# Table of Contents

1. Preface ........................................................................................................................................... 4  
   1.1 Intended Audience .................................................................................................................. 4  
   1.2 Documentation Accessibility .................................................................................................... 4  
   1.3 Access to Oracle Support ......................................................................................................... 4  
   1.4 Structure ................................................................................................................................. 4  
   1.5 Related Information Sources .................................................................................................... 4  
2. Branding ......................................................................................................................................... 5  
3. Implementation ............................................................................................................................... 6  
4. CASE I : Node HTTP Server is on same setup as OHS ................................................................. 7  
5. CASE II : Node HTTP Server is different setup from OHS ......................................................... 9  
6. Handling Assets .............................................................................................................................. 10
1. Preface

1.1 Intended Audience

This document is intended for the following audience:

- Customers
- Partners

1.2 Documentation Accessibility

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

1.3 Access to Oracle Support

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

1.4 Structure

This manual is organized into the following categories:

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

Introduction provides brief information on the overall functionality covered in the User Manual.

The subsequent chapters provide information on transactions covered in the User Manual.

Each transaction is explained in the following manner:

- Introduction to the transaction
- Screenshots of the transaction
- The images of screens used in this user manual are for illustrative purpose only, to provide improved understanding of the functionality; actual screens that appear in the application may vary based on selected browser, theme, and mobile devices.
- Procedure containing steps to complete the transaction- The mandatory and conditional fields of the transaction are explained in the procedure.

If a transaction contains multiple procedures, each procedure is explained. If some functionality is present in many transactions, this functionality is explained separately.

1.5 Related Information Sources

For more information on Oracle Banking Digital Experience Release 18.1.0.0.0, refer to the following documents:

- Oracle Banking Digital Experience Licensing Guide
- Oracle Banking Digital Experience Installation Manuals
2. **Branding**

Branding describes the process of applying a product or company brand to the user interface. One can regard it as "theming with the goal to represent a certain product/company on the user interface". So the typical changes are the same as theming, with the only difference that certain visual styles and entities are directly derived from a product or company brand.
3. **Implementation**

Branding in OBDX is implemented in terms of changing CSS and changing the image resources. CSS compilation is done using `node-sass` [NodeJS package]. Largely speaking, when a brand is created, the variables taken from user input provide the overriding variables for the scss which is then compiled to CSS by `node-sass`. The image resources are taken as zip file and extracted alongside brand artifacts.

**There are two basic requirements:**
1. Node HTTP Server to generate brand artifacts.
2. HTTP Server to serve brand artifacts*.

[*Not needed if Node HTTP server is on same setup as OHS. In this case, OHS can be used to serve the artifacts.]*

**There are two cases:**
1. The Node HTTP Server is on same setup as OHS. [Recommended]
2. The Node HTTP Server is on different setup from OHS.

Out of these, the first option is recommended as it enables sharing of SCSS resources and eliminates the need of a separate HTTP server required to serve brand artifacts.
4. **CASE I : Node HTTP Server is on same setup as OHS**

1. Install NodeJS on your system.
2. Make sure to update npm by running
   
   ```
   $ npm install npm@latest -g
   ```
   
   *You need root access to install packages globally*

3. Set npm proxy
   
   ```
   $ npm config set proxy <your-proxy-here>
   ```

4. Navigate to channel directory which is managed by OHS.
   
   *Note: All paths from hereon will be with respect to channel directory*

5. Create a folder brand
   
   ```
   $ mkdir brand
   ```

6. Navigate to brand-engine
   
   ```
   $ cd brand-engine
   ```

7. Install the required packages $ npm install

8. Configure the variables in properties.json according to your implementation.
   
   a. **Port**: The port number at which the Node HTTP Server will listen.
   b. **Logs**: The path to create debug logs. Defaults to same directory.
   c. **URLs**: The list of URLs accepted by Node HTTP Server. Usually, no need to modify.
   d. **Roles**: The list of roles (directories) for which you wish to create brand.
   e. **Files**:
   f. **sass-main**: The location of main.scss
   g. **assets**: The location at which assets will be deployed.
   h. **Image Base Path**: The root path from where images will be resolved.
   i. **Brand Base Path**: The root path from where images called via CSS will be resolved.
   j. The Node HTTP Server is started in clustered mode with 4 instances by default. To change the number of instances or to change the mode, modify process.yaml

9. Start the Node HTTP Server using
   
   ```
   $ npm start
   ```

Some common commands for server management:

```
$ npm run stop

--- Stop the server
```
CASE I: Node HTTP Server is on same setup as OHS

$ npm run restart

---

Restart the server

For more details, refer documentation for pm2

10. The OHS will server all requests to /brand directory and no need to create any additional HTTP Server.

11. In System Configuration setup, set **Content Publisher URL** as URL of the Node HTTP Server and **Brand Base URL** as OHS server URL till with /brand.

![System Configuration](image)

Home
5. **CASE II : Node HTTP Server are on different setup from Application HTTP server (OHS)**

1. Follow steps 1 – 10 of first case as mentioned above with exception of step 4.

2. For step 4, you need to copy or checkout channel directory and follow the subsequent steps for the aforementioned directory.

3. After completing the above steps, keep only following folders:
   a. admin/sass/
   b. corporate/sass/
   c. framework/sass/
   d. index/sass/
   e. retail/sass/
   f. third-party/sass/
   g. wallet/sass
   h. brand-engine/
   i. brand/

4. Start HTTP Server of your choice, OHS/Apache or any other HTTP Server with document root being the channel directory. This HTTP server must run on the same setup as the node server. This HTTP server will render the styles to the Application HTTP Server (OHS) which will be on a different set up.

5. Make sure your HTTP Server sets the header **Access-Control-Allow-Origin: <origin>**, where `<origin>` is the parent OHS.

6. If you have enabled **Content Security Policy** [refer Security Guide], add the new host as `<allowed host>` in `obdx.conf` and restart the OHS.
    For more detail, refer OHS setup guide and Security Guide

7. In System Configuration setup, set Content Publisher URL as URL of the Node Server and Content Server URL as URL of the HTTP server present on the same setup and append `/brand` to it. But these urls are to be configured through the system configuration screen of the Application HTTP Server (OHS) where the changes are to be reflected.
6. Handling Assets

Assets upload is for uploading image assets for branding purposes. For adding a new image asset to override the default image asset available, following steps need to be followed.

1. Whitelist the image assets as specified in `framework/js/constants/brand-assets.js`. Keep in mind that only the images whitelisted in above file can be overridden via branding.

2. Place the changed files which you wish to update in the same directory structure as images folder. Create a zip file at the root.

   For example, if you wish to override the bank logo, the corresponding image file is `images/common/logo.svg`

   Related section in `framework/js/constants/brand-assets.js` will look like

   ```javascript
   images : ['common/logo.svg']
   ```

   And folder structure of the zip file would be, assuming zip is named `images.zip` (zip name is not an issue).

   ![Folder Structure]

   Home