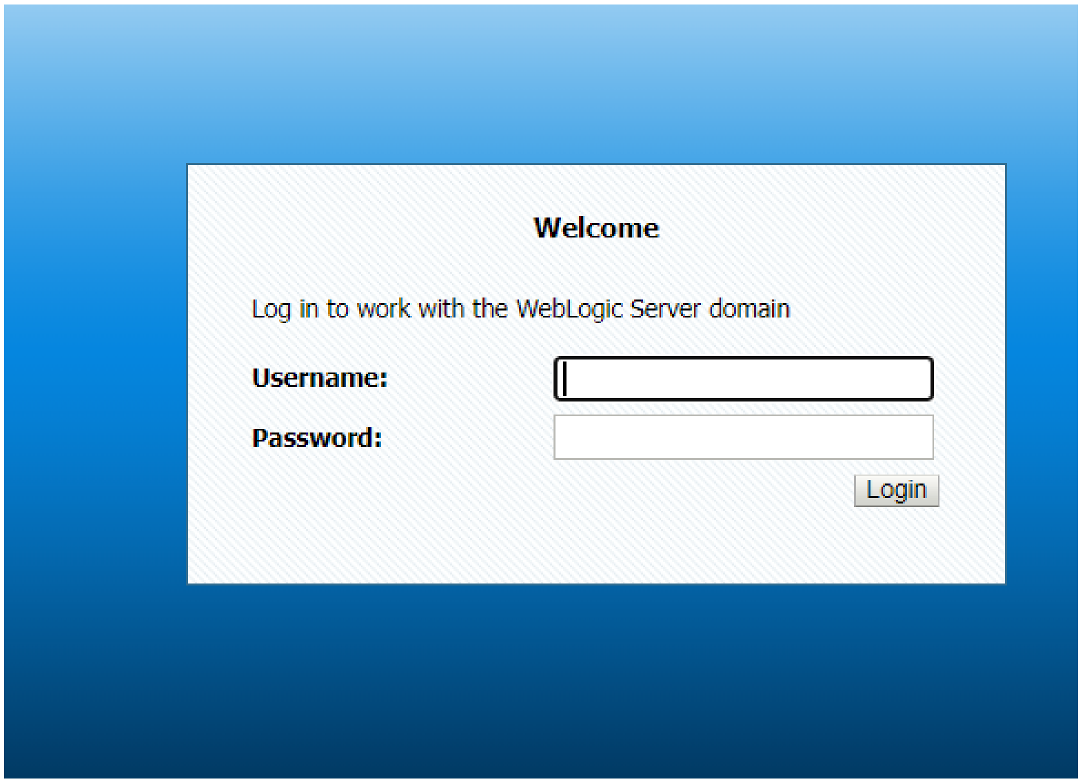
1. Login to Web Logic application server enterprise manager (e.g.:http://hostname:port/em).

   For example, **http://host01.example.com:8001/console**

   Note:

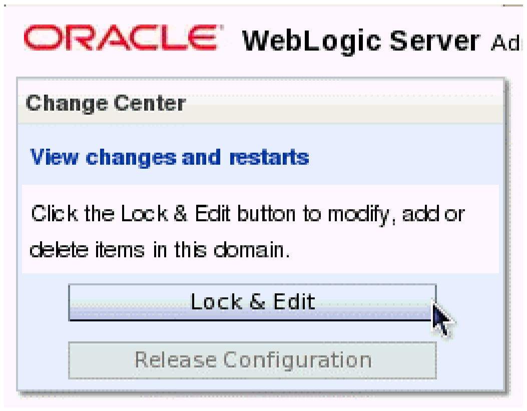
   Use the host name and port of the administration server of your domain.

   Figure 2-10 Web Logic application server - Login

   
   Welcome
   Log in to work with the WebLogic Server domain

2. Enter valid login credentials.
3. Deploying an application is a change to the domain's configuration, so it must first be locked. In the Change Center. Click **Lock & Edit**.

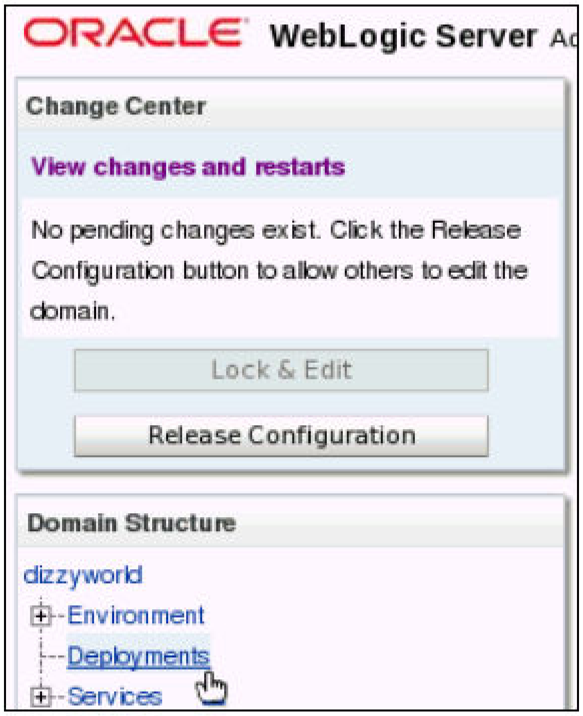
Figure 2-11   WebLogic Server - Change center



Click the Lock & Edit button to modify, add or delete items in this domain.

4. Under Domain Structure, click **Deployments**.

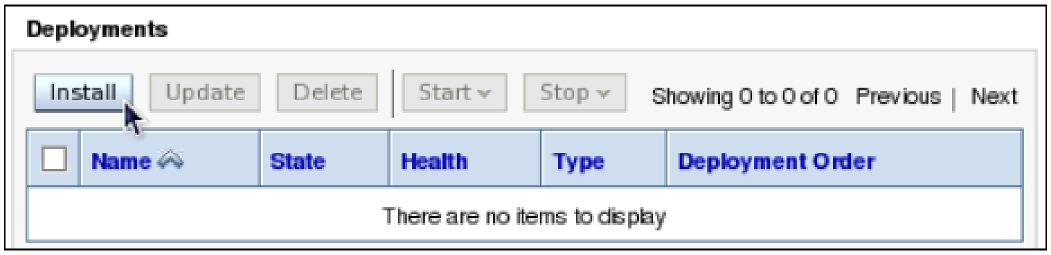
Figure 2-12   Deployments



No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

5. On the right, under Deployments, click **Install**.

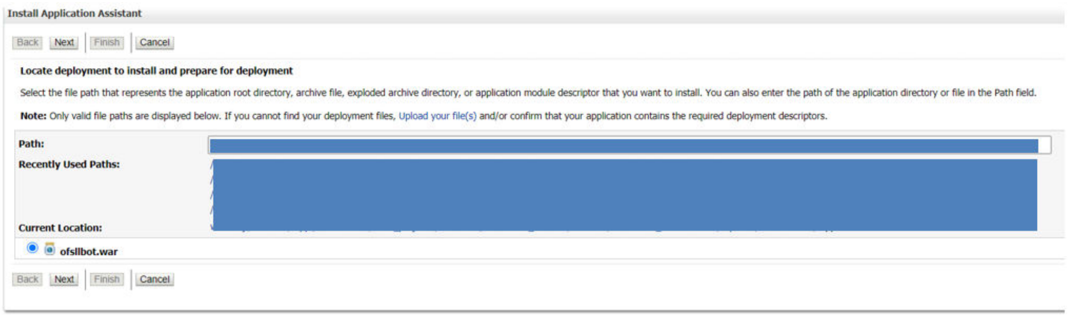
Figure 2-13   Install



☐	Name	State	Health	Type	Deployment Order
There are no items to display					

6. Find the Current Location field. Use the links to browse to the location in which you placed the downloaded `OracleFSLLChatBot.war` file.
7. The `.war` file is available in the path - `release\<14_x.0.0.0>\ws_as\ChatBot\OracleFSLLChatBot.war`. Select the `.war` file from the given path and click the radio button next to it. Using the links and the radio button, the console auto populates the Path fields. Alternatively, you can type in the path and file name in the Path field yourself. Click **Next**.

Figure 2-14 Install Application Assistant 1



Install Application Assistant

Locate deployment to install and prepare for deployment

Select the file path that represents the application root directory, archive file, exploded archive directory, or application module descriptor that you want to install. You can also enter the path of the application directory or file in the Path field.

Note: Only valid file paths are displayed below. If you cannot find your deployment files, Upload your file(s) and/or confirm that your application contains the required deployment descriptors.

Path: [redacted]

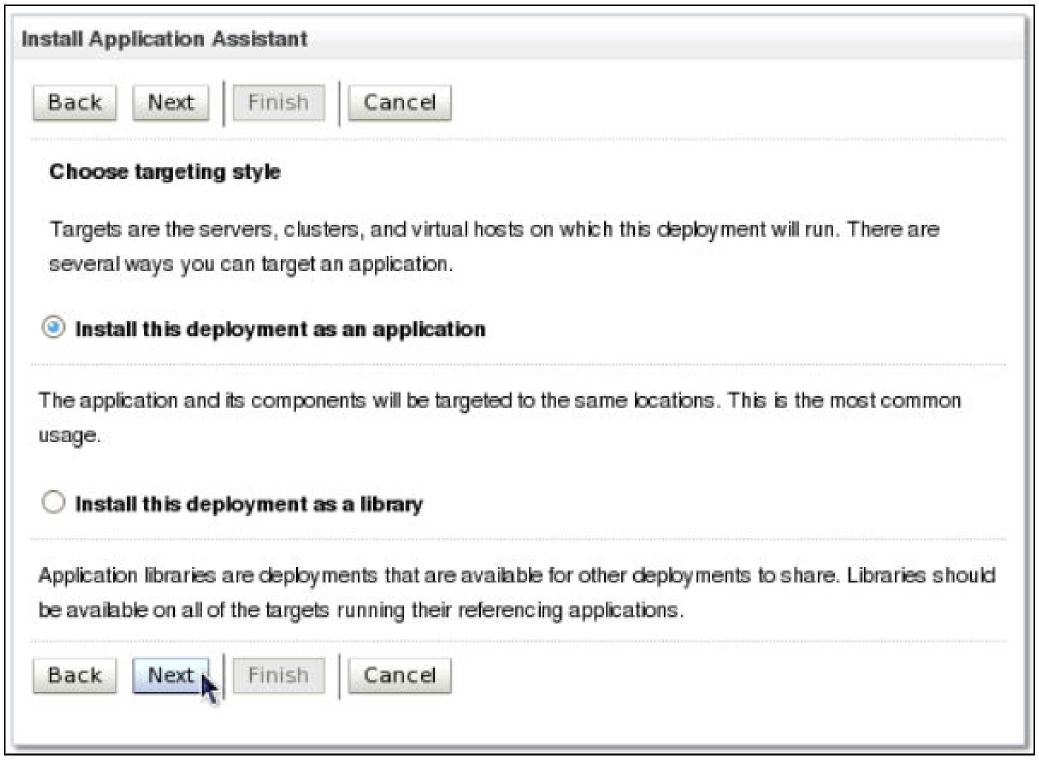
Recently Used Paths: [redacted]

Current Location:

● ofsllbot.war

8. Ensure that **Install this deployment as an application** option is selected. Click **Next**.

Figure 2-15 Install Application Assistant 2



Install Application Assistant

Choose targeting style

Targets are the servers, clusters, and virtual hosts on which this deployment will run. There are several ways you can target an application.

● Install this deployment as an application

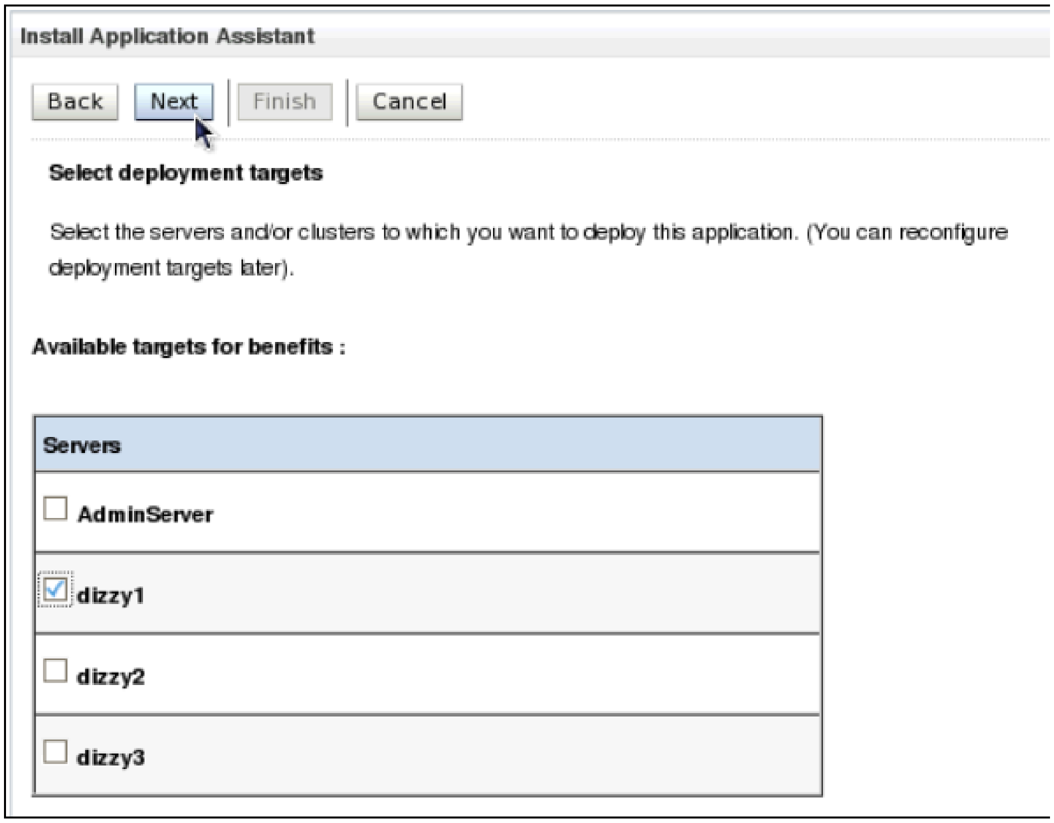
The application and its components will be targeted to the same locations. This is the most common usage.

○ Install this deployment as a library

Application libraries are deployments that are available for other deployments to share. Libraries should be available on all of the targets running their referencing applications.

9. In the below window, click **Next**.

Figure 2-16 Install Application Assistant 3



Select deployment targets

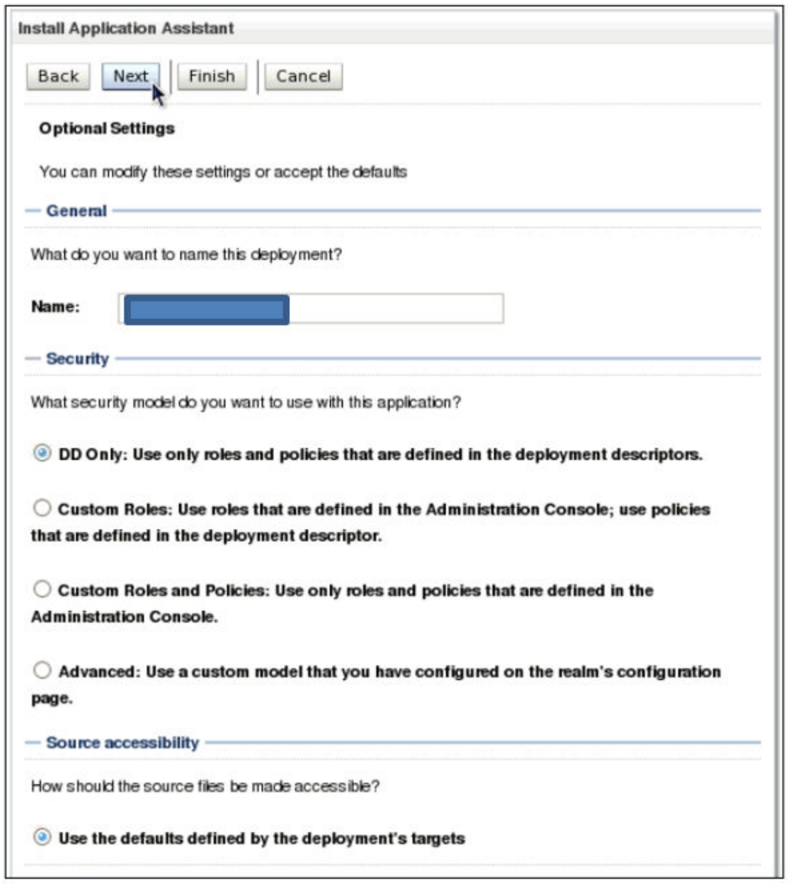
Select the servers and/or clusters to which you want to deploy this application. (You can reconfigure deployment targets later).

Available targets for benefits :

Servers
☐ AdminServer
✓ dizzy1
☐ dizzy2
☐ dizzy3

10. Retain the default values and click **Next**.

Figure 2-17   Install Application Assistant 4



Optional Settings

You can modify these settings or accept the defaults

General

What do you want to name this deployment?

Name: [redacted]

Security

What security model do you want to use with this application?

[✓ **DD Only: Use only roles and policies that are defined in the deployment descriptors.**]

Custom Roles: Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor.

Custom Roles and Policies: Use only roles and policies that are defined in the Administration Console.

Advanced: Use a custom model that you have configured on the realm's configuration page.

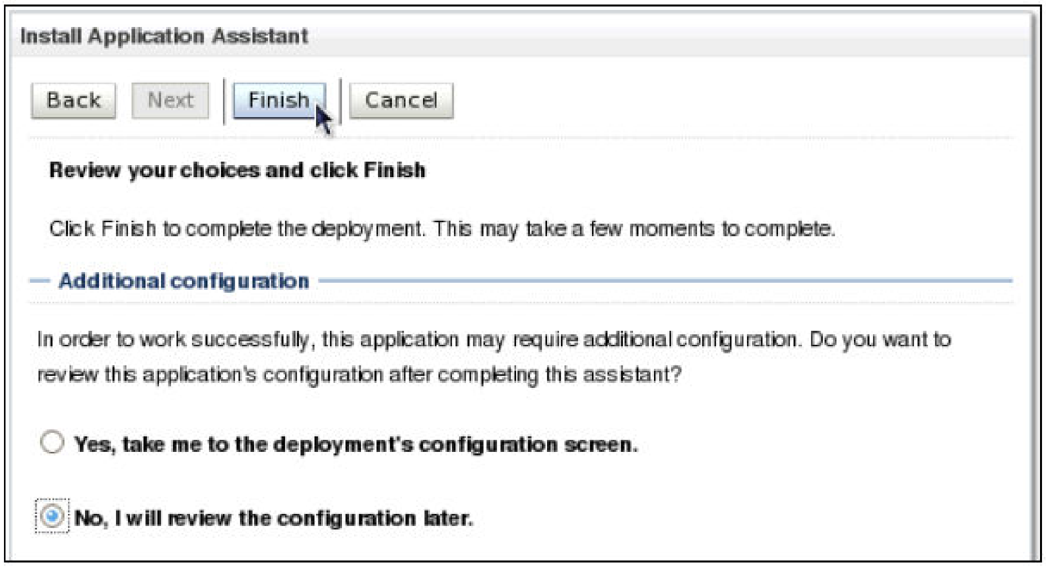
Source accessibility

How should the source files be made accessible?

[✓ **Use the defaults defined by the deployment's targets**]

11. In the below window, select the option **No, I will review the configuration later** and click **Finish**.

Figure 2-18   Install Application Assistant 5



Review your choices and click Finish

Click Finish to complete the deployment. This may take a few moments to complete.

Additional configuration

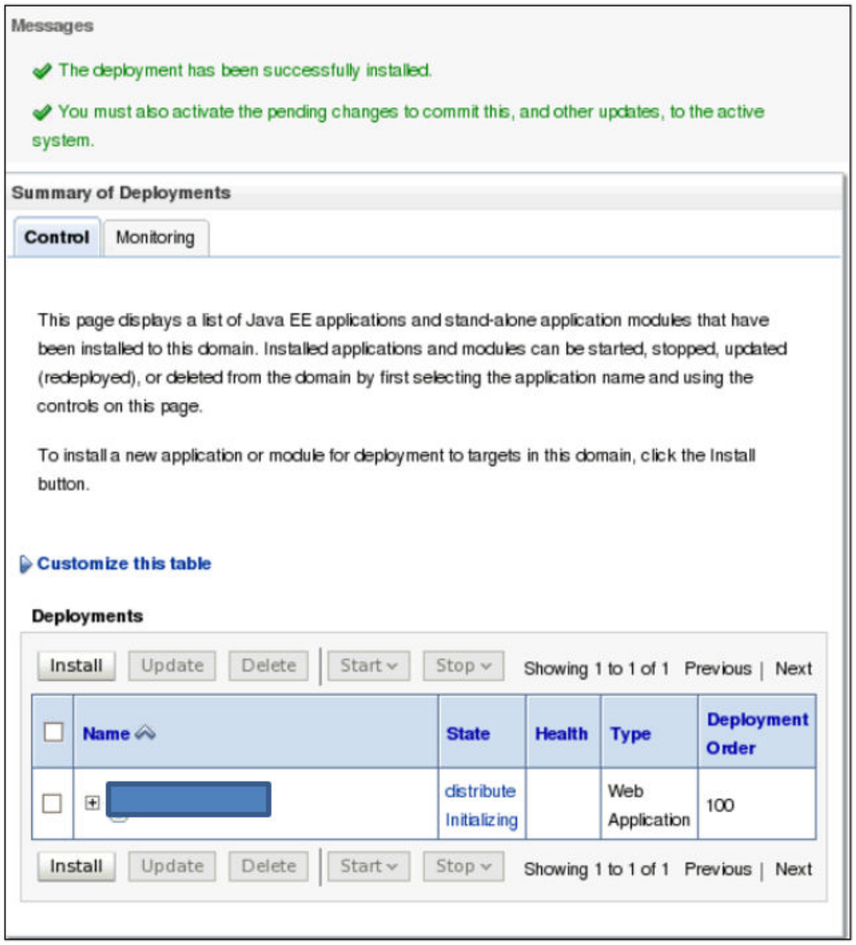
In order to work successfully, this application may require additional configuration. Do you want to review this application's configuration after completing this assistant?

Yes, take me to the deployment's configuration screen.

[✓ **No, I will review the configuration later.**]

Once done view the messages indicating that the deployment was installed, but changes must be activated. In addition, notice the benefits application listed in the Deployments table.

Figure 2-19 Installed Deployment - message



Messages

✓ The deployment has been successfully installed.

✓ You must also activate the pending changes to commit this, and other updates, to the active system.

Summary of Deployments

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

Deployments

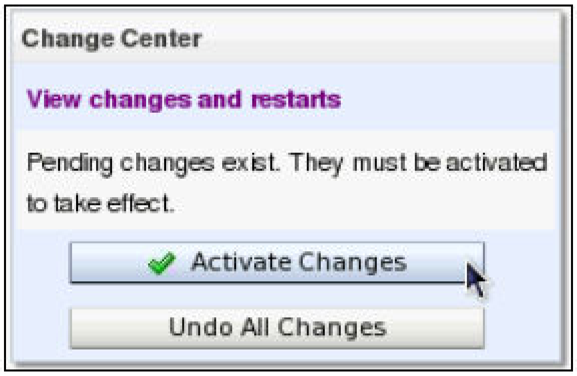
Showing 1 to 1 of 1 Previous | Next

	Name	State	Health	Type	Deployment Order
☐	[redacted]	distribute Initializing		Web Application	100

Showing 1 to 1 of 1 Previous | Next

12. In the Change Center, click the **Activate Changes** button.

Figure 2-20 Activate Changes

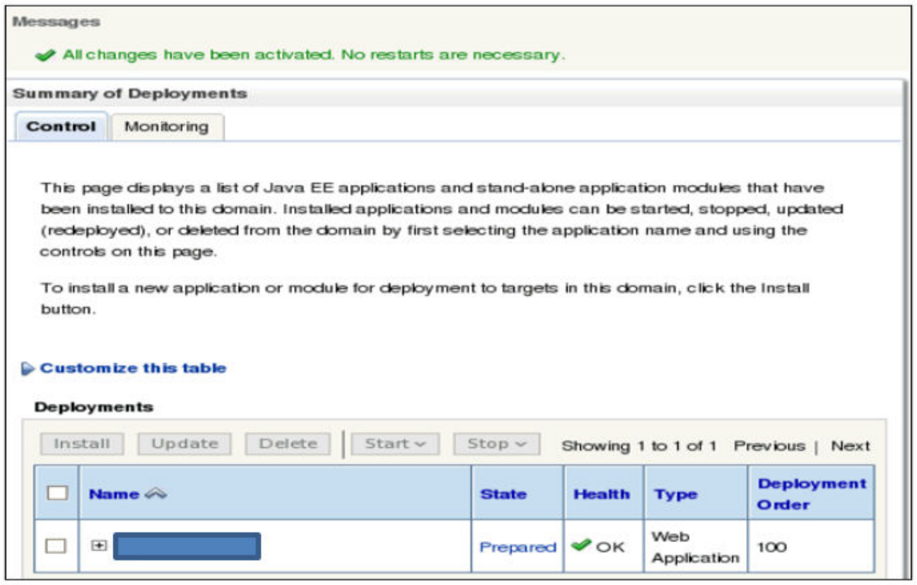


Change Center

Pending changes exist. They must be activated to take effect.

Notice the message indicating that the changes have been activated. In addition, notice the benefits application listed in the Deployments table is now in the **Prepared** state.

Figure 2-21 Deployments - changes activated



Messages

✓ All changes have been activated. No restarts are necessary.

Summary of Deployments

Control | Monitoring

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

▷ Customize this table

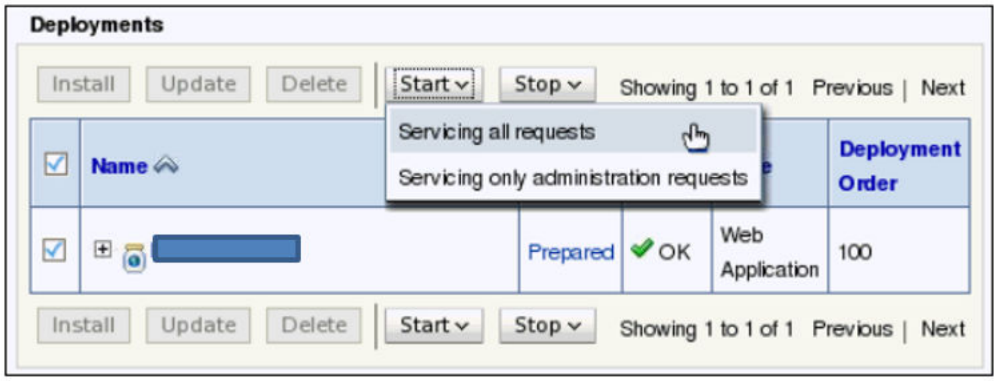
Deployments

Install | Update | Delete | Start | Stop — Showing 1 to 1 of 1 Previous | Next

☐	Name	State	Health	Type	Deployment Order
☐	⊞ [redacted]	Prepared	✓ OK	Web Application	100

13. Select the checkbox against the left of the benefits application in the Deployments table. In the Start drop-down list, select **Servicing all requests** option.

Figure 2-22 Checkbox - Servicing all requests



Deployments

Install | Update | Delete | Start | Stop — Showing 1 to 1 of 1 Previous | Next

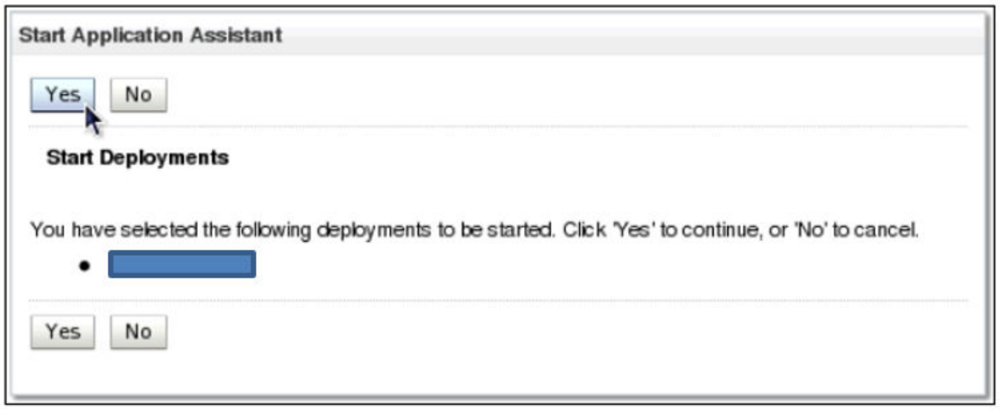
Servicing all requests
Servicing only administration requests

☑	Name	State	Health	Type	Deployment Order
☑	⊞ [redacted]	Prepared	✓ OK	Web Application	100

Install | Update | Delete | Start | Stop — Showing 1 to 1 of 1 Previous | Next

14. Click **Yes** to continue.

Figure 2-23   Start Application Assistant



Start Application Assistant

Yes No

Start Deployments

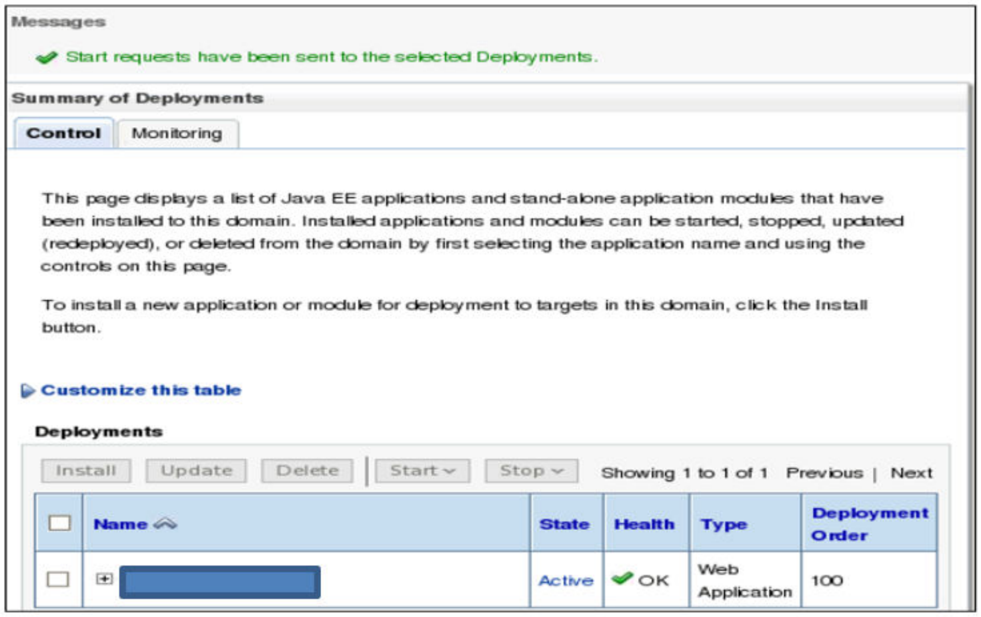
You have selected the following deployments to be started. Click 'Yes' to continue, or 'No' to cancel.

- [redacted]

Yes No

15. A message is displayed indicating a start request was sent. Subsequently Notice that the state of application is ‘**Active**’ indicating that the application is accessible.

Figure 2-24   Deployments - start request sent message



Messages

✓ Start requests have been sent to the selected Deployments.

Summary of Deployments

Control | Monitoring

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

▷ Customize this table

Deployments

Install | Update | Delete | Start ▾ | Stop ▾   Showing 1 to 1 of 1 Previous | Next

☐	Name	State	Health	Type	Deployment Order
☐	⊞ [redacted]	Active	✓ OK	Web Application	100

2.5 Web application UI for Accessing BOT

Web Application is User Interface where you can access the BOT functionality. The same can be integrated with OFSLL UI or any other front-end application such as customer support portal or financial institution website.

To configure WebApp, do one of the following:

- In case you wish to launch BOT as separate application, Modify **index.html** in OracleFSLLChatBot (or `OracleFSLLChatBot.war`) and update the following 2 fields with required details:
  - URI: '<ODA host>',

  - channelId: 'published bot channel ID'

- In case you wish to integrate BOT in an existing front-end application, use the provided **index.html** with the modified value and **web-sdk.js**

The BOT needs to be published on the login page and the only way it come be done is by adding the above properties in the Weblogic

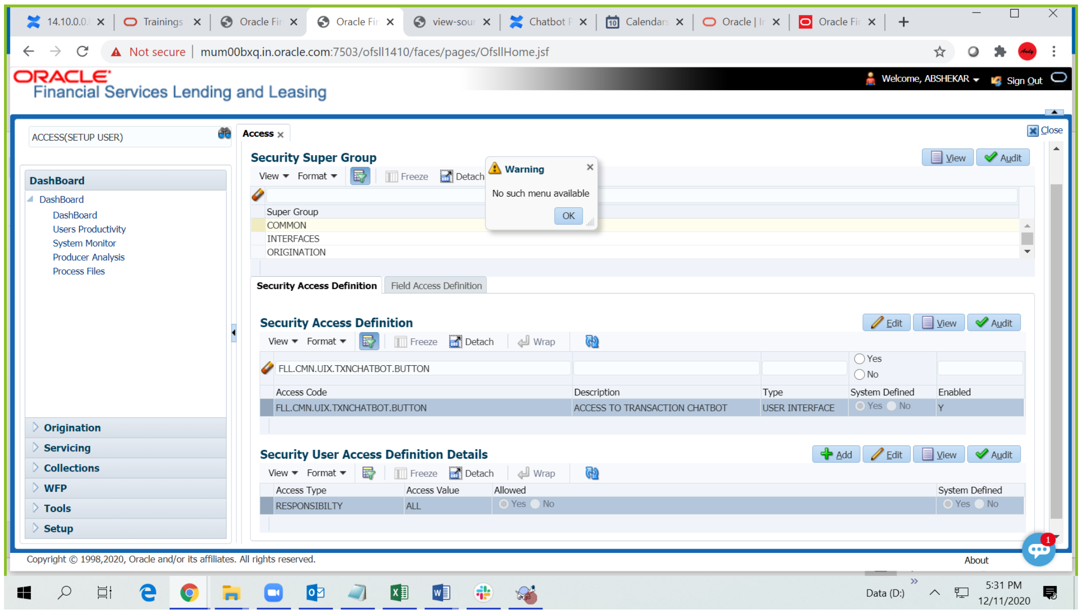
For additional information, contact Oracle Financial Services Lending and Leasing Product Engineering team.

2.6 App configuration for enabling chatbot

The below section details the process of app configuration for enabling chatbot to appear on OFSLL home page.

1. Enabling BOT and adding parameters:
   - Channel ID
   - URI
   - Enabled – Yes / No
2. Enable the FLS Access Key - FLL.CMN.UIX.TXNCHATBOT.BUTTON as indicated in the Access Setup screen.

Figure 2-25 Enable the FLS Access Key



Security Super Group

Warning
No such menu available
OK

Super Group
COMMON
INTERFACES
ORIGINATION

Security Access Definition | Field Access Definition

Security Access Definition

FLL.CMN.UIX.TXNCHATBOT.BUTTON

Access Code	Description	Type	System Defined	Enabled
FLL.CMN.UIX.TXNCHATBOT.BUTTON	ACCESS TO TRANSACTION CHATBOT	USER INTERFACE	Yes No	Y

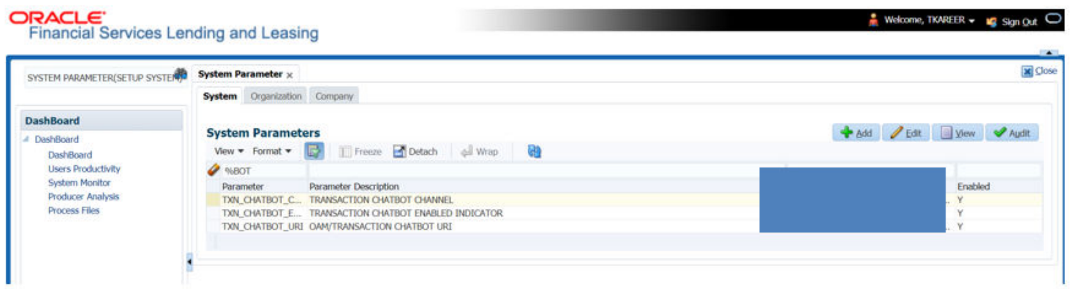
Security User Access Definition Details

Access Type	Access Value	Allowed	System Defined
RESPONSIBILTY	ALL	Yes No	Yes No

Copyright © 1998,2020, Oracle and/or its affiliates. All rights reserved.

3. Search for **System parameter** in the box below.

Figure 2-26   System parameter

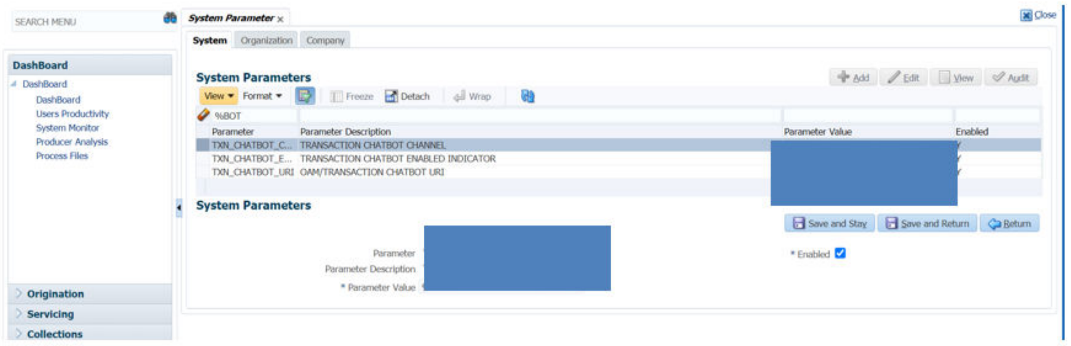


%BOT

Parameter	Parameter Description		Enabled
TXN_CHATBOT_C...	TRANSACTION CHATBOT CHANNEL	[redacted]	Y
TXN_CHATBOT_E...	TRANSACTION CHATBOT ENABLED INDICATOR	[redacted]	Y
TXN_CHATBOT_URI	OAM/TRANSACTION CHATBOT URI	[redacted]	Y

4. Enter the Channel id, click **save and return**.

Figure 2-27   Channel id



%BOT

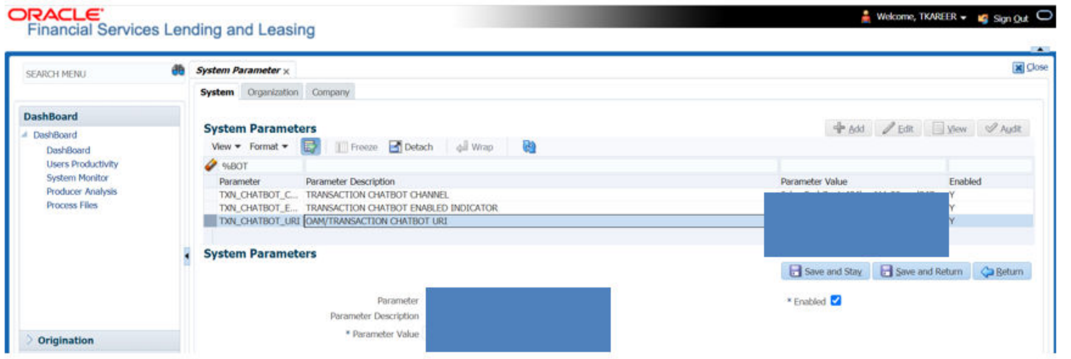
Parameter	Parameter Description	Parameter Value	Enabled
TXN_CHATBOT_C...	TRANSACTION CHATBOT CHANNEL	[redacted]	Y
TXN_CHATBOT_E...	TRANSACTION CHATBOT ENABLED INDICATOR	[redacted]	Y
TXN_CHATBOT_URI	OAM/TRANSACTION CHATBOT URI	[redacted]	Y

Parameter [redacted]
Parameter Description
* Parameter Value

* Enabled ☑

5. Enter the URI , click **save and return**.

Figure 2-28   URI



%BOT

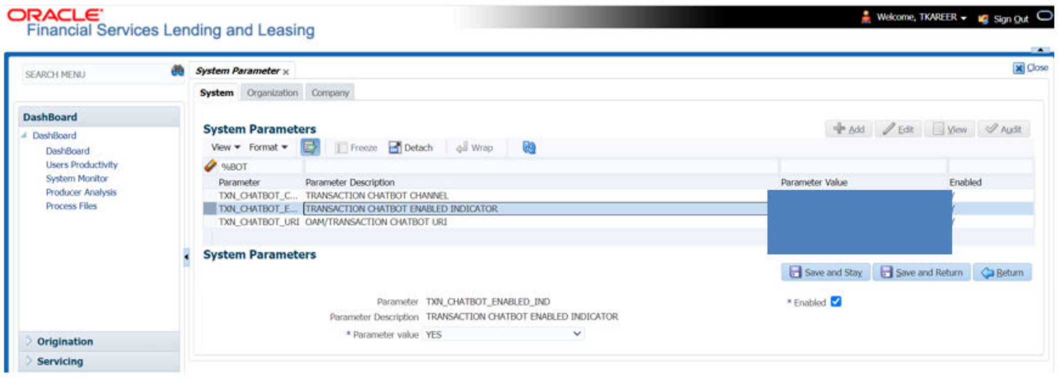
Parameter	Parameter Description	Parameter Value	Enabled
TXN_CHATBOT_C...	TRANSACTION CHATBOT CHANNEL	[redacted]	Y
TXN_CHATBOT_E...	TRANSACTION CHATBOT ENABLED INDICATOR	[redacted]	Y
TXN_CHATBOT_URI	OAM/TRANSACTION CHATBOT URI	[redacted]	Y

Parameter [redacted]
Parameter Description
* Parameter Value

* Enabled ☑

6. Enable Transaction bot.

Figure 2-29   Enable Transaction bot



System Parameters

%BOT

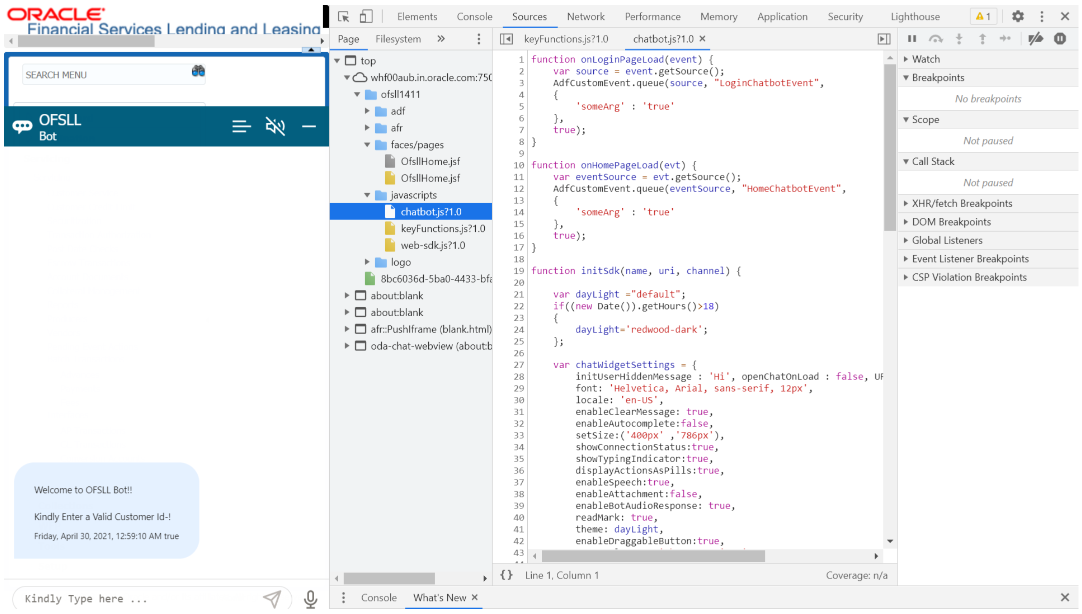
Parameter	Parameter Description	Parameter Value	Enabled
TXN_CHATBOT_C...	TRANSACTION CHATBOT CHANNEL	[redacted]	Y
TXN_CHATBOT_E...	TRANSACTION CHATBOT ENABLED INDICATOR	[redacted]	Y
TXN_CHATBOT_URI	OAM/TRANSACTION CHATBOT URI	[redacted]	Y

System Parameters

Parameter	TXN_CHATBOT_ENABLED_IND
Parameter Description	TRANSACTION CHATBOT ENABLED INDICATOR
* Parameter value	YES
* Enabled	✓

The below code needs to be implemented in the `chatbot.js` file as shown below:

Figure 2-30   Code implementation



```
function onLoginPageLoad(event) {
    var source = event.getSource();
    AdfCustomEvent.queue(source, "LoginChatbotEvent",
    {
        'someArg' : 'true'
    },
    true);
}

function onHomePageLoad(evt) {
    var eventSource = evt.getSource();
    AdfCustomEvent.queue(eventSource, "HomeChatbotEvent",
    {
        'someArg' : 'true'
    },
    true);
}

function initSdk(name, uri, channel) {

    var dayLight ="default";
    if((new Date()).getHours()>18)
    {
        dayLight='redwood-dark';
    };

    var chatWidgetSettings = {
        initUserHiddenMessage : 'Hi', openChatOnLoad : false, UF
        font: 'Helvetica, Arial, sans-serif, 12px',
        locale: 'en-US',
        enableClearMessage: true,
        enableAutocomplete:false,
        setSize:('400px' ,'786px'),
        showConnectionStatus:true,
        showTypingIndicator:true,
        displayActionsAsPills:true,
        enableSpeech:true,
        enableAttachment:false,
        enableBotAudioResponse: true,
        readMark: true,
        theme: dayLight,
        enableDraggableButton:true,
```

Welcome to OFSLL Bot!!

Kindly Enter a Valid Customer Id-!

Friday, April 30, 2021, 12:59:10 AM true

Ensure that no changes are done to the following js code:

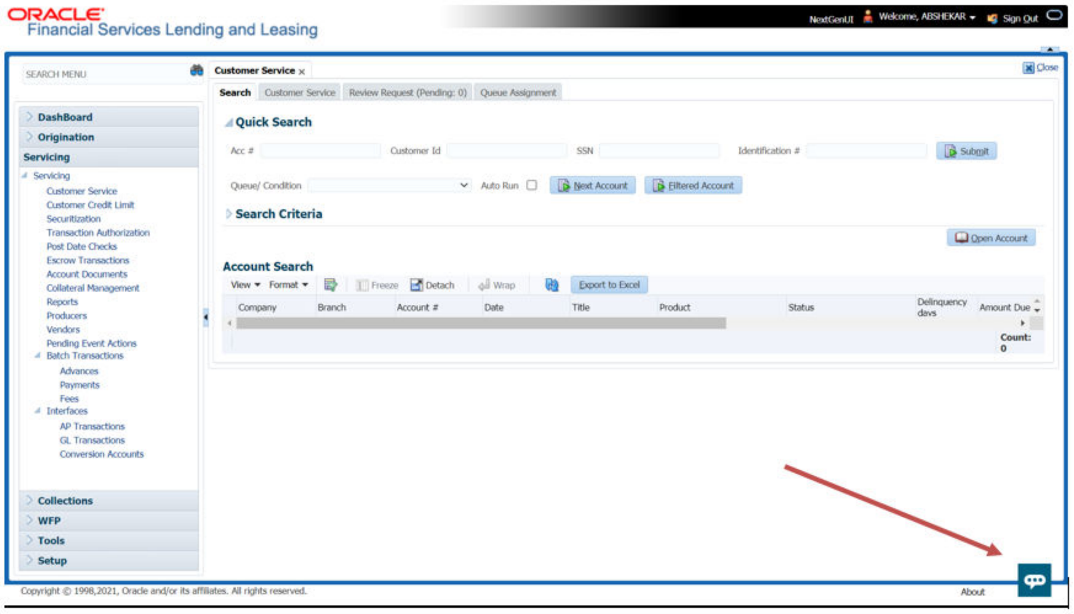
```
function onLoginPageLoad(event) {
var source = event.getSource();
AdfCustomEvent.queue(source, "LoginChatbotEvent",
{
'someArg' : 'true'
},
true);
}
function onHomePageLoad(evt) {
var eventSource = evt.getSource();
AdfCustomEvent.queue(eventSource, "HomeChatbotEvent",
{
```

```
'someArg' : 'true'
},
true);
}
function initSdk(name, uri, channel) {
var chatWidgetSettings = {
initUserHiddenMessage : 'Hi', openChatOnLoad : false, URI : uri,
channelId : channel,
font: '12px "Helvetica Neue", Helvetica, Arial, sans-serif',
locale: 'en-US',
enableClearMessage: true,
enableAutocomplete:false,
setSize:('400px' ,'786px'),
showConnectionStatus:true,
showTypingIndicator:true,
displayActionsAsPills:true,
enableSpeech:true,
enableAttachment:false,
enableBotAudioResponse: true,
skillVoices: [{
lang: 'en-US',
name: 'Samantha'
}, {
lang: 'en-US',
name: 'Alex'
}, {
lang: 'en-UK'
}]
};
if (!name) {
name = 'Bots';
}
setTimeout(function () {
const Bots = new WebSDK(chatWidgetSettings);// Initiate library with
configuration
Bots.connect()// Connect to server
.then(function () {
})
window[name] = Bots;
});
}
```

7. Web-sdk.js needs to be added from the << OFSLL Installed Directory >>/ /web_interface/ofsllbot/WebApp/scripts.

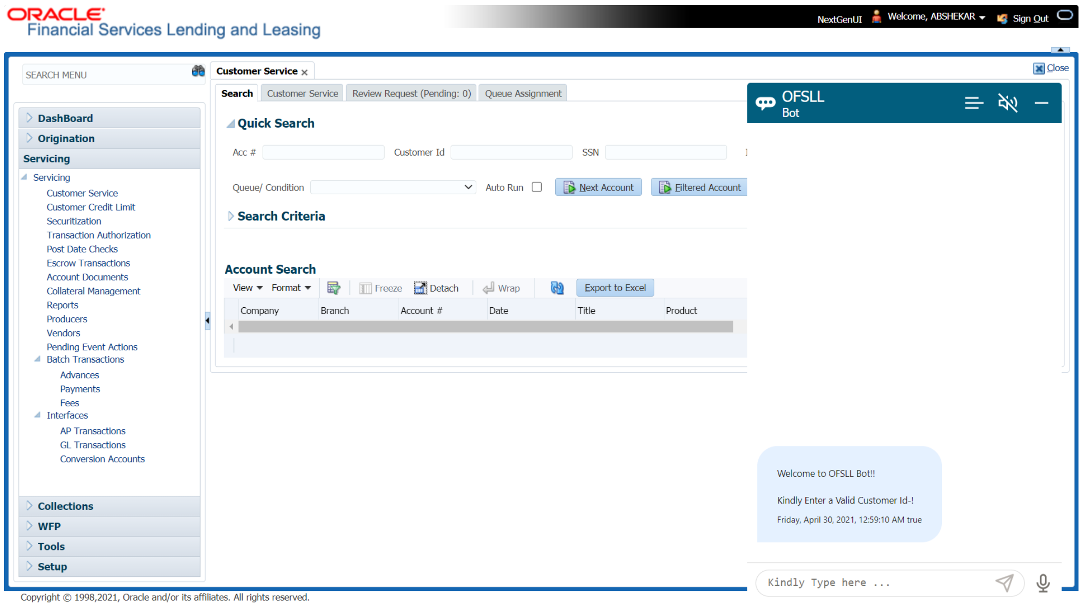
   The BOT after login is as shown below:

Figure 2-31 BOT logged in



On clicking bot icon, the interface is as displayed:

Figure 2-32 BOT interface



Welcome to OFSLL Bot!!

Kindly Enter a Valid Customer Id-!

Friday, April 30, 2021, 12:59:10 AM true

2.7 BOT Configuration

For the BOT to function, the following parameters are to be defined in the application.properties file available in the .war (OracleFSLLChatBot.war) in the path indicated below.

<OFSLL Installed Directory path>LL\release\<release version>\ws_as\ChatBot\OracleFSLLChatBot.war\WEB-INF\classes\

The below tables lists all the parameters of the properties file. However, only those fields marked as **Y** in Update required (Y/N) column are to be updated.

Table 2-1 BOT Configuration - Parameters

Sl.No	Parameter Name	Fields	Description	Update required (Y/N)	Sample
1	paymentPurposeRequired=Y	Boolean	Captures the Payment purpose Required	N	Y
2	accessToken=	String	Captures the access token	N	
3	proxyIP=	String	Captures the Proxy	N	
4	proxyPort=	Integer	Captures the Proxy Port	N	
5	googleAPIKey=	String	Captures the Google API key	N	
6	imageUrl=	Path	Captures the Image URL	N	
7	defaultHomeEntity=	String	Captures the home entity	N	
8	stockCode=	String	Captures the Stock Code	N	
9	moneyTransferPay=	String	Captures the Money Transfer Pay	N	
10	defaultBaseContext=	String	Captures the default base content	N	
11	sessionExpiryInMinutes = 15	Integer	Captures the Session timeout value	N	
12	ofsll.suffix = htm	String	Suffix of the files	N	Keep as .htm
13	ofsll.otmHttpUrl=https://docs.oracle.com/cd/	String	Captures the suffix for OTM Url	N	Keep as https://docs.oracle.com/cd/
14	ofsll.fIndex=/findex.htm	String	Captures the Findex path	N	Keep as /findex.htm
15	ofsll.index=index.htm	String	Captures the index.htm	N	Keep as index.htm
16	ofsll.video=/videos.htm	String	Captures the video file path	N	Keep as /video.htm
17	ofsll.ofsllReleaseNotes=/pdf/refdocs/ofsll_release_notes.pdf	String	Captures the OFSLL release notes suffix	N	Do not change

Table 2-1 (Cont.) BOT Configuration - Parameters

Sl.No	Parameter Name	Fields	Description	Update required (Y/N)	Sample
18	ofsll.ofsllReleaseDoc=https://docs.oracle.com/en/industries/financial-services/financial-lending-leasing/index.html	String	Captures the OFSLL release doc URL	N	Do not change
19	ofsll.splitSeparator==	String	Captures the Split separator	N	Do not change
20	ofsll.maxHitsResults=100	String	Captures the Max no of its results of the document query	Y (optional)	Change depending upon search results
21	ofsll.baseURL =	String	Captures the Service API URL	Y	Application URL
22	ofsll.username =	String	Captures the username of weblogic server	Y	Weblogic username
23	ofsll.pasd =	String	Captures the Password of weblogic server	Y	Weblogic password
24	ofsll.indexDir =/folder path	Path	Captures the complete folder path where index files are placed (In this location, copy the index files from respective release folder. The index dir specific files are available in the below location: LL\release\14_x_0_0_0\ws_as\ChatBot\14.x)	Y	Change as per server indexed folder. **Note**: Ensure to use the same dir file indicated the path.
25	ofsll.releaseVersionUrl=	Path	Captures the Part Number	Y	Refer Release Specific Indexing table.
26	ofsll.releaseNo=	Decimal	Captures the Release Number	Y	Refer **Folder Name** column Release Specific Indexing table.

Table 2-1 (Cont.) BOT Configuration - Parameters

Sl.No	Parameter Name	Fields	Description	Update required (Y/N)	Sample
27	ofsll.releaseHighlights=/pdf/refdocs/release_highlights.htm	String	Captures the release highlights file path	N	Keep as /pdf/refdocs/release_highlights.htm