**WD Application - Deployment Guide**

**WD 1.1.0**

**For Appian Software Version: 17.1+ and MySQL Database. If using a non-MySQL database, the scripts will need to be converted to appropriate database syntax.**

# Full Deployment (Clean Installation)

**Required Plugins**

* People Functions v.1.6

**Application Packages**

* WD All Contents - Release 1.1.0.zip
* WD Site Page Management Add On - Release 1.1.0.zip

**Database Scripts**

* MySQL
  + WD\_primaryDDL.sql

**Deployment Steps**

1. Deploy Plug-ins:
   1. For Cloud environments, navigate to the “Plug-ins” page of the Admin Console. Click “Deploy New Plug-ins” and search/select the plug-ins specified.
   2. For On-Premise environments: Copy required plugins listed above into <<APPIAN\_INSTALL>>/\_admin/plugins folder of your Appian instance
2. Create the table structure and insert reference data by executing the following SQL script:
   1. **WD\_primaryDDL.sql**
3. Import **WD All Contents - Release 1.1.0.zip**
   1. All objects should be imported successfully except for the WD Data Store. If a problem was detected:
      1. Navigate to the data store
      2. Select the appropriate data source
      3. Verify + Save and Publish the data store
4. Optionally, the **WD Site Page Management Add On - Release 1.1.0.zip** application package can be imported to provide the ability for end users to bookmark and reorder site tabs.
5. Go to /designer and navigate to the People tab
   1. Search for the **WD All Users** group
      1. Add any group(s) you want to expose the Widget functionality to as a child member
6. Navigate to the “User Start Pages” page of the Admin Console. Update the configuration so that specified groups are directed to the Home Page (site or tempo) upon initially logging in.
7. Update the permissions of the default widgets to be available to the WD All Users Group.
   1. Run the following SQL script:

*UPDATE wd\_widget SET groupid = <WD All Users Group ID> WHERE active = 1;*

# Post-Deployment (Configurations)

## Adding Additional Widgets

The “My Tasks” and “Reminders” widgets are available upon initial deployment of the application. To configure additional widgets, follow the below steps

1. Create a standalone SAIL section that you want to be a widget
   1. Ensure the component performs well at scale
2. Add an entry in the wd\_widget table corresponding to the new widget
   1. Review the WD Data Dictionary file for additional details about the expected values for each column
3. Navigate to the WD\_widgetChooser expression rule in Appian
   1. Call the interface created in step 1 in the choose statement at the index corresponding to the identifier that was just allocated to the widget in the database.

### Sample Widget

A sample application package has been provided to showcase the configuration steps mentioned above. By following the steps below, an additional Course widget will be accessible to the WD All Users group.

**Ensure that these steps are only performed after initial import of the application as this will overwrite previously imported objects.**

1. Create the table structure necessary for the widget by executing the following script. This also contains an INSERT statement for the new row being adding into the wd\_widget table.
   1. **WD\_sampleWidgetDDL.sql**
2. Update the permissions of the new widget to be available to the WD All Users Group.
   1. Run the following SQL script:

*UPDATE wd\_widget SET groupid = <WD All Users Group ID> WHERE name = 'Courses' AND type = 'widget';*

1. Sample course bookmarks were created for the “test.user” sample user.
   1. To update these to another in the system, run the following SQL script:

*UPDATE wdcoursebookmark SET user = '<username>' WHERE 1;*

1. Import the **WD Sample Widget - Release 1.1.0.zip** application.
   1. As before, the data store found in the package may need to be updated post deployment.
   2. The package contains the following notable objects:

|  |  |  |
| --- | --- | --- |
| **Object** | **Type** | **Description** |
| WD\_coursesWidget | Interface | Standalone SAIL section that provides the ability for a user to view and bookmark their courses and view. |
| WD\_widgetChooser | Expression | Modifies the existing object that was previously imported so that the interface above is called in position #3. This correlates to the identifier added to the new widget. |

## Adding Quick Access Links

Upon initial deployment, only the “My Profile” quick access link has been configured. Follow the below steps to initialize new links:

1. Add an entry in the wd\_widget table corresponding with the new link being added
   1. Review the WD Data Dictionary file for additional details about the expected values for each column

### Sample Links

An example SQL script has been provided to load sample quick access links into the application.

1. Execute the below SQL script to load 17 sample links into the wd\_widget table:
   1. **WD\_sampleLinkDDL.sql**
2. Update the permissions of the new links to be available to the WD All Users Group.
   1. Run the following SQL script:

*UPDATE wd\_widget SET groupid = <WD All Users Group ID> WHERE type IN ('recordLink', 'safeLink', 'tab', 'tab');*

## Adding Tabs

**This is only applicable if the WD Site Page Management Add On - Release 1.1.0.zip has been deployed on your environment.**

If the **WD Site Page Management Add On - Release 1.1.0.zip** application package has been imported in your environment, the end-users have the capability to bookmark which site tabs they want to view. New tabs can be exposed to the end-users by following the below steps:

1. Create a standalone SAIL report dashboard that you want to be exposed as a site tab
2. Add an entry in the wd\_widget table corresponding to the new tab
   1. Review the WD Data Dictionary file for additional details about the expected values for each column
3. Navigate to the WD\_chooseReport expression rule in Appian
   1. Call the interface created in step 1 in the choose statement at the index corresponding to the identifier that was just allocated to the tab in the database.

**Note**: The cons!WD\_BOOKMARK\_LIMIT determines the max number of tabs a user can bookmark at one time. This can be modified to meet the site’s need.

### Sample Tabs

An example application package has been provided to showcase the configuration steps mentioned above. By following the steps below, 4 additional tabs will be accessible to the WD All Users group.

**Ensure that these steps are only performed after initial import of the application as this will overwrite previously imported objects.**

1. Execute the below SQL script to load 4 sample tabs into the wd\_widget table:
   1. **WD\_sampleTabDDL.sql**
2. Update the permissions of the new tabs to be available to the WD All Users Group.
   1. Run the following SQL script:

*UPDATE wd\_widget SET groupid = <WD All Users Group ID> WHERE type = 'tab';*

1. Import the **WD Sample Tabs - Release 1.1.0.zip** application.
   1. The package contains the following notable objects:

|  |  |  |
| --- | --- | --- |
| **Object** | **Type** | **Description** |
| WD\_travelChart  WD\_advisorChart  WD\_healthRequestChart  WD\_homeworkChart | Interface | Standalone SAIL dashboards that simulate reports |
| WD\_chooseReport | Expression | Modifies the existing object that was previously imported so that the four interfaces listed above are included in the choose statement based upon their database identifier. |

## 