Oracle® Financial Services Lending and Leasing

OFSLL Docubot Overview and Developer guide

Release 14.12.0.0.0
F82322-01
August 2024

ORACLE

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

F82322-01

Copyright © 2022, 2024, Oracle and/or its affiliates.

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.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. 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 fail-safe, 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.

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

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about 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 unless otherwise set forth in an applicable agreement between you and Oracle. 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, except as set forth in an applicable agreement between you and Oracle.

Contents

1 OFSLL DOCUBOT Overview and Developer Guide

1.1 Introduction 1-1
1.1.1 DocuBot Overview 1-2
1.1.2 About 1-2
1.1.3 Purpose 1-2
1.1.4 Audience 1-2
1.1.5 Accessibility 1-2
1.1.6 Access 1-2
1.1.7 Prerequisites 1-2
1.2 Indexing Workflow 1-3
1.2.1 How is Indexing Done? 1-3
1.2.2 Release Specific Indexing 1-4
1.3 How does Search Work? 1-4
1.4 Third Party Licenses 1-5
1.5 Features of BOT 1-5
1.5.1 Support of Text and Voice Based inputs 1-6
1.6 Launch OFSLL BOT 1-6
1.6.1 BOT UI Elements 1-8
1.6.2 BOT Usability Workflow 1-9

2 Developer Guide for BOT Customization

2.1 Pre-requisites 2-1
2.2 OFSLL Wrapper customization 2-2
2.3 ODA – Dialog Flow Development 2-5
2.4 Deploying war file on WebLogic Server 2-10
2.5 Web application UI for Accessing BOT 2-17
2.6 Configure CSF Mapping in Weblogic 2-18
2.7 BOT Configuration 2-22
2.8 Bot Customization 2-25
2.8.1 Label Customization 2-25
2.8.2 Bot Day/Night Theme Settings 2-26

2.8.3 Floating Icon 2-27

1

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

Following topics are discussed in OFSLL DOCUBOT Overview chapter:

- Introduction
- Indexing Workflow
- How does Search Work?
- Third Party Licenses
- Features of BOT
- Launch OFSLL BOT

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.

- DocuBot Overview
- About
- Purpose
- Audience
- Accessibility
- Access
- Prerequisites

1.1.1 DocuBot Overview

OFSLL integrated Docubot (Documentation searchable Chabot) is a functionality for guiding product end-users to navigate through release documentation with dynamic search capability along with other key features like access to latest release documents, module specific documentation, index glossary and so on.

This serves as knowledge repository and one point reference for information related to product usability, process workflow, installation, administration of all product release till date.

The Documentation ChatBot or DocuBot - hereafter is referred to as **BOT** in the document.

1.1.2 About

The documentation search engine adapted in BOT facilitates for all types of data search including textual, alphanumeric, numeric, keyword, phrases, and sentences. This is an **Elastic Search** and uses a data structure called **Inverted Index** which is designed for fast and full-text searches. An inverted index lists every unique word that appears in any document and identifies all of the documents in which the word is present.

The advantage of using elastic search is the speed, scalability and its ability to index most format of content.

1.1.3 Purpose

The purpose of this document is to demonstrate the capability of OFSLL BOT in handling documentation search requirements 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.

1.1.4 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.

1.1.5 Accessibility

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

BOT is agnostic of which self-service site / portal is used to provide access and interface to the users for help documentation.

1.1.6 Access

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

1.1.7 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 Indexing Workflow

The elastic search for OFSLL BOT requires pre-indexing of content. Hence, indexing is done for 14.12.0.0.0 release documents. 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.

- How is Indexing Done?
- Release Specific Indexing

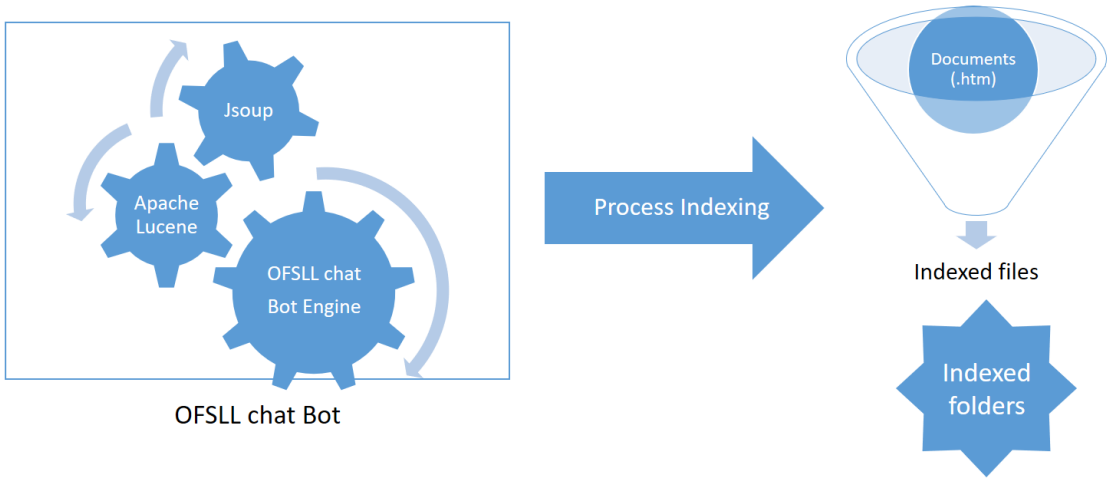
1.2.1 How is Indexing Done?

Apache Lucene manages an index over a dynamic collection of documents and provides very rapid updates to the index as and when documents are either added or removed from the collection. An index may store a heterogeneous set of documents.

Index in OFSLL BOT is handled by feeding all the release related html files. These htmls files are indexed both as single page reference and as well as at topic level using the hash tag to which it is mapped.

The workflow indicated below illustrates on how 3rd party Apache Lucene engine creates indexed files and stores in respective release specific folder.

Figure 1-1 Indexing



1.2.2 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

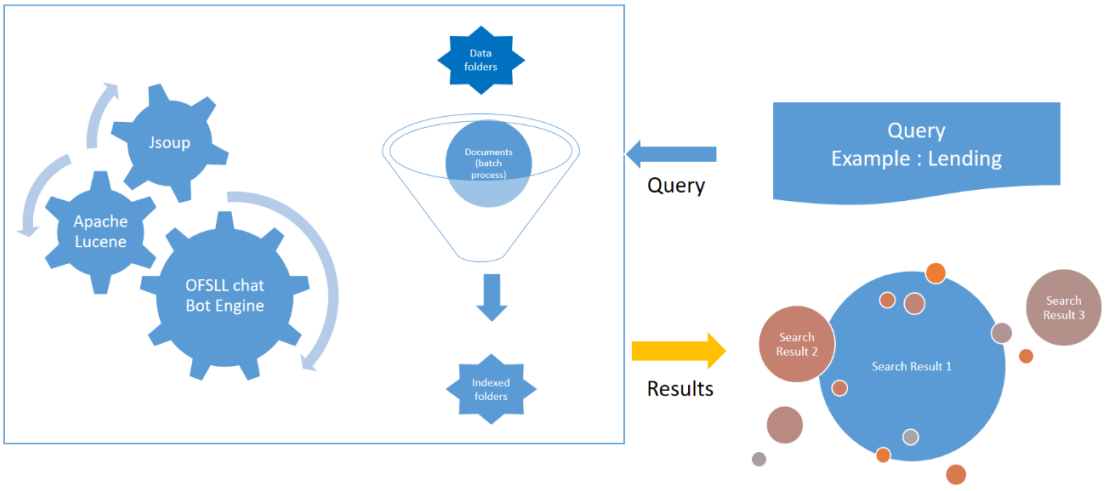
Release No	Folder Name	Part Number
14.12.0.0.0	14.12	F53373_01

1.3 How does Search Work?

On initiating the Search, the keyword is searched in the index directory and a URL link is returned in two ways for the specific release number mapped. The search results are provided in both the below combinations:

- Page(s) link in which the Search keyword is present
- Direct heading link in which the Search keyword is present. This is fetched using hash tag reference for the topic.

Figure 1-2 Search



1.4 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.

1.5 Features of BOT

Following are the unique features of OFSLL BOT:

- 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

1.5.1 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.

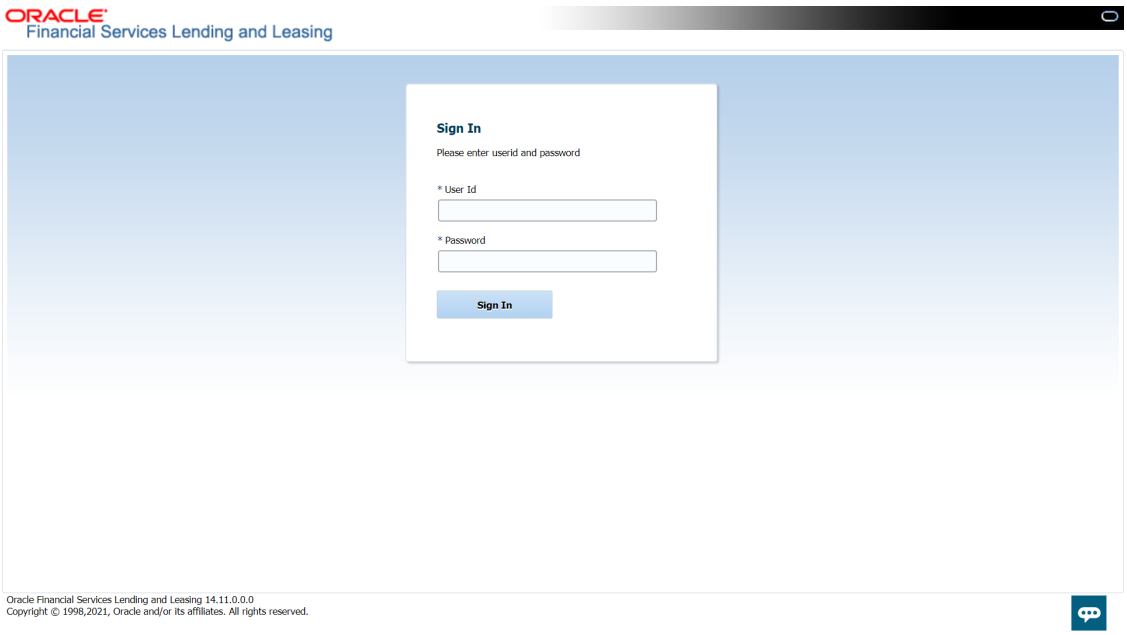
However, note that voice based input does not support to open a URL (link) reference.

1.6 Launch OFSLL BOT

OFSLL 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 launch of OFSLL application the BOT is available at right bottom corner.

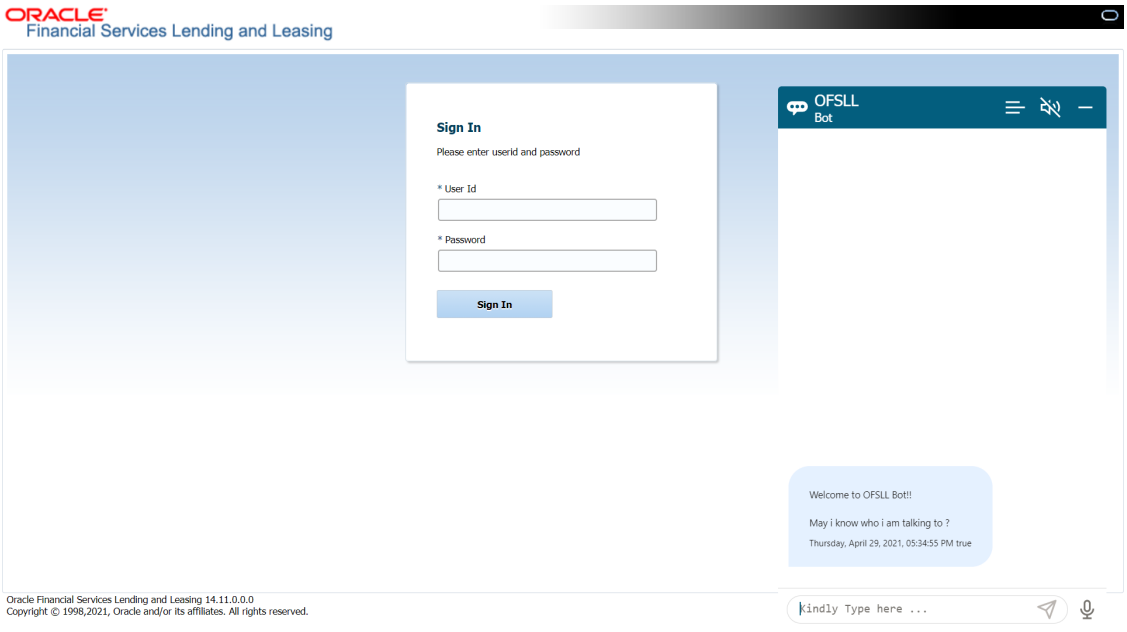
> **Note:**
>
> Note: Before you being, ensure to perform the required configuration as detailed in Developer Guide for BOT Customization chapter.

Figure 1-3 OFSLL BOT



On clicking the BOT icon, the welcome message is as displayed:

Figure 1-4 OFSLL BOT- Welcome message



This topic consists of the following sections:

- BOT UI Elements
- BOT Usability Workflow

1.6.1 BOT UI Elements

Figure 1-5 BOT UI Elements

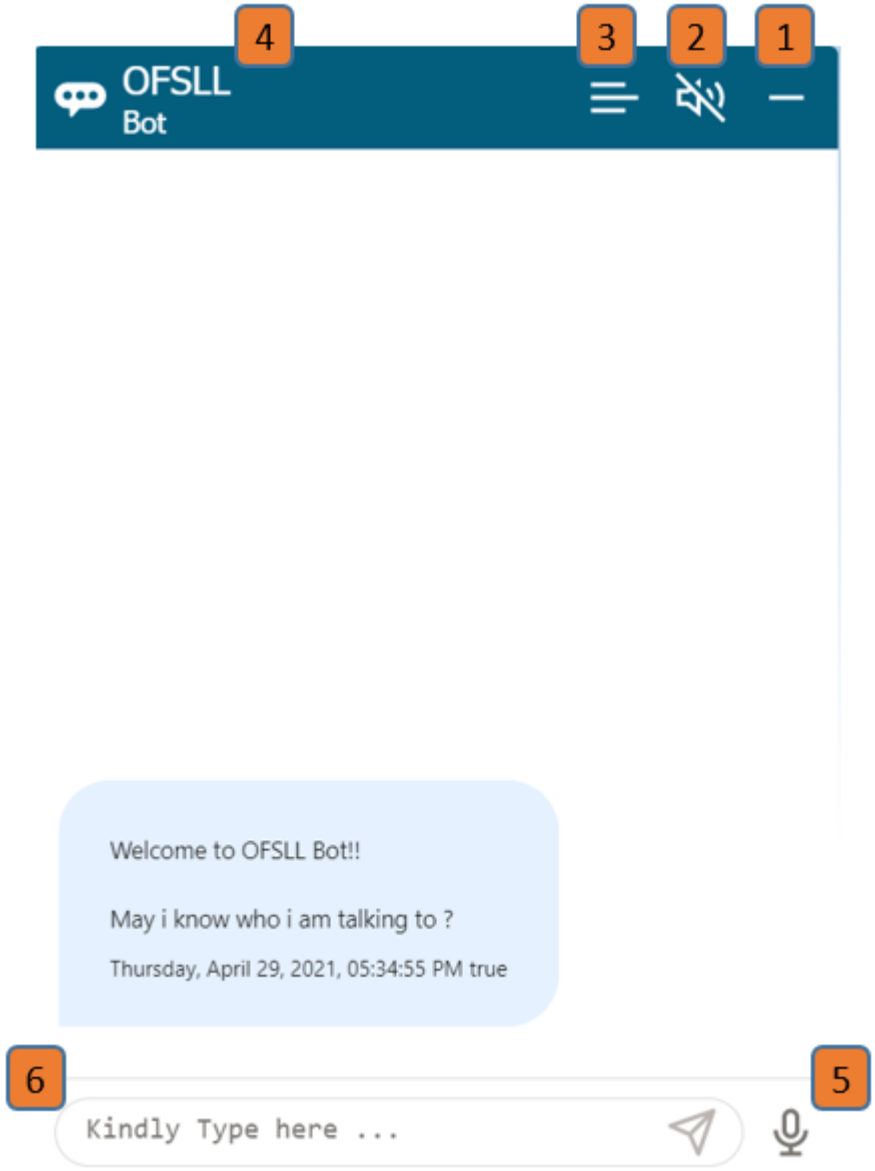


Table 1-2 BOT UI Elements

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 1-2 (Cont.) BOT UI Elements

Sl.No	Option	View / Action
4	Customized label	Customization for title label is detailed in Bot Customization section .
5	Mic Input	Enable Mic for voice based input
6	Text Input	Enter search string using keyboard

1.6.2 BOT Usability Workflow

Below is a simple sequence of user interaction with OFSLL BOT.

Table 1-3 BOT Usability Workflow

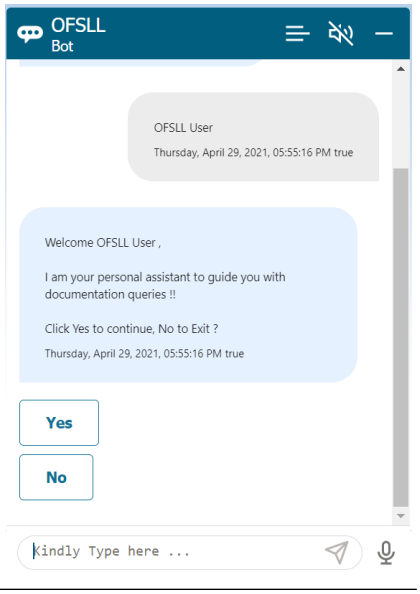
Sl.No	Action	BOT response	
1	Enter your name	BOT registers and provides a confirmation message to continue.	

Table 1-3 (Cont.) BOT Usability Workflow

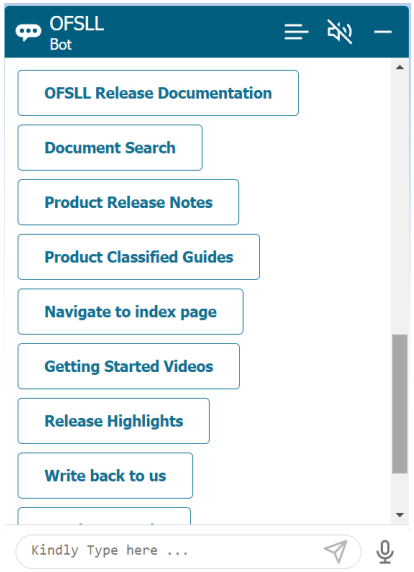
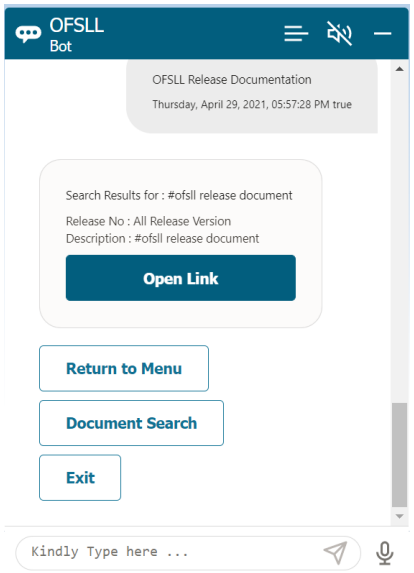
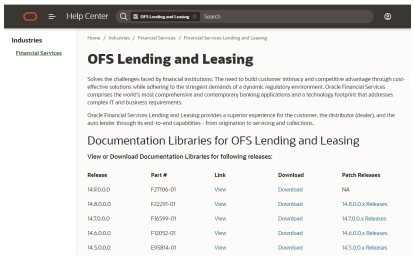
Sl.No	Action	BOT response	
2	Clicking on **Yes**	BOT presents list of documentation resorces for selection.	
	Clicking on **No**	Docubot displays an exit message. However, there is option to start again.	
3	Clicking on OFSLL Release Documentation	 Clicking on the Open Link, BOT displays the home screen with access to all release documenation libraries.	

Table 1-3 (Cont.) BOT Usability Workflow

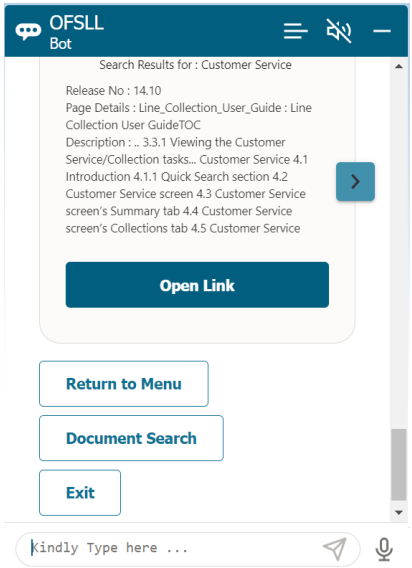
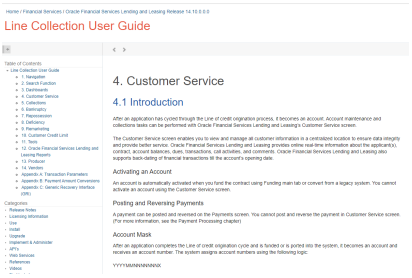
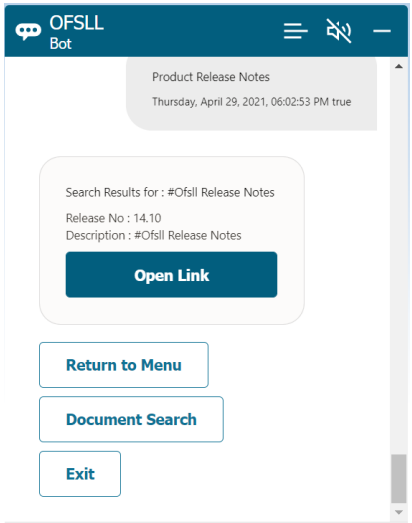
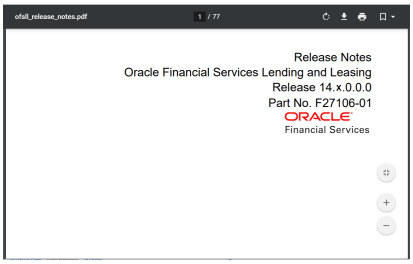
Sl.No	Action	BOT response	
4	Clicking on Documentation Search	 BOT displays dynamic results in horizontal navigating cards based on ranks from the list of documents which were indexed. Clicking on the Open Link, BOT displays the result page which contains the search keyword.	
5	Clicking on Product Release Notes	 Clicking on the Open Link, BOT displays the release notes of current release.	

Table 1-3 (Cont.) BOT Usability Workflow

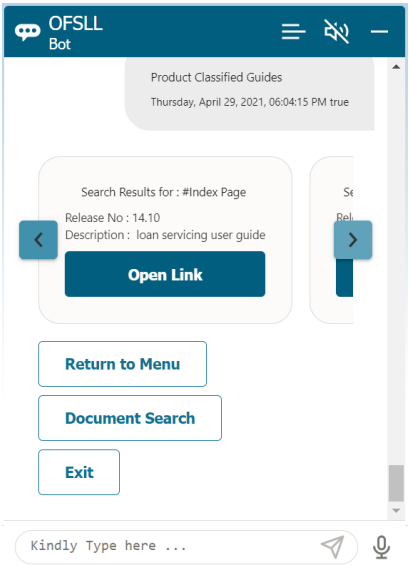
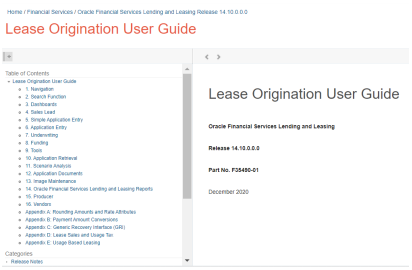
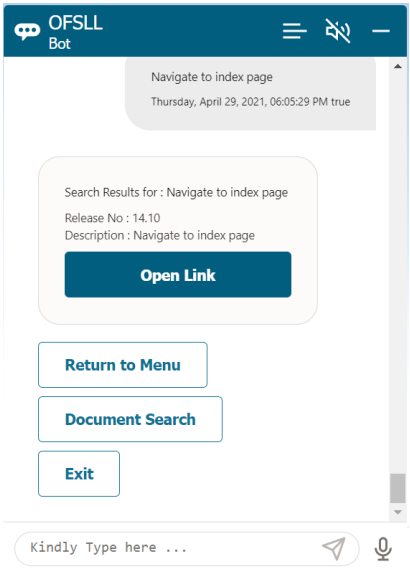
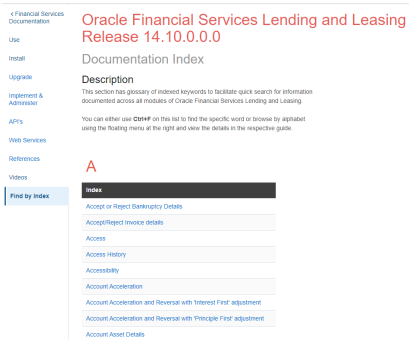
Sl.No	Action	BOT response	
6	Clicking on Product Classified Guides	 Clicking on the Open Link, BOT displays the module specific guide.	
7	Clicking on Navigate to index page	 Clicking on the Open Link, BOT displays the Find by Index page which consists of all indexed keywords in documentation.	

Table 1-3 (Cont.) BOT Usability Workflow

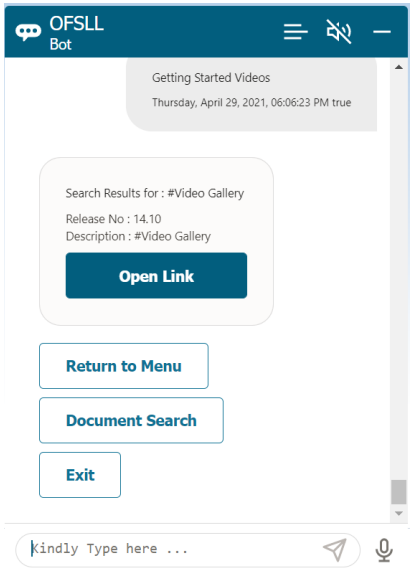
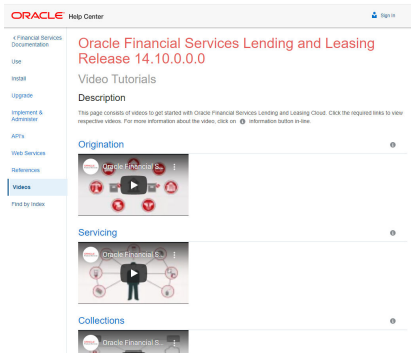
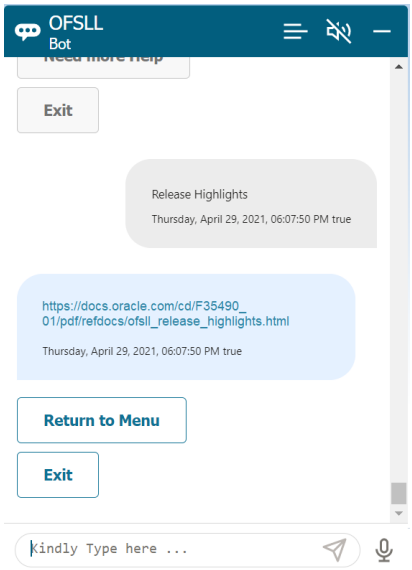
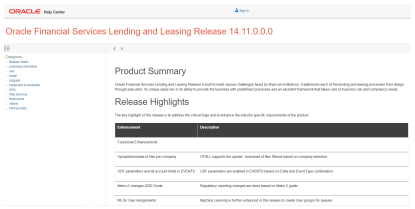
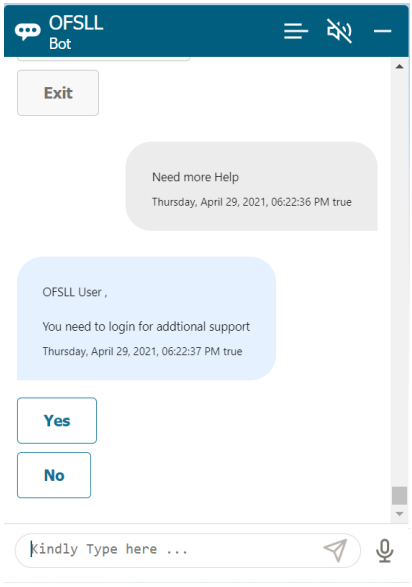
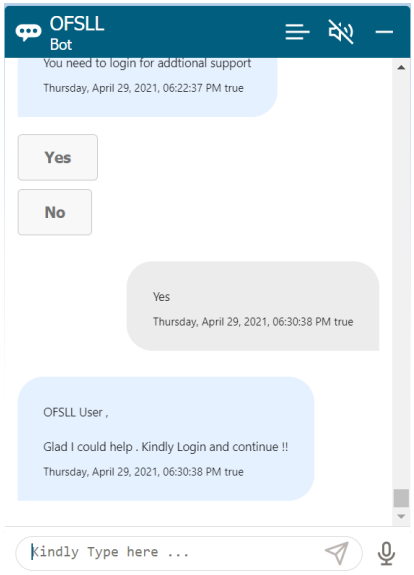
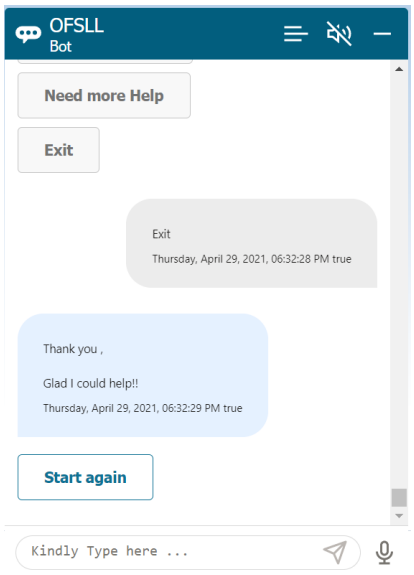
Sl.No	Action	BOT response	
8	Clicking on Getting Started Videos	 Clicking on the Open Link, BOT displays the video gallery link with getting started videos.	
9	Clicking on Release Highlights	 Clicking on the link, BOT displays the Release Highlights page which consists of new developments made to the current version of the product.	

Table 1-3 (Cont.) BOT Usability Workflow

Sl.No	Action	BOT response	
10	Clicking on Need More Help		
11	Clicking on Exit	BOT displays an exit message. However, there is option to start again.	

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
- Configure CSF Mapping in Weblogic
- BOT Configuration
- Bot Customization

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();
}
```

3. Export project as 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

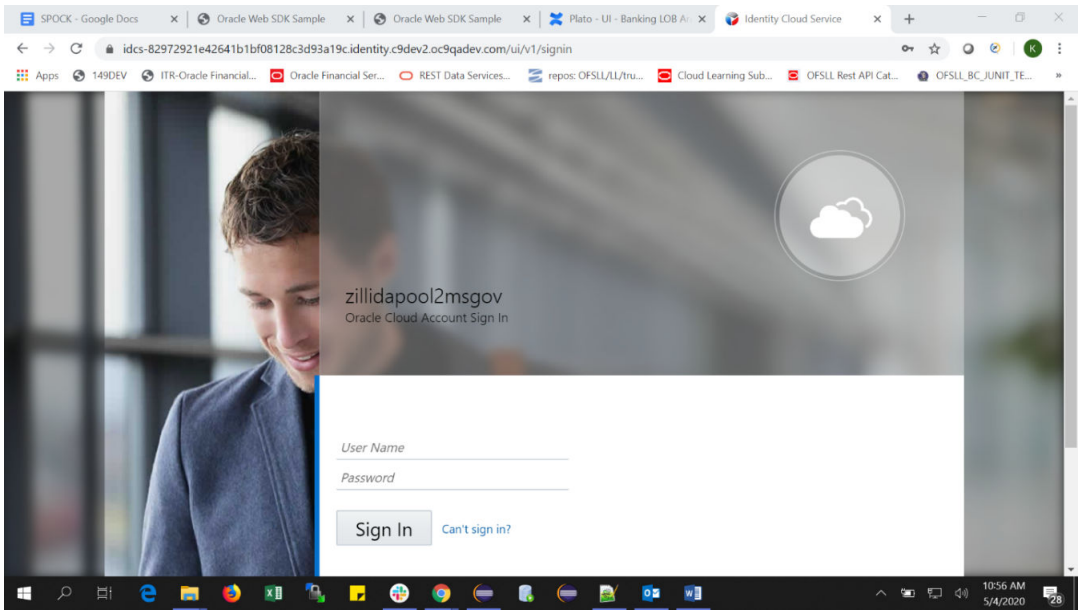
It is recommended to refer to ODA documentation for detailed information - https://docs.oracle.com/en/cloud/paas/digital-assistant/index.html

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\documentation-bot`

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

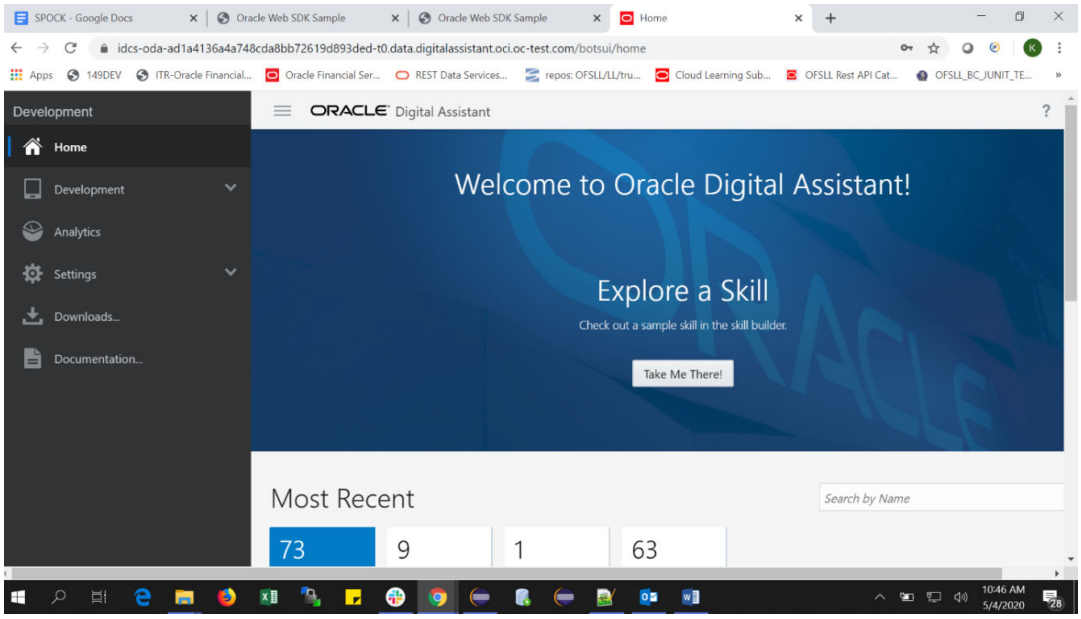
1. Login to ODA UI

Figure 2-1 ODA UI - Login page



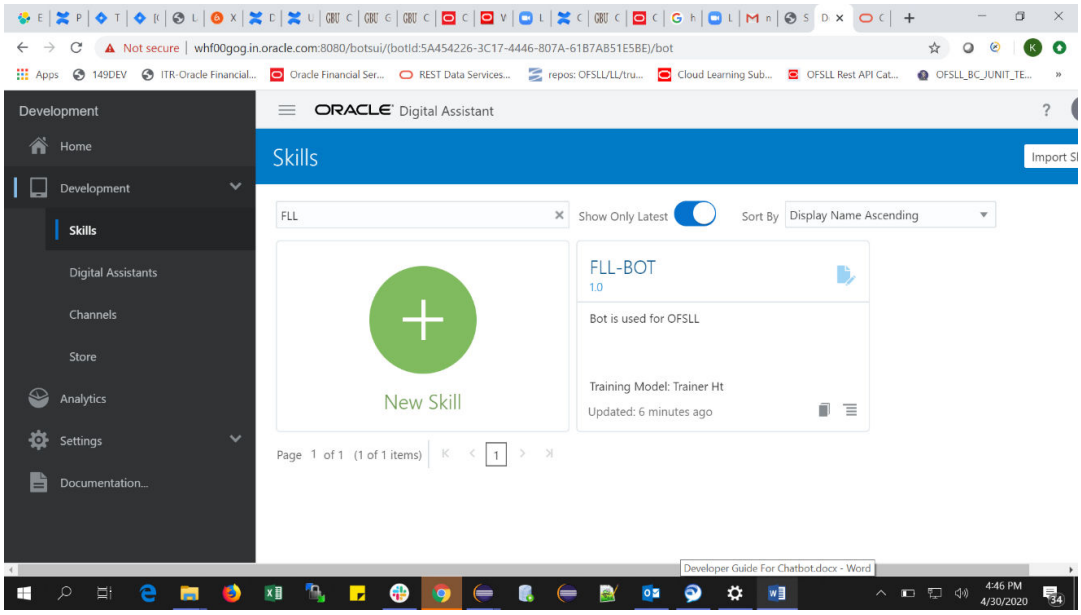
2. Go to Home

Figure 2-2 ODA UI - Home



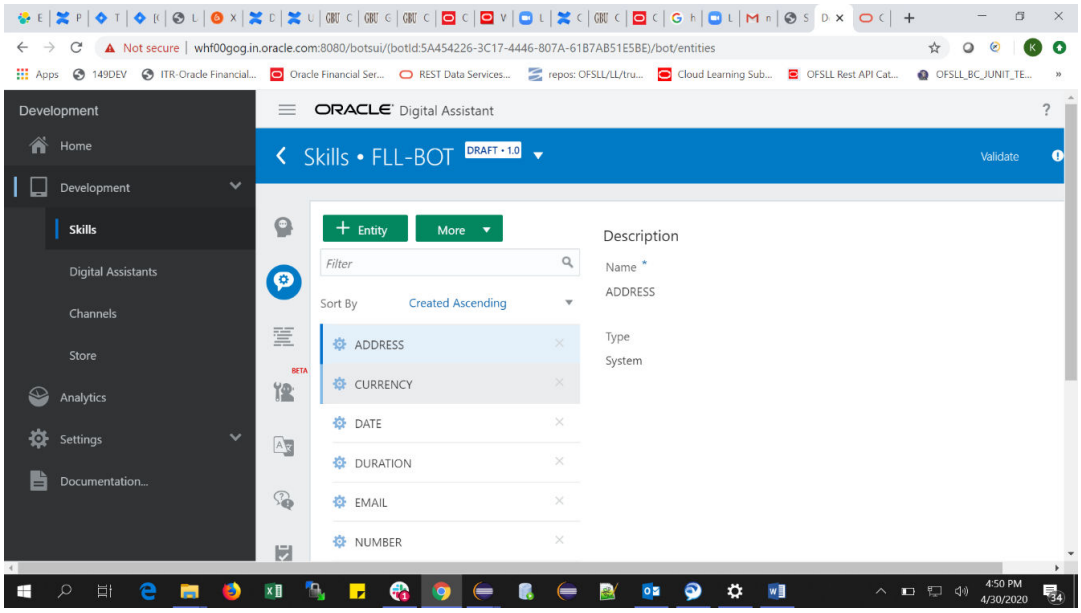
3. Create Skill/Digital Assistant.

Figure 2-3 Skill Assistant



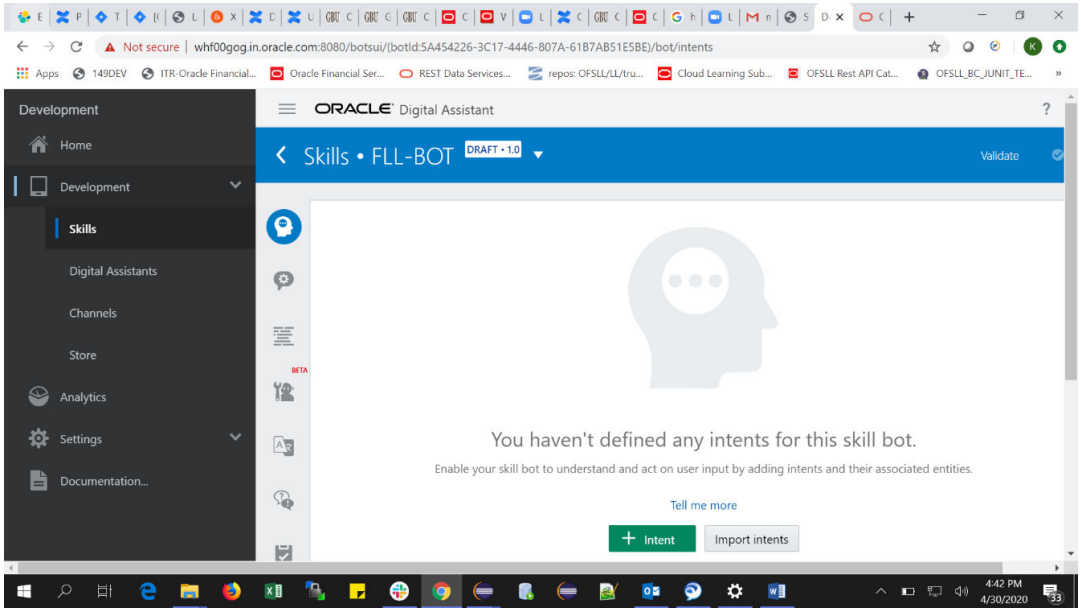
4. Add Entities

Figure 2-4 Add Entities



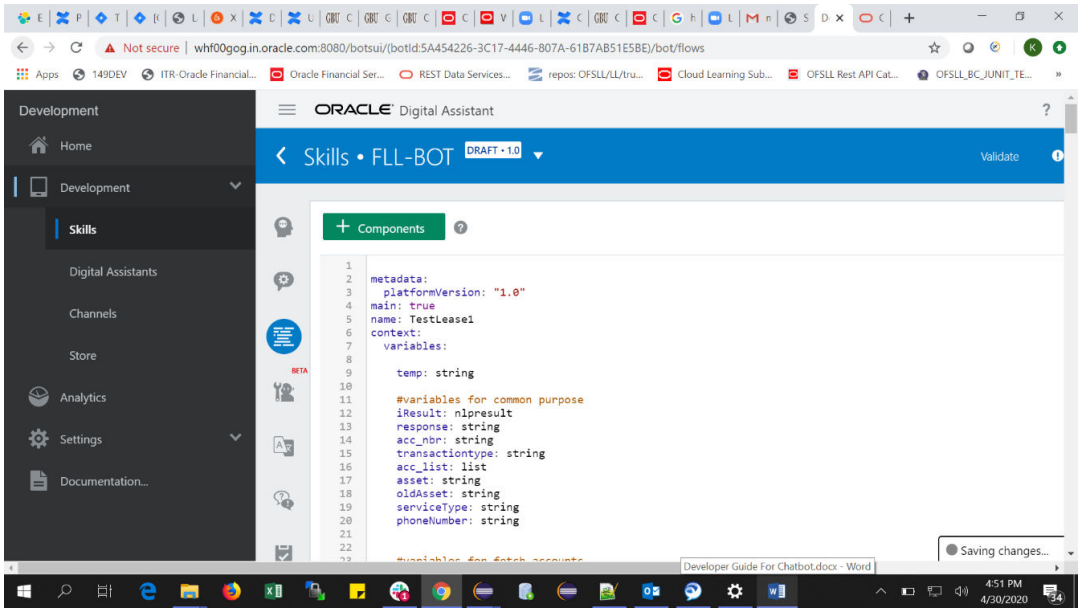
5. Add Intents. This involves defining Activity, Available option, Next level, Breakpoint, intermediate steps.

Figure 2-5 Add Intents



6. Add Bot flow using Yaml

Figure 2-6 Add Bot flow



7. Add OFSLL REST Service

Figure 2-7 OFSLL REST Service

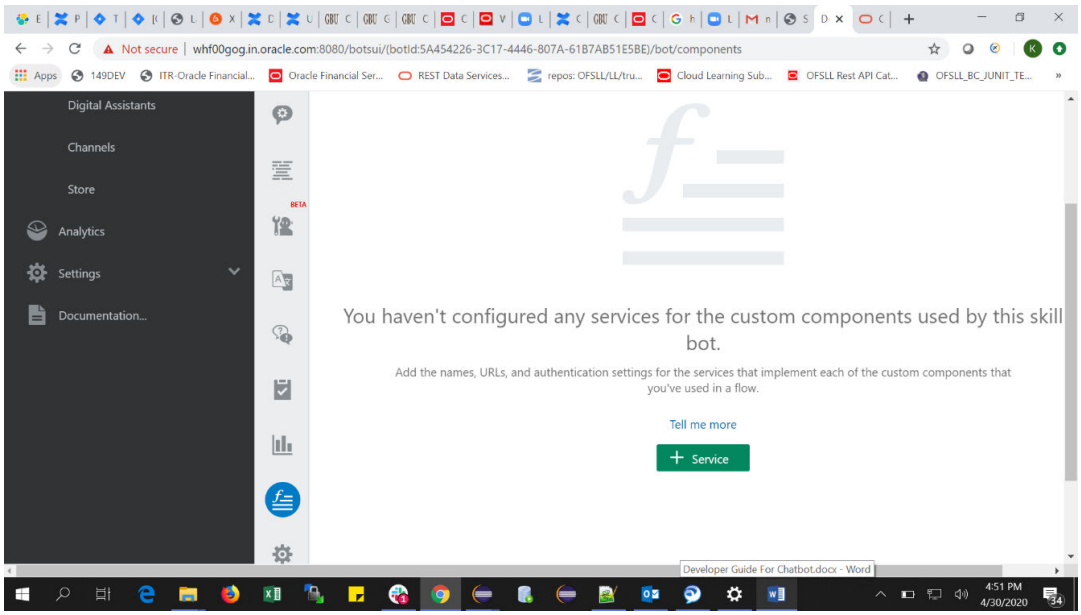
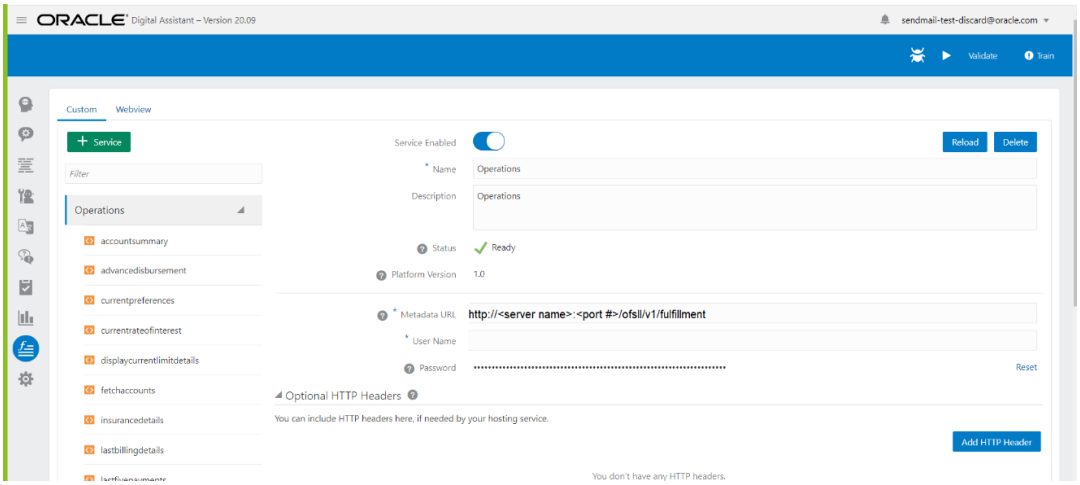
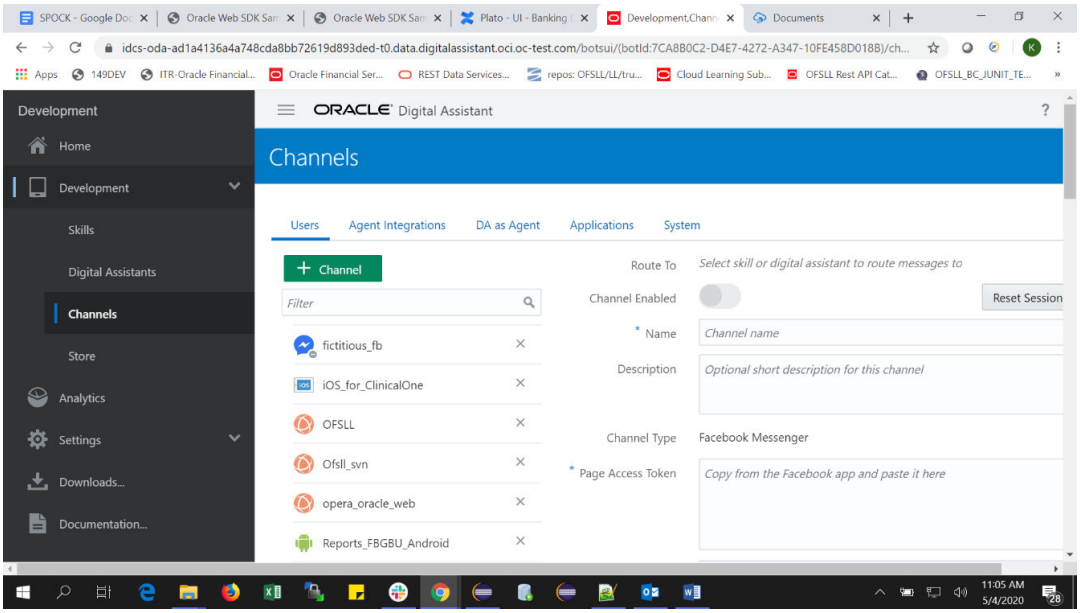


Figure 2-8 REST Service



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



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-documentation-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

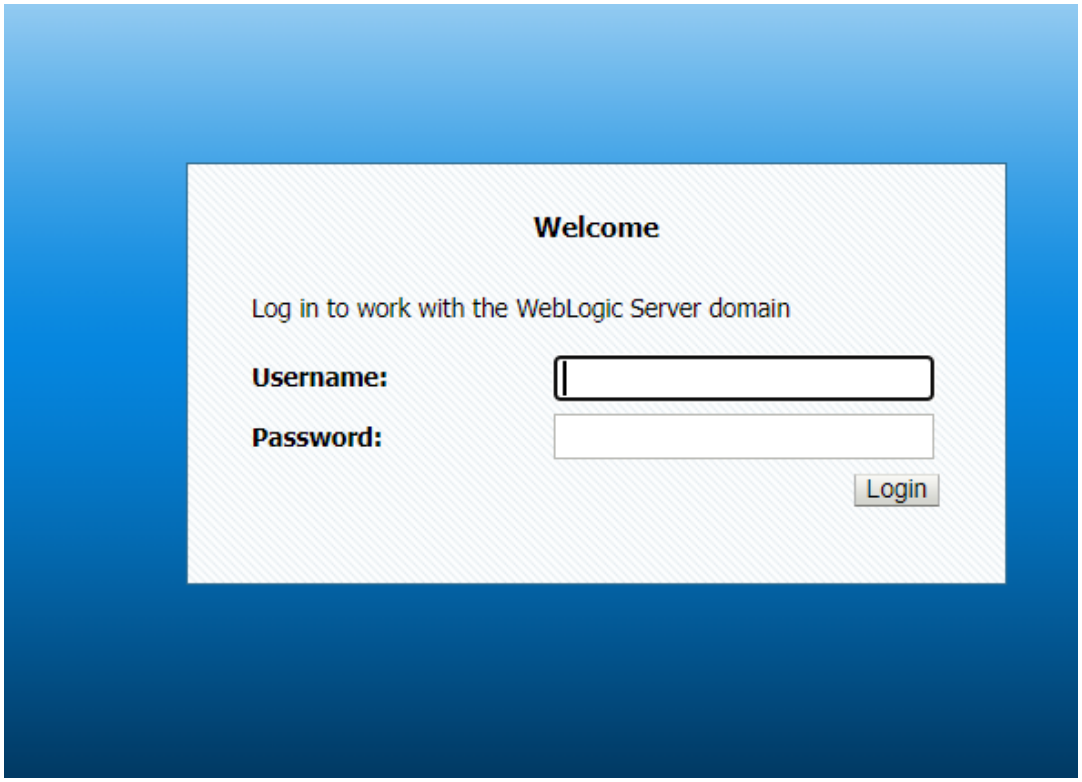
Before you begin, ensure to use the war file for deployment of OFSLL BOT available in the path – `release\<14_x.0.0.0>\ws_as\ChatBot\OracleFSLLChatBot.war`.

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

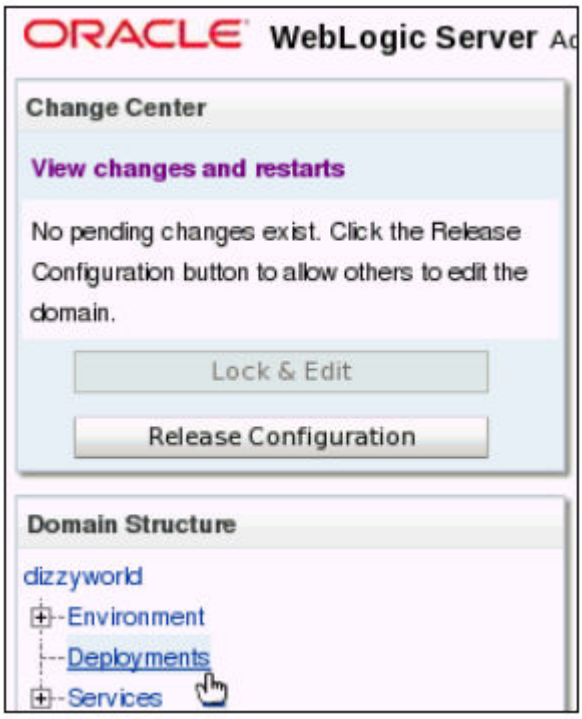


2. Enter valid login credentials.
3. Deploying an application is a change to the domain's configuration, so it must first be locked. In the Change Center. Click **Lock & Edit**.

Figure 2-11 WebLogic Server - Change center

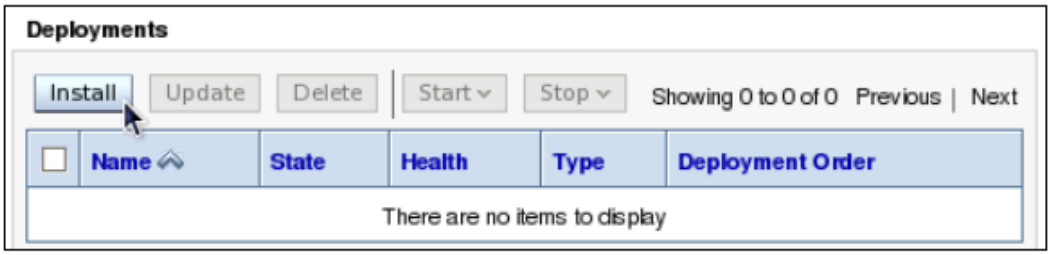
4. Under Domain Structure, click **Deployments**.

Figure 2-12 Deployments



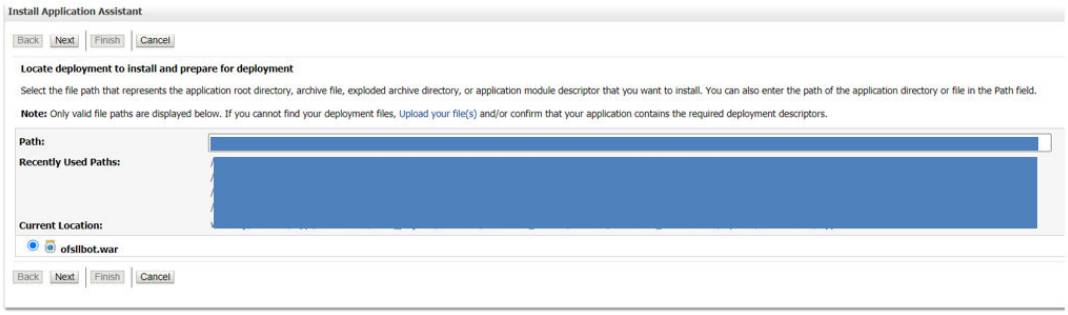
5. On the right, under Deployments, click **Install**.

Figure 2-13 Install



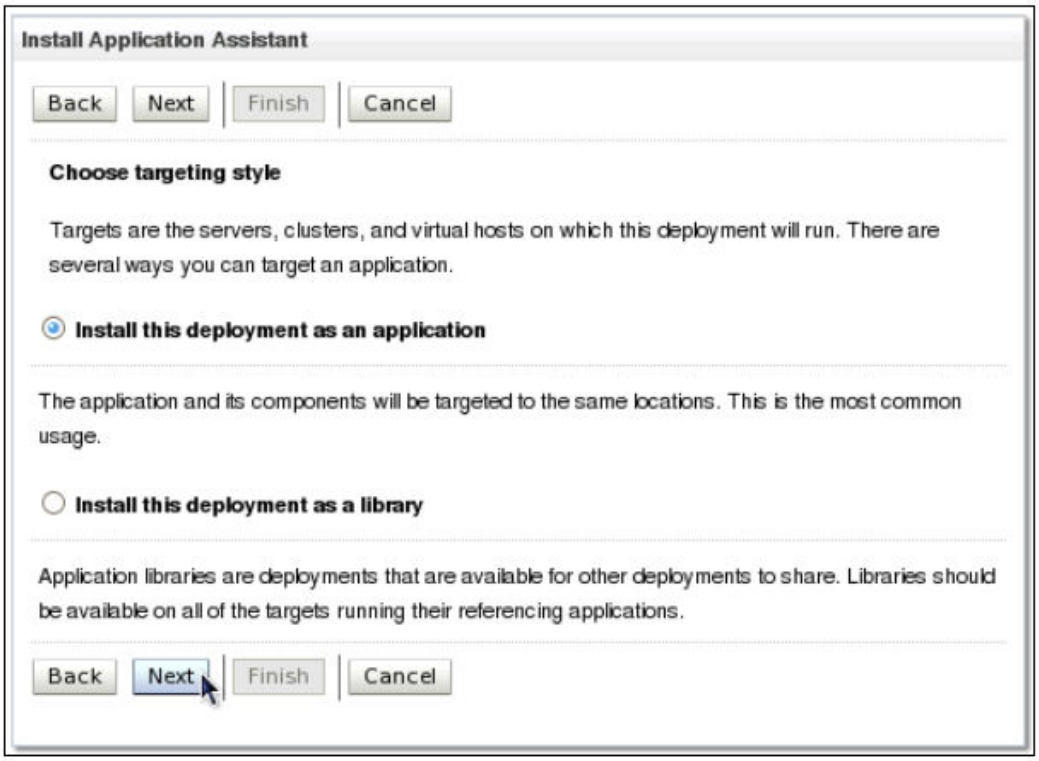
6. Find the Current Location field. Use the links to browse to the location in which you placed the downloaded `OracleFSLLChatBot.war` file.
7. The `.war` file is available in the path - `release\<14_x.0.0.0>\ws_as\ChatBot\OracleFSLLChatBot.war`. Select the `.war` file from the given path and click the 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



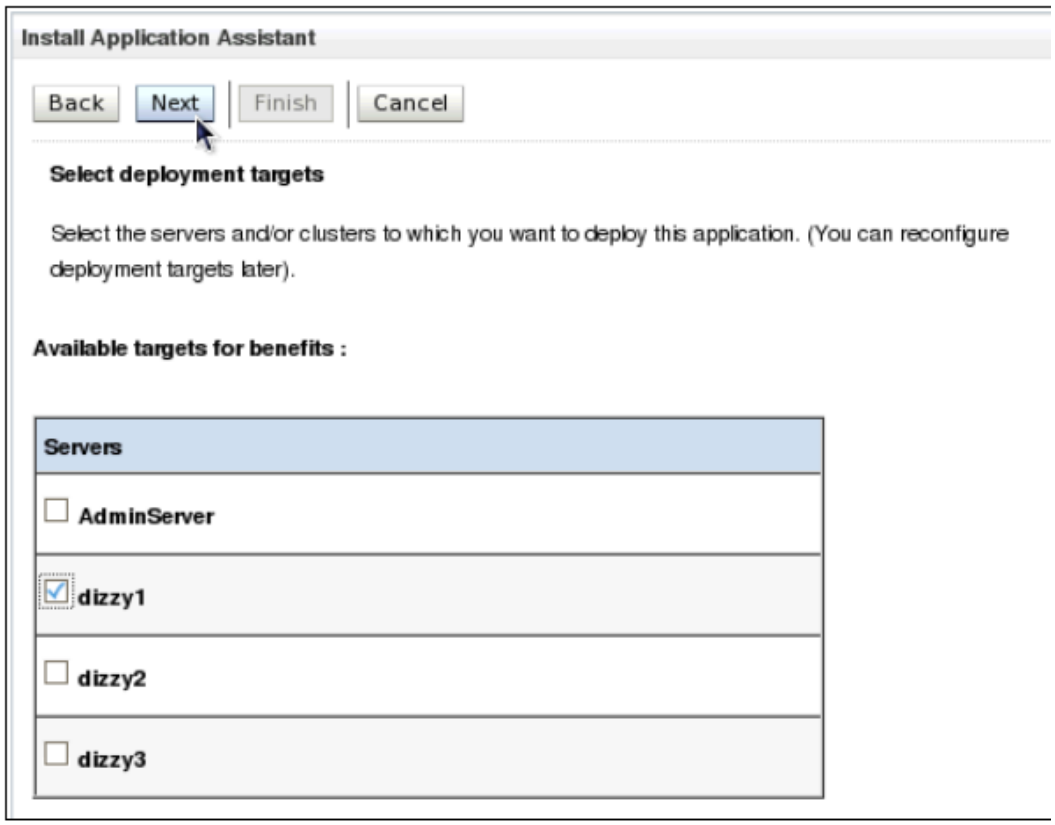
8. Ensure that **Install this deployment as an application** option is selected. Click **Next**.

Figure 2-15 Install Application Assistant 2



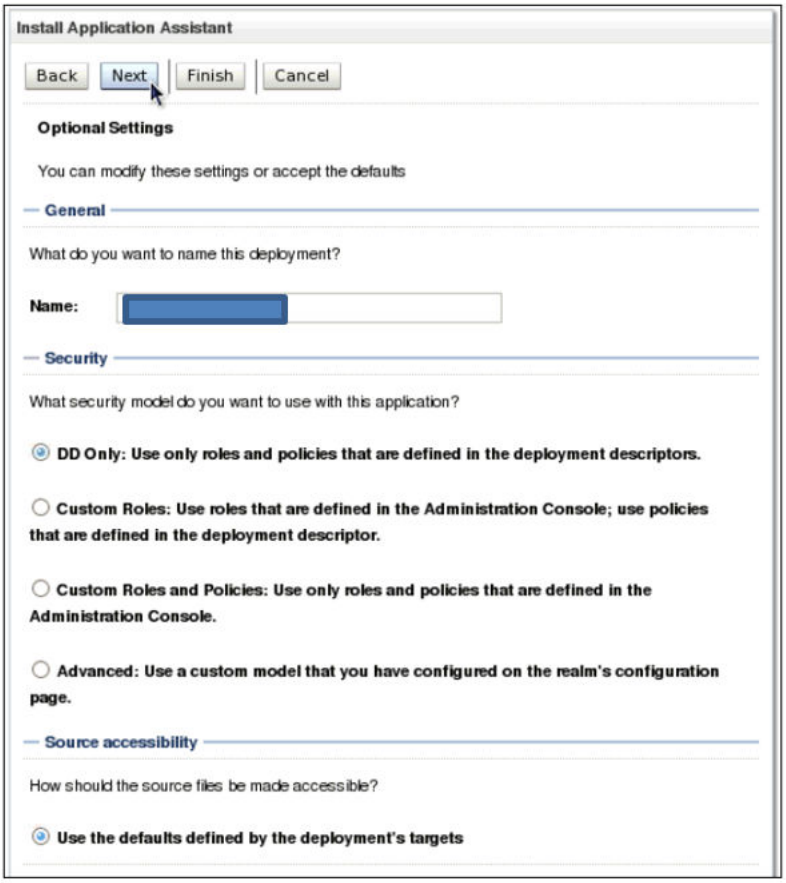
9. In the below window, click **Next**.

Figure 2-16 Install Application Assistant 3



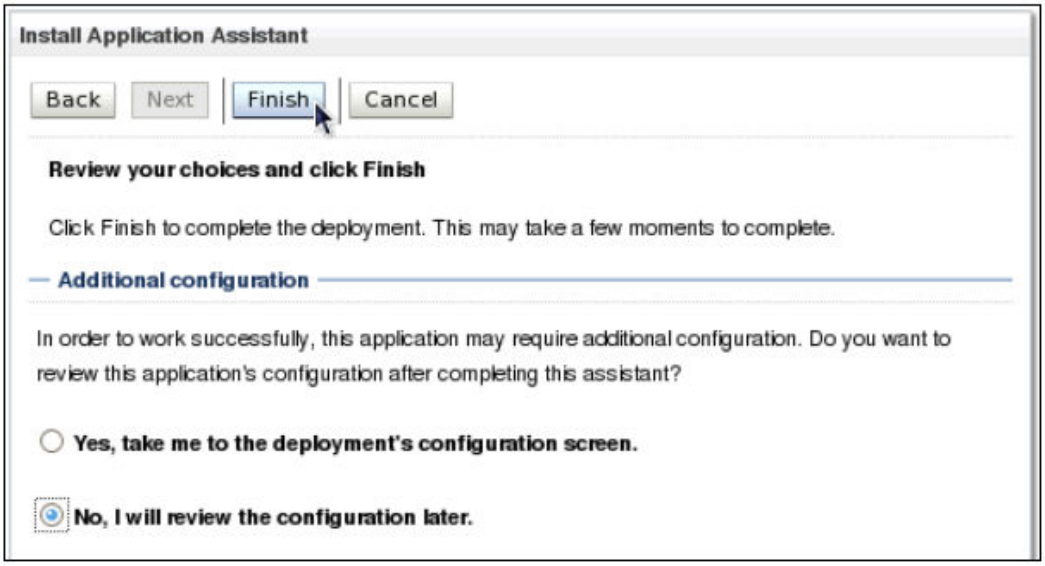
10. Retain the default values and click **Next**.

Figure 2-17 Install Application Assistant 4



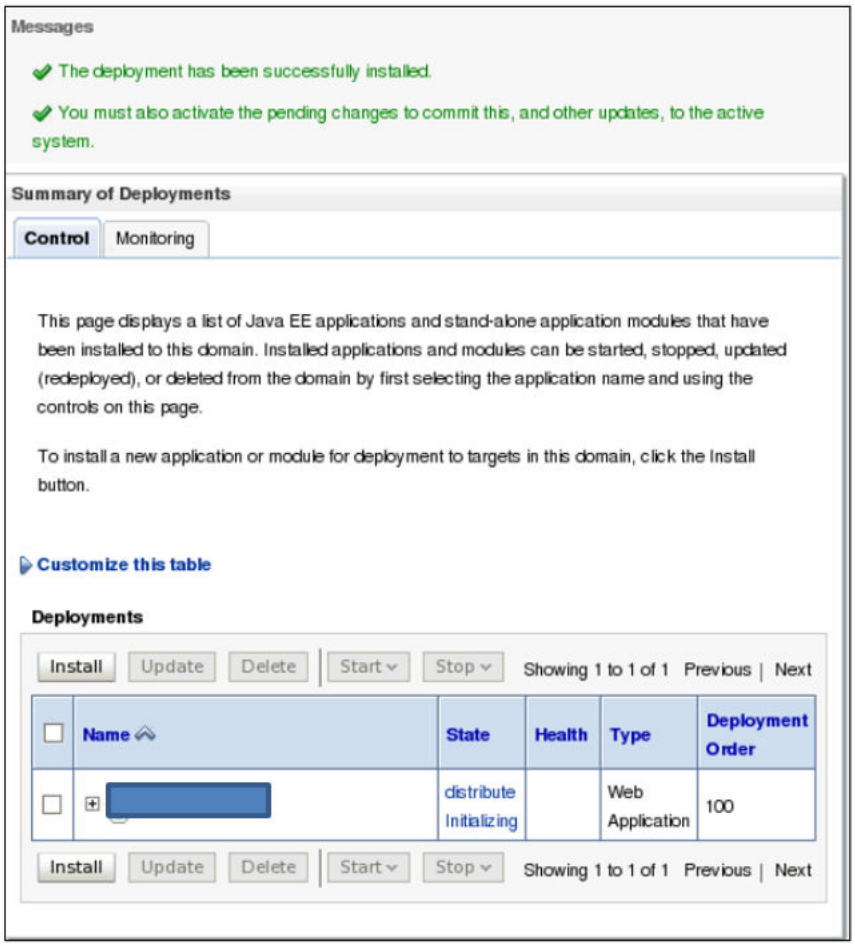
11. In the below window, select the option **No, I will review the configuration later** and click **Finish**.

Figure 2-18 Install Application Assistant 5



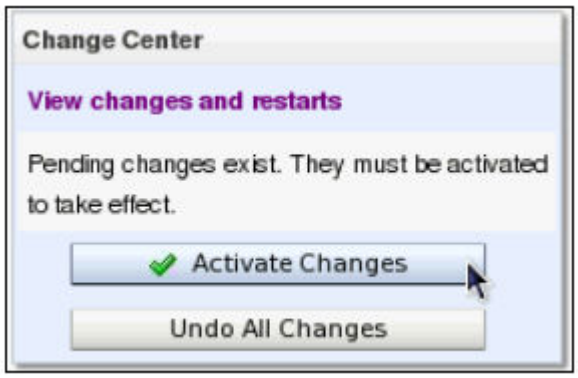
Once done view the messages indicating that the deployment was installed, but changes must be activated. In addition, notice the benefits application listed in the Deployments table.

Figure 2-19 Installed Deployment - message



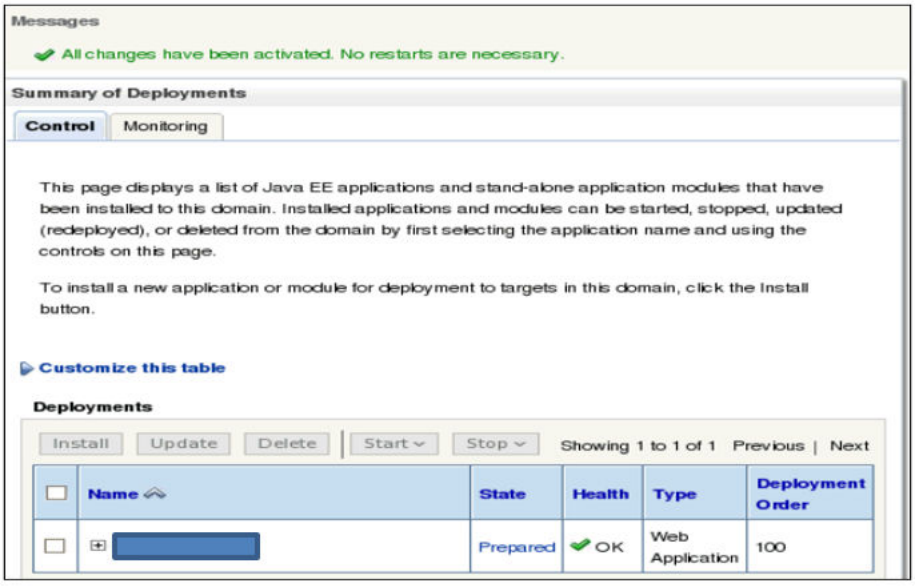
12. In the Change Center, click the **Activate Changes** button.

Figure 2-20 Activate Changes



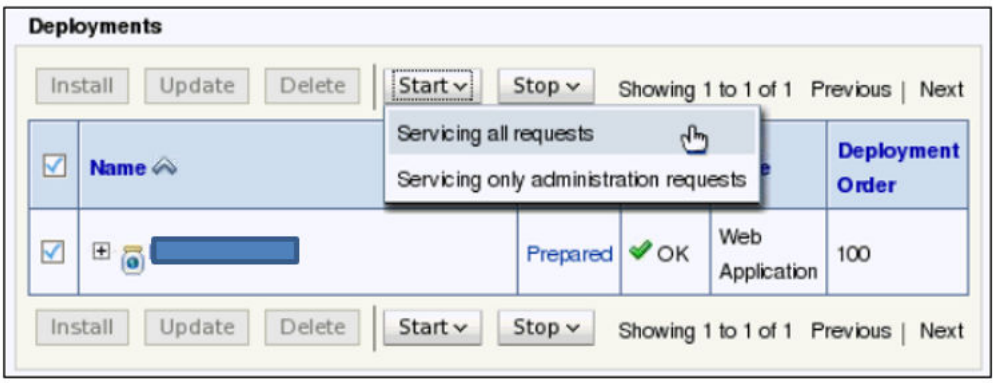
Notice the message indicating that the changes have been activated. In addition, notice the benefits application listed in the Deployments table is now in the **Prepared** state.

Figure 2-21 Deployments - changes activated



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

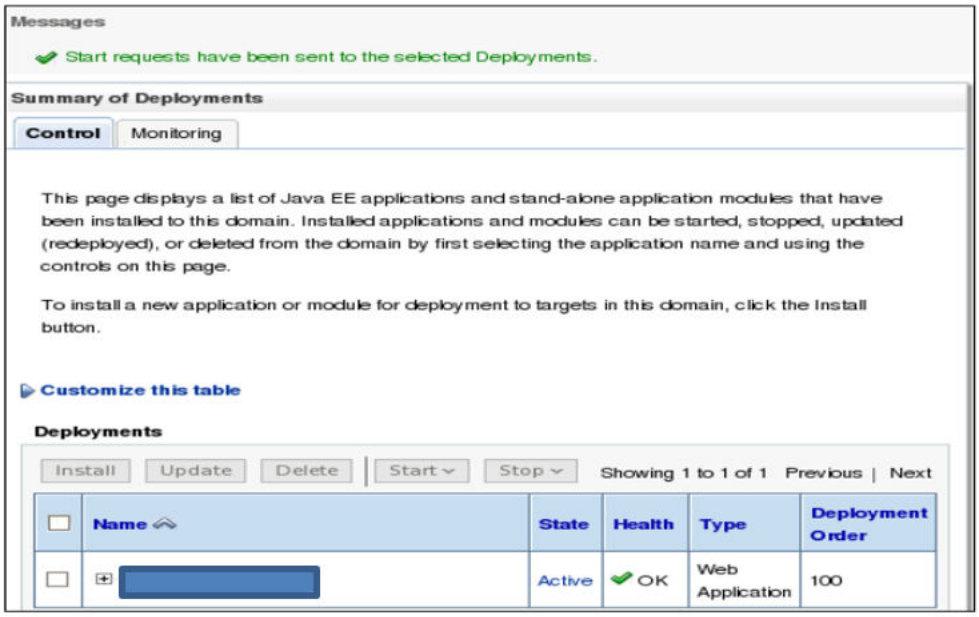


14. Click **Yes** to continue.

Figure 2-23 Start Application Assistant

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



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.

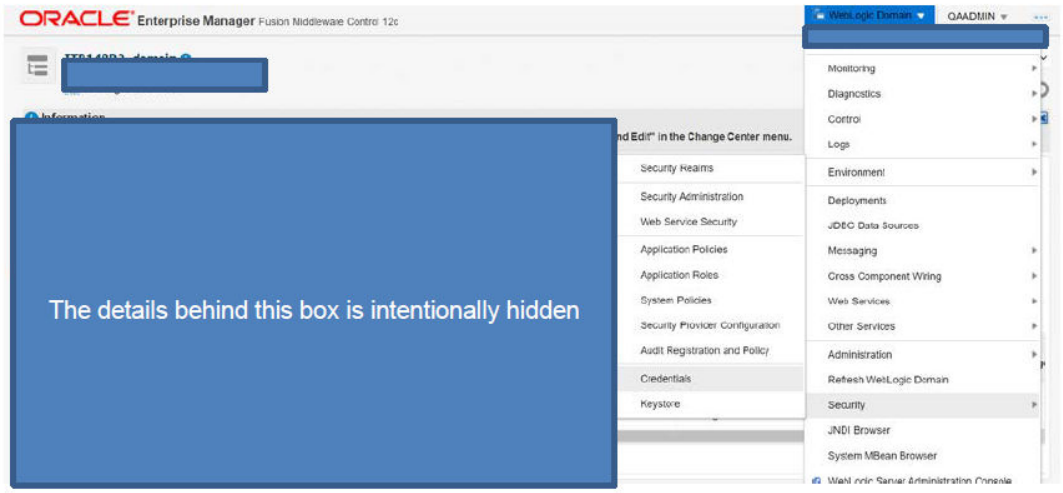
2.6 Configure CSF Mapping in Weblogic

The below section details the process of CSF configuration for BOT to appear on OFSLL home page.
Following are the Parameters:

- Create a map called LOGIN_CHATBOT_MAP_NAME = ofsll.login.chatbot
- Channel ID
- URI
- BOT enabled – Y/N

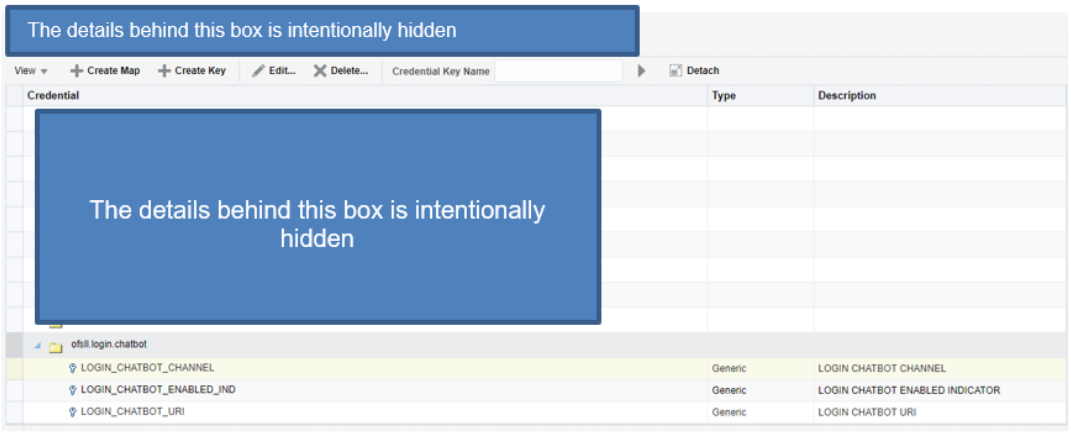
1. Login to the Weblogic server.

Figure 2-25 WebLogic Server Login



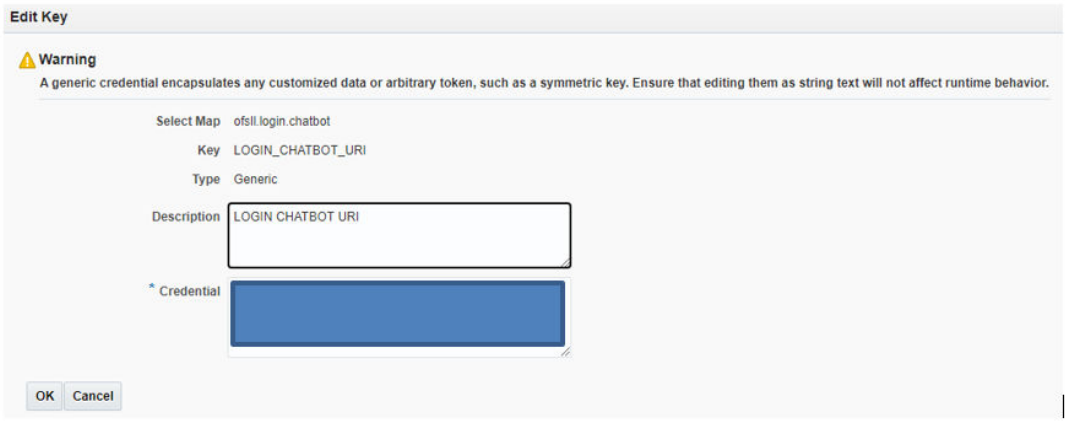
2. In the following screen, click **NEW MAP**

Figure 2-26 Map



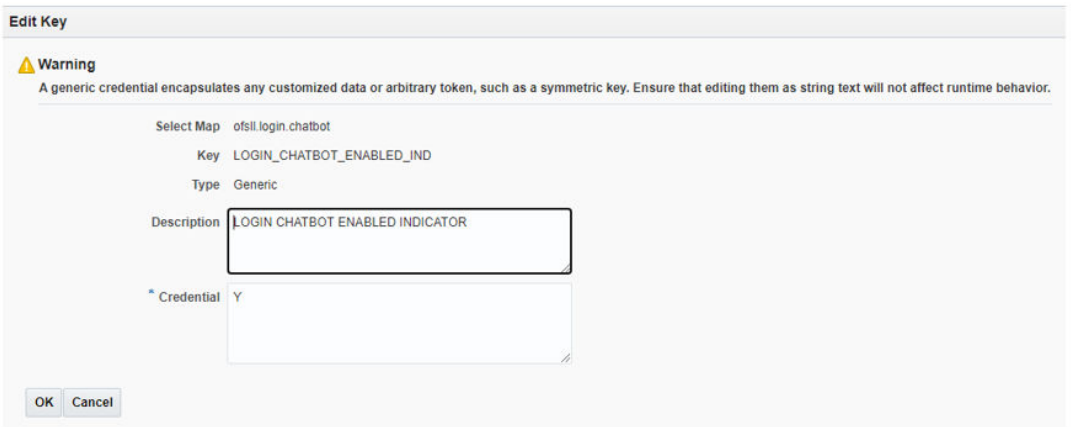
3. Add URI

Figure 2-27 URI



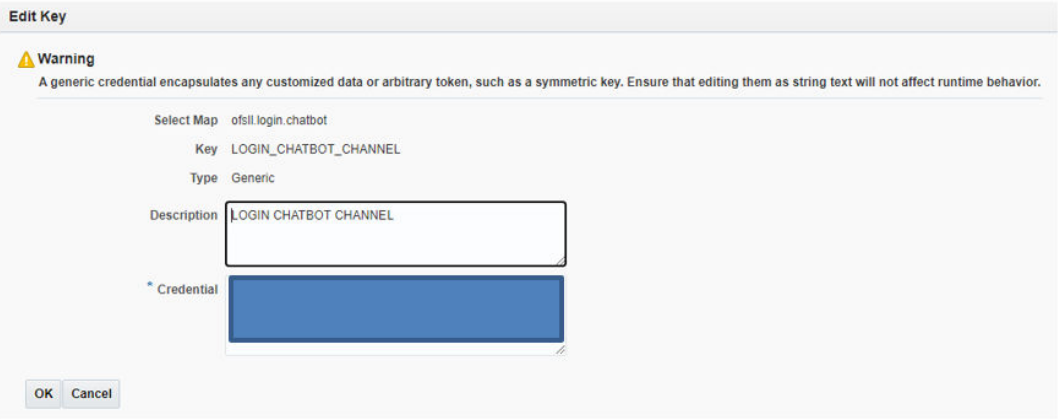
4. Enter BOT INDICATOR.

Figure 2-28 BOT Indicator



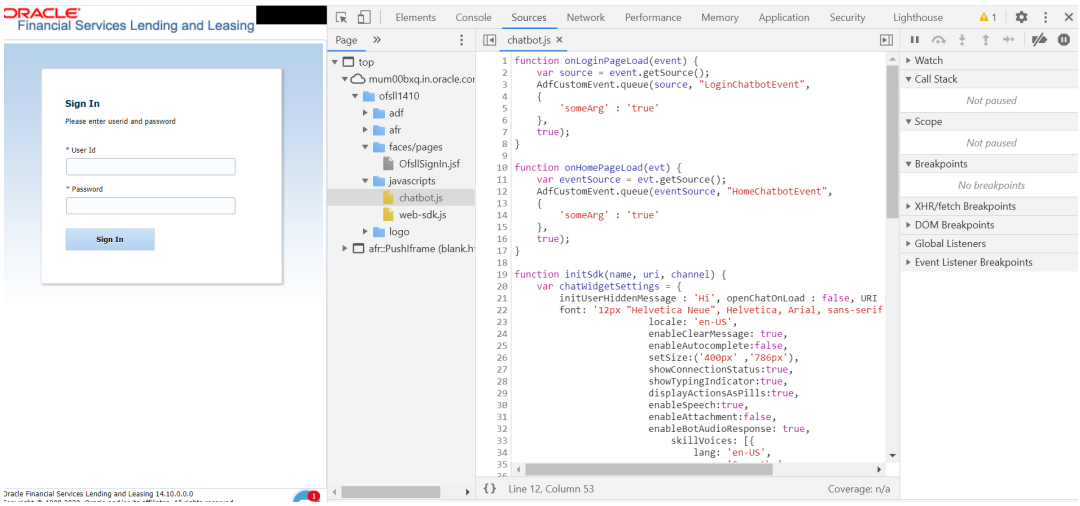
5. Enter BOT CHANNEL ID

Figure 2-29 Bot Channel ID



6. Configure the `chatbot.js` on the login page. Refer to the below image and **.js** code for reference.

Figure 2-30 Configure



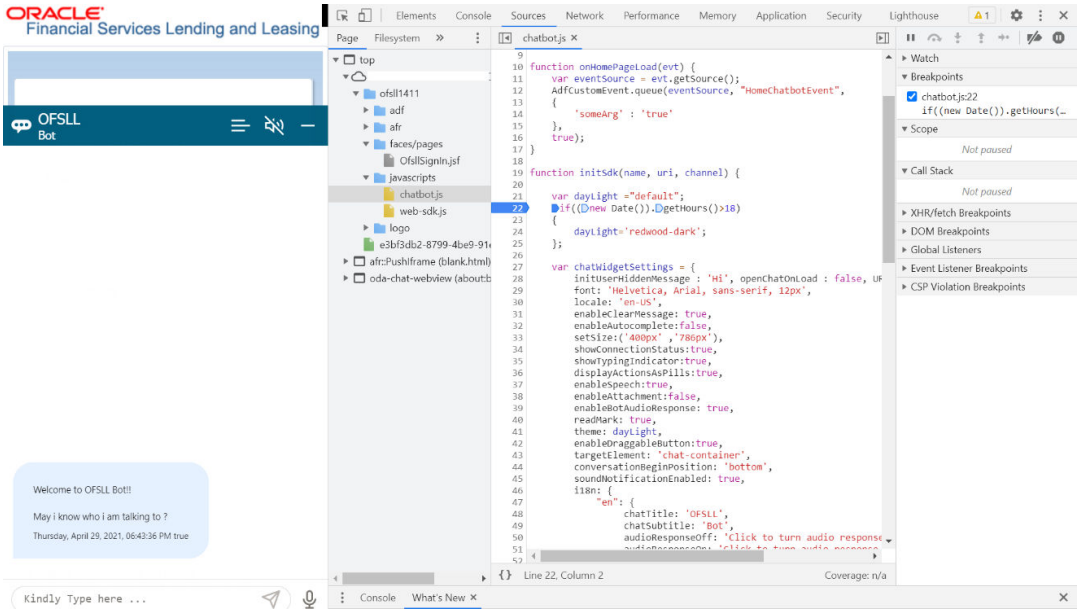
Ensure that no changes are done to the following js code:

```
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;
});
}
```

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

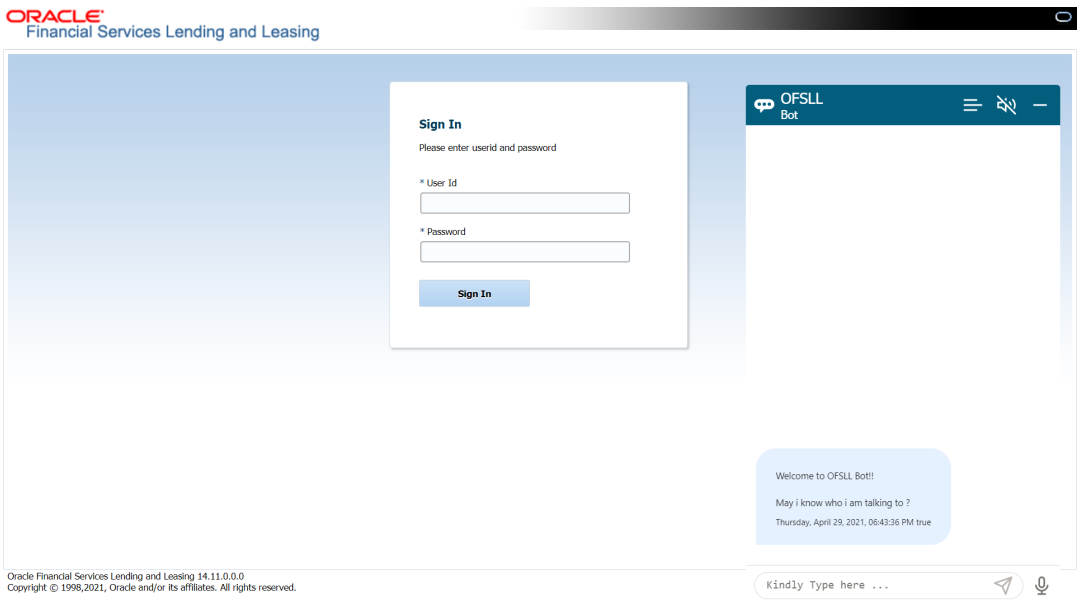
Figure 2-31 Implementation



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

   The BOT when launched from Web Application is as shown below:

Figure 2-32 BOT launch



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 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.splitSeperator==	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	Keep this blank for documentation bot
22	ofsll.username =	String	Captures the username of weblogic server	Y	Keep this blank for documentation bot
23	ofsll.pasd =	String	Captures the Password of weblogic server	Y	Keep this blank for documentation bot
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

2.8 Bot Customization

The Bot interface supports some of quick customization options as detailed below:

- Label Customization
- Bot Day/Night Theme Settings
- Floating Icon

2.8.1 Label Customization

To have a customized label for BOT interface do the following:

1. Open the `chatbot.js` file available in below location.

   `<Installation Director >\javascripts\chatbots.js`

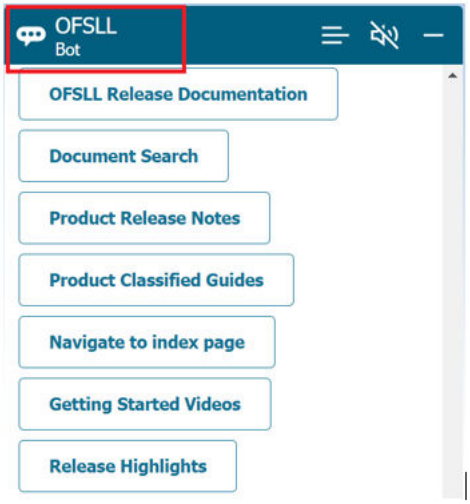
2. Locate **OFSLL** and edit the following code property as required.

```
i18n: {
    "en": {
    chatTitle: 'OFSLL', // Tool tip for title
    chatSubtitle: 'Documentation Bot', // Tool tip for Sub Title
    audioResponseOff: 'Click to turn audio response on', // Tool tip for
the speaker off button
    audioResponseOn: 'Click to turn audio response off', // Tool tip for
the speaker on button
    inputPlaceholder: 'Kindly Type here ... ', // Replaces Type a message
    send: 'Send (Enter)' // Replaces Send tool tip
        }
}
```

3. Save the `chatbots.js` file and redeploy the application.

   Once done, the BOT interface displays the customized label.

Figure 2-33 Label Customization



2.8.2 Bot Day/Night Theme Settings

The Bot interface can be customized to display in dark theme based on Day/Night time defined. Do the following:

1. Open the `chatbot.js` file available in below location.
   `<Installation Director >\javascripts\chatbots.js`
2. Locate and edit the pre-defined variable **daylight** property as required.

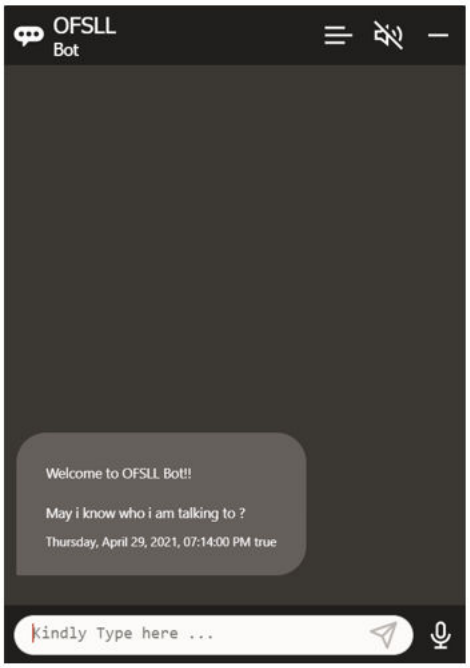
```
var dayLight ="default";
if((new Date()).getHours()>18)
{
    dayLight='redwood-dark';
};
```

Note the following:

- Here, **default** is the generic theme provided by ODA based Chabot.
- The number 18 represents 6PM, and Chabot theme changes after this time has elapsed in hours. The same can be changed as required.
- Ensure not to change the variable since the same is called in the chat Widget Settings assigned to attribute theme - Widget Settings : theme: dayLight,

Once done, the daylight theme is as displayed based on time zone:

Figure 2-34 Bot Day_Night Theme Settings



2.8.3 Floating Icon

The Bot icon when minimized can be used as floating icon and moved over the screen area. However, on maximizing the Bot window is displayed in its native position.

To have the Bot icon floating, do the following:

1. Open the `chatbot.js` file available in below location.
   <Installation Director >\javascripts\chatbots.js
2. Locate the below attribute and change the value to true.

```
enableDraggableButton:true,
```

> **Note:**
>
> When set to **true**, floating icon property is enabled and if set to **false**, floating icon property is disabled.

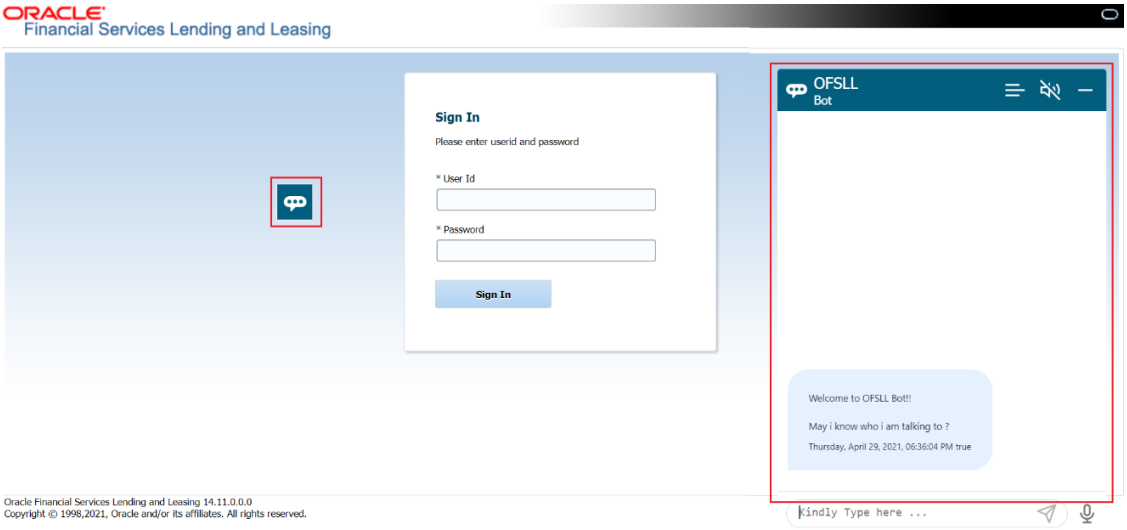



Figure 2-35 Bot Customization_Floating Icon