Liberty Dash

Test Plan and Low Level Design

Reference Document: Liberty Dash SRS
Created By: Team Iceberg
Date of Creation: December 1, 2016
Date of Review: December 16, 2016

Project Manager:
Maya Bergandy

Team Members:

Bryce Bodley-Gomes	Tony Gao	Kevin Silva
Matthew Hinsley	Bhavik Jain	Adrian Poveda McKay
Chenhao Huang	Zachary Tousignant	Michael Schiller

Table of Contents

1 - Introduction

This document will cover the basic testing plans for the units in our Liberty Dash system. It will include our data model to illustrate the expected data in our system, as well as how this data relates to our database tables. We will also enumerate our interfaces for the system, how they are accessed and how they function.

2 - Test Cases

This section specifies the test cases for all of the functional requirements in our Requirements Specification document.

2.1 Invoke Macro

This section contains all test cases related to invoking a Macro.

Test Case ID: Test Case #01
SRS Use Case: 3.1.2 (Invoke Macro)
Objective: User successfully invokes a “View” type Macro.
Preconditions: User is logged in.

Step #	Step Description	Test Data	Expected Result
1.	The User clicks on the “Access Database” dropdown menu.	N/A	The User has submitted a request to invoke a Macro with type View against the External Liberty Mutual Database.
2.	The User selects “Invoke Macro” from the “Access Database”.	N/A	
3.	The User selects a Macro of type “view” that they wish to execute.	N/A	
4.	The User inputs the necessary parameters for that Macro.	Valid View Macro parameters	
5.	The User adds any Tags they wish to associate with the process.	N/A	
6.	The User clicks the “Enter” button.	N/A	
7.	The User select “Yes” from the “Are you sure?”	N/A	

Postconditions: The User has successfully initiated a request to run a “View” type Macro.

Test Case ID: Test Case #02

SRS Use Case: 3.1.2 (Invoke Macro)

Objective: User fails to successfully invokes a “View” type Macro due to entry of invalid parameters

Preconditions: User is logged in.

Step #	Step Description	Test Data	Expected Result
1.	The User clicks on the “Access Database” dropdown menu.	N/A	The User has failed to submit a request to run a View type Macro against the External Liberty Mutual Database.
2.	The User selects “Invoke Macro” from the “Access Database”.	N/A	
3.	The User selects a “View” type Macro they wish to execute.	N/A	
4.	The User inputs the necessary parameters for that Macro.	Invalid parameters	
5.	The User adds any Tags they wish to associate with the process.	N/A	
6.	The User clicks the “Enter” button.	N/A	
7.	The User select “Yes” from the “Are you sure?”	N/A	

Postconditions: User’s request to run a “View” type Macro is returned as a failed process notification to the User.

Test Case ID: Test Case #03

SRS Use Case: 3.1.2 (Invoke Macro)
Objective: User invokes an “Edit” type Macro successfully.
Preconditions: User is logged in.

Step #	Step Description	Test Data	Expected Result
1.	The User clicks on the “Access Database” dropdown menu.	N/A	The User has submitted a request to run an “Edit” type Macro against the External Liberty Mutual Database.
2.	The User selects “Invoke Macro” from the “Access Database”.	N/A	
3.	The User selects an “Update” type Macro they wish to execute.	N/A	
4.	The User inputs the necessary parameters for that Macro.	Correct parameters	
5.	The User adds any Tags they wish to associate with the process.	N/A	
6.	The User clicks the “Enter” button.	N/A	
7.	The User select “Yes” from the “Are you sure?”	N/A	

Postconditions: The User has successfully initiated a request to run an “Edit” type Macro.

Test Case ID: Test Case #04

SRS Use Case: 3.1.2 (Invoke Macro)

Objective: User invokes a “Delete” type Macro successfully.

Preconditions: User is logged in.

Step #	Step Description	Test Data	Expected Result
1.	The User clicks on the “Access Database” dropdown menu.	N/A	The User has submitted a request to run an “Delete” type Macro against the External Liberty Mutual Database.
2.	The User selects “Invoke Macro” from the “Access Database”.	N/A	
3.	The User selects a “Delete” type Macro they wish to execute.	N/A	
4.	The User inputs the necessary parameters for that Macro.	Correct “Delete” type Macro parameters	
5.	The User adds any Tags they wish to associate with the process.	N/A	
6.	The User clicks the “Enter” button.	N/A	
7.	The User select “Yes” from the “Are you sure?”	N/A	

Postconditions: User has initiated a request to delete a entry.

2.2 View Changelog

This section contains all test cases related to the Changelog.

Test Case ID: Test Case #05

SRS Use Case: 3.1.3 (View Changelog)

Objective: User is able to access the Changelog.

Preconditions: User is on the “Homepage” page.

Step #	Step Description	Test Data	Expected Result
1.	User clicks on the “Access Database” dropdown menu.	N/A	The Changelog page is displayed with correct listing of changes.
2.	User selects “Changelog” from the “Access Database” drop down menu and is navigated to the “Changelog” page.	N/A	

Postconditions: User accesses the Changelog.

2.3 Peer Review

This section contains all test cases related to Peer Reviews.

Test Case ID: Test Case #06

SRS Use Case: 3.1.4 (View Submitted Peer Reviews)

Objective: User views the “Submitted Peer Reviews” page and is shown a mock peer review input into the system prior to test case.

Preconditions: User is logged in with at least one Peer Review currently existing in the system

Step #	Step Description	Test Data	Expected Result
1.	User clicks the “Peer Review” dropdown menu.	N/A	The Submitted Peer Reviews page is displayed.
2.	User selects “Submitted Peer Reviews” from the “Peer Review” dropdown menu and is navigated to the “Submitted Peer Review” page.	N/A	

Postconditions: User sees the submitted “Peer Reviews” page.

Test Case ID: Test Case #07

SRS Use Case: 3.1.5 (Accept or Reject Pending Peer Reviews)

Objective: User accepts a pending Peer Review.

Preconditions: User is logged in. User has at least one Peer Review pending their approval. A mock Peer Review with a state of "pending" is input into the system prior to test case execution.

Step #	Step Description	Test Data	Expected Result
1.	User clicks the "Peer Review" dropdown menu.	N/A	The selected Peer Review is given a state of "Accepted".
2.	User selects "Pending Peer Reviews" from the "Peer Review" dropdown menu and is navigated to the "Pending Peer Review" page.	N/A	
3.	User clicks the particular Peer Review being examined and is navigated to the "Accept or Reject Peer Reviews" page.	N/A	
4.	User selects the checkbox "Accept".	N/A	
5.	User clicks the "Submit" button.	N/A	
6.	User select "Yes" from the "Are you sure?" prompt.	N/A	

Postconditions: The selected Peer Review is given a state of "Accepted".

Test Case ID: Test Case #08

SRS Use Case: 3.1.5 (Accept or Reject Pending Peer Reviews)

Objective: User rejects a pending Peer Review.

Preconditions: User is logged in. The User has Peer Reviews pending their approval.

Step #	Step Description	Test Data	Expected Result
1.	User clicks the “Peer Review” dropdown menu.	N/A	The selected Peer Review is given a state of “Rejected”.
2.	User selects “Pending Peer Reviews” from the “Peer Review” dropdown menu and is navigated to the “Pending Peer Review” page.	N/A	
3.	User clicks the particular Peer Review being examined and is navigated to the “Accept or Reject Peer Reviews” page.	N/A	
4.	User selects the checkbox “Reject”.	N/A	
5.	User enters the reasoning for rejecting the request into the textbox.	“This is a mock reasoning of rejection”	
6.	User clicks the “Submit” button.	N/A	
7.	User select “Yes” from the “Are you sure?” Prompt.	N/A	

Postconditions: The selected Peer Review is given a state of “Rejected”.

2.4 Process Management

This section contains all test cases related to processes.

Test Case ID: Test Case #09

SRS Use Case: 3.1.6 (View Pending Processes)

Objective: User views Pending Processes.

Preconditions: User is logged in. A mock pending process has been input into the system prior to running this test case.

Step #	Step Description	Test Data	Expected Result
1.	User clicks the “Processes” dropdown menu.	N/A	The “Pending Processes” page is displayed.
2.	User selects “Pending” from the “Processes” menu and is navigated to the “Pending Processes” page.	N/A	

Postconditions: The Pending Processes are displayed.

Test Case ID: Test Case #10

SRS Use Case: 3.1.7 (View Processes On Hold)
Objective: User is able to view Processes on Hold.
Preconditions: User is logged in. A mock Process On Hold would be added to the system prior to test case to ensure that the Process On Hold is displayed correctly.

Step #	Step Description	Test Data	Expected Result
1.	User clicks on the “Processes” dropdown menu.	N/A	The Processes on Hold are displayed.
2.	User selects “Hold” from the “Processes” menu and is navigated to the “Processes on Hold” page.	N/A	

Postconditions: User is able to view Processes on Hold.

Test Case ID: Test Case #11

SRS Use Case: 3.1.8 (View Running Processes)
Objective: User is able to view Running Processes.
Preconditions: User is logged in. A mock process with a state of “running” will be added to the system prior to the test case execution.

Step #	Step Description	Test Data	Expected Result
1.	User clicks on the “Processes” dropdown menu.	N/A	The Running Processes are displayed.
2.	User selects “Running” from the “Processes” menu and is navigated to the “Running Processes” page.	N/A	

Postconditions: User is able to view Running Processes.

Test Case ID: Test Case #12

SRS Use Case: 3.1.9 (View Successful Processes)
Objective: User is able to view Successful Processes.
Preconditions: User is logged in. A mock completed process has been added to the database prior to test case to ensure that the Successful Process is displayed correctly.

Step #	Step Description	Test Data	Expected Result
1.	User clicks on the “Processes” dropdown menu.	N/A	The Successful Processes are displayed.
2.	User selects “Successful” from the “Processes” menu and is navigated to the “Successful Processes” page.	N/A	

Postconditions: User is able to view Successful Processes.

Test Case ID: Test Case #13

SRS Use Case: 3.1.10 (View Failed Processes)
Objective: User is able to view Failed Processes.
Preconditions: User is logged in. A mock Failed Process has been added to the database prior to test case to ensure that the Failed Process is displayed correctly.

Step #	Step Description	Test Data	Expected Result
1.	User clicks on the “Processes” dropdown menu.	N/A	The Failed Processes are displayed.
2.	User selects “Resubmit” from the Processes Failure Error Message.	N/A	

Postconditions: User is able to view Failed Processes.

Test Case ID: Test Case #14

SRS Use Case: 3.1.11 (Restart Processes)

Objective: User is able to correct the process and restart it.

Preconditions: User is logged in. User is on the "Failed Processes" page. There is at least one failed process to restart (a mock case is input into system prior to test case to ensure this).

Step #	Step Description	Test Data	Expected Result
1.	User selects failed process and selects resubmit.	N/A	The process is removed from the failed process list and added to the pending process list.
2.	User fixes errors in parameters	Correct the erroneous parameters	

Postconditions: The process is removed from the failed process listing and added to the pending process list.

Test Case ID: Test Case #15

SRS Use Case: 3.1.12 (Search Processes)

Objective: The system returns the correct process associated with the ID.

Preconditions: The user is logged in and on the "Search Processes" page. A mock process with ID of 01 has been input into the system prior to running of test case.

Step #	Step Description	Test Data	Expected Result
1.	The user inputs the ID of the process they wish to search for into the search bar.	"01"	Process data is displayed.

Postconditions: Data associated with the requested process is returned.

Test Case ID: Test Case #16

SRS Use Case: 3.1.13 (Cancel Process)

Objective: User cancels a queued process.

Preconditions: User is logged in, and on the “Pending Processes” page. Mock process is in the queued process list.

Step #	Step Description	Test Data	Expected Result
1.	The User clicks on a desired Pending Process.	Select a process from the list.	The selected process is removed from the queue of pending processes.
2.	In the popup displayed, the User selects “Cancel process”.	N/A	
3.	The User select “Yes” from the “Are you sure?” Prompt	N/A	

Postconditions: The selected pending process is canceled.

2.5 Login/Logout

This section contains all test cases related to user Login and Logout.

Test Case ID: Test Case #17

SRS Use Case: 3.1.14 (Login to Account)
Objective: The user is successfully logged into system.
Preconditions: The user is on the “Login” page. The user has a valid account.

Step #	Step Description	Test Data	Expected Result
1.	The User enters username.	“testuser”	The User is logged in to the system, the Main Menu page is displayed.
2.	The User enters password.	“12345”	
3.	The User clicks “Enter” button.	N/A	

Postconditions: The User is logged in.

Test Case ID: Test Case #18

SRS Use Case: 3.1.14 (Login to Account)
Objective: Failed login to system: Invalid Username
Preconditions: The User is on the “Login” page.

Step #	Step Description	Test Data	Expected Result
1.	The User enters username.	“blablabla”	The User is not logged in, and a popup notification is displayed indicating an error in login credentials.
2.	The User enters password.	“123456”	
3.	The User clicks “Enter” button.	N/A	

Post conditions: The User is returned to “Login” page.

Test Case ID: Test Case #19

SRS Use Case: 3.1.14 (Login to Account)
Objective: Failed login to system: Invalid Password
Preconditions: The User is on the “Login” page.

Step #	Step Description	Test Data	Expected Result
1.	The User enters username.	“testuser”	The User is not logged in, and a popup notification is displayed indicating an error in login credentials.
2.	The User enters password.	“123a456”	
3.	The User clicks “Enter” button.	N/A	

Post conditions: The User is returned to “Login” page

Test Case ID: Test Case #20

SRS Use Case: 3.1.15 (Log out of Account)
Objective: The user is able to log out.
Preconditions: The user is logged in.

Step #	Step Description	Test Data	Expected Result
1.	The user hovers the mouse over the User Accounts Icon	N/A	The user is logged out of there account and are no longer able to access the system.
2.	The user selects the menu item “Log Out”	N/A	
3.	The user select “Yes” from the “Are you sure?” Prompt	N/A	

Post conditions: The user is returned to the “Login Page”.

2.6 Notifications

This section contains all test cases related to notifications.

Test Case ID: Test Case #21

SRS Use Case: 3.1.16 (View Notifications)
Objective: The User is able to view notifications waiting for them.
Preconditions: The User has notifications waiting for them.

Step #	Step Description	Test Data	Expected Result
1.	The User hovers the mouse over the User Accounts icon.	N/A	The User is shown pending peer review requests, and submitted peer reviews that were rejected.
2.	The User selects the menu item “Notifications”	N/A	

Post conditions: The User has reviewed notifications.

2.7 Manage Application Settings

This section contains all test cases related to managing the application's settings.

Test Case ID: Test Case #22

SRS Use Case: 3.1.17 (View or Edit Settings)
Objective: The User edits settings and the changes are saved.
Preconditions: The User is on the "Settings" page.

Step #	Step Description	Test Data	Expected Result
1.	The User checks/unchecks any settings that the User wishes to update, and clicks Save .	N/A	
2.	Toggle Peer Review: An Admin User has a unique setting to globally require Peer Review for all General Users. By selecting this option, General Users would no longer be able to toggle off Peer Review when submitting requests to edit the External Liberty Mutual Database.	Select "Peer Review is Required"	General Users are no longer able to bypass Peer Reviews
3.	Change Environment: Any User can change the Environment that they are working in by selecting either "Dev" (Development), "Test", "QA" (Quality Assurance), or "Prod" (Production).	"Select new Environment"	The user's now changed into the new Environment.

Post conditions: The User is returned to "Login" page.

3 - Interfaces

This section specifies all interfaces between our Liberty Dash system and external systems. This interface specification includes the interface's functions, the data passed through the functional parameters, the values of these functions, and the communication protocols used to connect the systems.

3.1 Java Database Connectivity (JDBC)

The following function establishes a connection between our system and databases, including both the Internal Liberty Dash Database and the External Liberty Mutual Database.

1. getExecutionBinding(user, password, connectString)
    - Parameters:
        - Username
        - Password
        - JDBC Connect String.
    - Return Values: an anonymous function executeSql that takes (sql, cb) as parameters. 'Cb' is a function of two parameters, (err, result) objects.
    - Communication Protocols: JDBC

3.2 Macros

The following functions relate to updating either the External Liberty Mutual Database, or retrieving fields from the Internal Liberty Dash Database. The communication protocols for all of the macros are through JDBC.

1. **PM_EDW_META_D.M_DL_DR_SCHED_RN**
    - Parameters:
        - p_run_nme[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

2. **PM_EDW_META_D.M_UD_DR_SCHED_START_RN_AID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_audt_id[NUMBER]
        - p_sched_start[DATE]
    - Return Values: Void
    - Communication Protocols: JDBC

3. **PM_EDW_META_D.M_UD_DR_SCHED_STTS_RN_AID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_audt_id[NUMBER]
        - p_status[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

4. **PM_EDW_META_D.M_UD_DR_SCHED_VAL_END_RN_AID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_audt_id[NUMBER]
        - p_val_end[DATE]
    - Return Values: Void
    - Communication Protocols: JDBC

5. **PM_EDW_META_D.M_UD_DR_SCHED_VAL_START_RN_AID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_audt_id[NUMBER]
        - p_val_start[DATE]
    - Return Values: Void
    - Communication Protocols: JDBC

6. **PM_EDW_META_D.M_UD_DR_SCHED_SLA_AID**
    - Parameters:
        - SLA_DT[DATE]
        - SLA_TIME[TIME]
        - AUDT_ID[NUMBER]
        - PERF_DTL_DESC[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

7. **PM_EDW_META_D.M_UD_DR_SCHED_SLA_RN**
    - Parameters:
        - SLA_TME[DATE]
        - RUN_NME[VARCHAR]
    - Return Values: Void
    - Communication Protocols: JDBC

8. **PM_EDW_META_D.M_DL_DR_STEP_RN**
    - Parameters:
        - p_run_nme[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

9. **PM_EDW_META_D.M_DL_DR_STEP_RN_GN**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_grp_nbr[NUMBER]
    - Return Values: Void
    - Communication Protocols: JDBC

10. **PM_EDW_META_D.M_DL_DR_STEP_RN_SID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_drvr_step_id[NUMBER]
    - Return Values: Void
    - Communication Protocols: JDBC

11. **PM_EDW_META_D.M_DL_DR_STEP_DTL_RN**
    - Parameters:
        - p_run_nme[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

12. **PM_EDW_META_D.M_UD_DR_STEP_DTL_RN_GN**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_grp_nbr[NUMBER]
        - p_status[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

13. **PM_EDW_META_D.M_UD_DR_STEP_DTL_RN_STPDTLID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_drvr_step_dtl_id[NUMBER]
        - p_status[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

14. **PM_EDW_META_D.M_UD_DR_STEP_ASI_SID**
    - Parameters:
        - p_drvr_step_id[NUMBER]
        - p_actv_step_ind[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

15. **PM_EDW_META_D.M_UD_DR_STEP_ASI_RN_SID**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_drvr_step_id[NUMBER]
        - p_actv_step_ind[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

16. **PM_EDW_META_D.M_UD_DR_STEP_ASI_RN**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_actv_step_ind[VARCHAR]
    - Return Values: Void
    - Communication Protocols: JDBC

17. **PM_EDW_META_D.M_UD_DR_STEP_ASI_RN_GN**
    - Parameters:
        - p_run_nme[VARCHAR2]
        - p_grp_nbr[NUMBER]
        - p_actv_step_ind[VARCHAR2]
    - Return Values: Void
    - Communication Protocols: JDBC

18. **PEER_REVIEW.PENDING_REVIEWS**
    - Parameters:
        - Void
    - Return Values:
        - macroInstanceID[NUMBER]
        - reviewerID[NUMBER]
        - state[VARCHAR2]
        - peerReviewComment[VARCHAR2]
        - dateReviewed[NUMBER]
        - TimeReviewed[NUMBER]
    - Communication Protocols: JDBC

19. PEER_REVIEW.SUBMITTED_REVIEWS

- Parameters:
    - Void
- Return Values:
    - macroInstanceID[NUMBER]
    - reviewerID[NUMBER]
    - state[VARCHAR2]
    - peerReviewComment[VARCHAR2]
    - dateReviewed[NUMBER]
    - TimeReviewed[NUMBER]
- Communication Protocols: JDBC

20. PROCESSES.RUNNING

- Parameters:
    - Void
- Return Values:
    - ID[NUMBER]
    - OwnerName[VARCHAR2]
    - Reviewer[VARCHAR2]
    - TimeCreated[DATE]
    - TimeStarted[DATE]
    - Dependencies[TEXT]
- Communication Protocols: JDBC

21. PROCESSES.HOLD

- Parameters:
    - Void
- Return Values:
    - ID[NUMBER]
    - OwnerName[VARCHAR2]
    - Reviewer[VARCHAR2]
    - TimeCreated[DATE]
    - TimeProjected[DATE]
    - Dependencies[TEXT]
- Communication Protocols: JDBC

22. PROCESSES.PENDING

- Parameters:
    - Void
- Return Values:
    - ID[NUMBER]
    - OwnerName[VARCHAR2]
    - Reviewer[VARCHAR2]
    - TimeCreated[DATE]
    - Dependencies[TEXT]
- Communication Protocols: JDBC

23. PROCESSES.SUCCESSFUL

- Parameters:
    - Void
- Return Values:
    - ID[NUMBER]
    - OwnerName[VARCHAR2]
    - Reviewer[VARCHAR2]
    - TimeCreated[DATE]
    - TimeCompleted[DATE]
    - Dependencies[TEXT]
- Communication Protocols: JDBC

24. PROCESSES.FAILED

- Parameters:
    - Void
- Return Values:
    - ID[NUMBER]
    - OwnerName[VARCHAR2]
    - Reviewer[VARCHAR2]
    - TimeCreated[DATE]
    - TimeFailed[DATE]
    - Dependencies[TEXT]
- Communication Protocols: JDBC

25. **PROCESSES.SEARCH**
    - Parameters:
        - InputParams[TEXT]
    - Return Values:
        - ID[NUMBER]
        - OwnerName[VARCHAR2]
        - Reviewer[VARCHAR2]
        - *TimeCreated[DATE]
        - *TimeProjected[DATE]
        - *TimeStarted[DATE]
        - *TimeCompleted[DATE]
        - *TimeFailed[DATE]
        - Dependencies[TEXT]

      * indicates that not all of these values will necessarily be returned - what is returned will depend on the InputParams
    - Communication Protocols: JDBC

3.3 Azure Active Directory Authentication Library

The following processes connect our system to Active Directory and allow us to authenticate users in our Liberty Dash system for security purposes.

1. **ADAL.processAdalCallback()**
    - Side Effects: Redirects to adal.CONSTANTS.STORAGE.START_PAGE
    - Parameters:
        - Void
    - Return Values:
        - Void
    - Communication Protocols: Azure Active Directory

2. **ADAL.isAuthenticated()**
    - Side Effects: Checks if the current user is logged into active directory, starts the token acquisition process for a login, processAdalCallback() is called after.
    - Parameters:
        - Void
    - Return Values:
        - True/False
    - Communication Protocols: Azure Active Directory

3. **ADAL.adalRequest(settings)**
    - Side Effects: Makes a request to an Azure Active Directory protected resource. This can be used to get user list with the proper parameters. The resource can return any object as data
    - Parameters:
        - Settings JS Object
    - Return Values:
        - JS Object
    - Communication Protocols: Azure Active Directory

4 - Data Model

Figure 4.1 below shows the relationships of the entities stored in the Internal Liberty Dash Database.

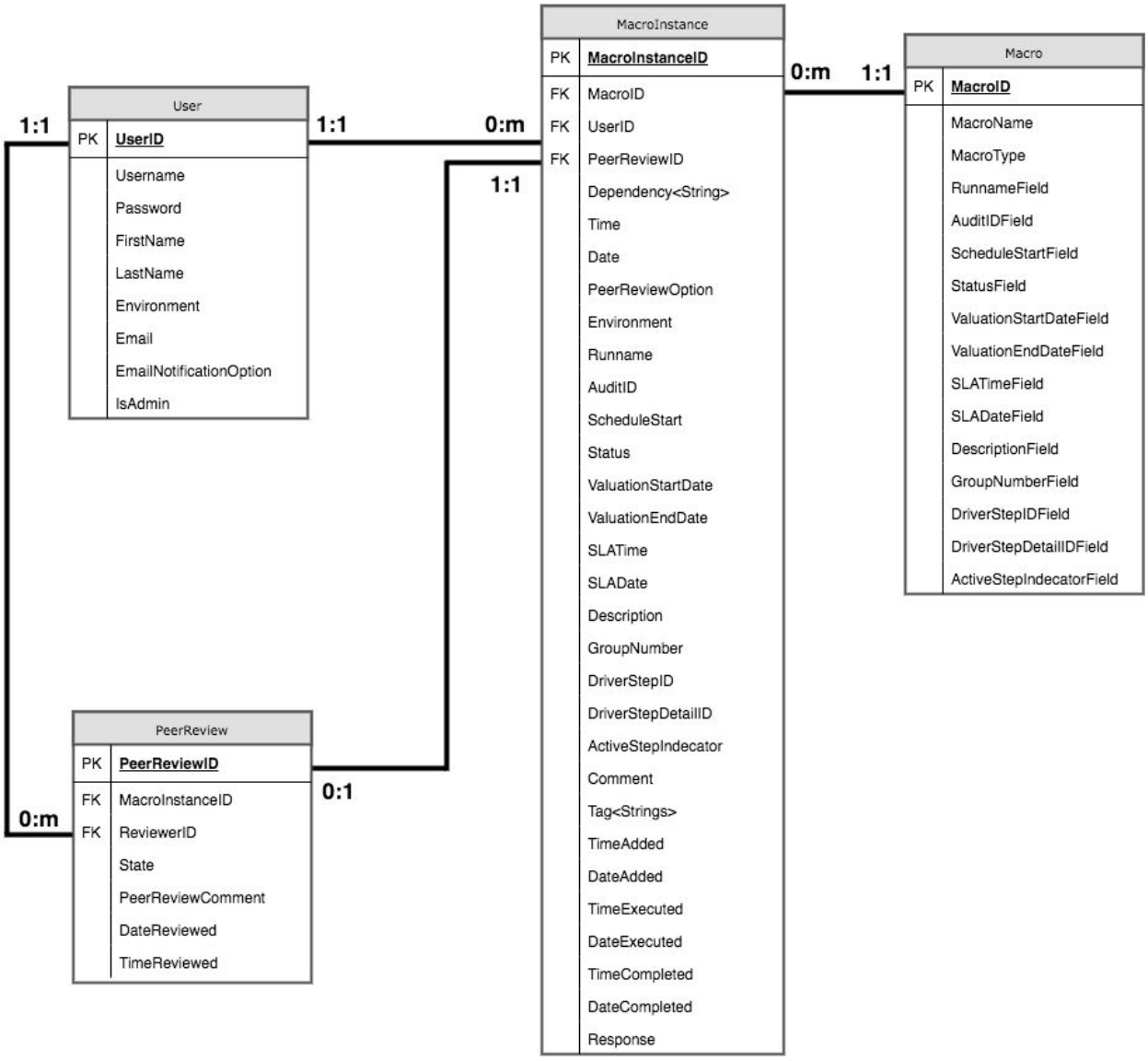

Figure 4.1: Internal Liberty Dash Database ER Diagram

The structure of the Internal Liberty Dash Database consists of multiple relational database tables. The data dictionary for these tables are outlined below.

Entity Name	Entity Description	Column Name	Column Description	Data Type	Length
User	A user with access to all basic functionalities of the system.	UserID	The unique identification number for each individual User	Integer	25
		UserName	The unique account name of the User used to log in	Varchar	25
		Password	The password for the User to log in	Varchar	25
		FirstName	First name of the User	Varchar	25
		LastName	Last name of the User	Varchar	25
		Environment	The Environment that the User is working in (D: Development, T: Test, Q: Quality Assuance, P: Procudtion)	Character	1
		Email	The email address of the User used for Notifications	Varchar	50
		EmailNotificationOption	Whether or not the User wishes to receive email notifications	Boolean	1
		IsAdmin	Whether or not the User is an Admin User	Boolean	1

Table 4.1: User Entity

Entity Name	Entity Description	Column Name	Column Description	Data Type	Length
PeerReview	The review of process by which submitted tasks are reviewed and either approved or rejected by other user in the system.	PeerReviewID	For the unique identification of Peer Review Infomation	Integer	10
		MacroInstanceID	The ID of the Macro Instance the Peer Review is related to	Integer	10
		ReviewerID	The UserID of the User that reviews the Peer Review	Integer	25
		State	The state of the Peer Review (P: Pending, A: Accepted, R: Rejected)	Character	1
		PeerReviewComment	The comment for the Peer Review (reason for rejection)	Varchar	512
		DateReviewed	The date the Macro is reviewed	Date	
		TimeReviewed	The time the Macro is reviewed	Time	

Table 4.2: Peer Review Entity

Entity Name	Entity Description	Column Name	Column Description	Data Type	Length
MacroInstance	A Macro Instance is created by an User with a value for each parameter	MacroInstanceID	For the unique identification of Macro Instance information	Integer	10
		MacroID	The ID of the Macro	Integer	10
		UserID	The UserID of the owner of the Macro Instance	Varchar	25
		PeerReviewID	The ID of the Peer Review the Macro Instance is related to	Integer	10
		Dependency<String>	The list of dependencies that the Macro Instance is related to	List<String>	
		Time	The time the Macro Instance is submitted	Time	
		Date	The date the Macro Instance is submitted	Date	
		PeerReviewOption	Whether or not the Macro Instance needs to be Peer Reviewed	Boolean	1
		Environment	The environment that the Macro Instance is working in (D:Development, T: Test, Q: Quality Assuance, P: Procudtion)	Character	1
		Runname	The runname parameter for the Macro Instance	Varchar	52
		AuditID	Audit ID parameter for the Macro Instance	Varchar	10
		ScheduleStart	Schedule start parameter for the Macro Instance	Date	
		Status	Status parameter for the Macro Instance	Varchar	26
		ValuationStartDate	Valuation start date parameter for the Macro Instance	Date	
		ValuationEndDate	Valuation end date parameter for the Macro Instance	Date	
		SLATime	SLA time parameter for the Macro Instance	Time	
		SLADate	SLA date parameter for the Macro Instance	Date	
		Description	Description parameter for the Macro Instance	Varchar	256
		GroupNumber	Group number parameter for the Macro Instance	Varchar	5
		DriverStepID	Driver step ID parameter for the Macro Instance	Varchar	20
		DriverStepDetailID	Driver step detail ID parameter for the Macro Instance	Varchar	26
		ActiveStepIndicator	Active step indicator parameter for the Macro Instance	Varchar	26
		Comment	The comment of the change	Varchar	512
		Tag<Strings>	A list of the Tags the User added to associate with the process.	List<String>	
		TimeAdded	The time the process is added	Time	
		DateAdded	The date the process is added	Date	
		TimeExecuted	The time the process is executed	Time	
		DateExecuted	The date the process is executed	Date	
		TimeCompleted	The time the process is completed	Time	
		DateCompleted	The date the process is completed	Date	
		Response	The response from the External Liberty Mutual Database (Accepted, Rejected or Pending)	Varchar	10

Table 4.3: Macro Instance Entity

Entity Name	Entity Description	Column Name	Column Description	Data Type	Length
Macro	Macro Types that are supported and requrie parameters fields	MacroID	For the unique identification of Macro Infomation	Integer	10
		MacroName	The name of the Macro	Varchar	255
		MacroType	The type of the Macro (D:Delete, E: Edit, V:View)	Character	1
		RunnameField	Runname field is required for the Macro	Boolean	1
		AuditIDField	Audit ID parameter field is required for the Macro	Boolean	1
		ScheduleStartField	Schedule start parameter field is required for the Macro	Boolean	1
		StatusField	Status parameter field is required for the Macro	Boolean	1
		ValuationStartDateField	Valuation start date parameter field is required for the Macro	Boolean	1
		ValuationEndDateField	Valuation end date parameter field is required for the Macro	Boolean	1
		SLATimeField	SLA time parameter field is required for the Macro	Boolean	1
		SLADateField	SLA date parameter field is required for the Macro	Boolean	1
		DescriptionField	Description parameter field is required for the Macro	Boolean	1
		GroupNumberField	Group Number parameter field is required for the Macro	Boolean	1
		DriverStepIDField	Driver step ID parameter field is required for the Macro	Boolean	1
		DriverStepDetailIDField	Driver step detail ID parameter field is required for the Macro	Boolean	1
		ActiveStepIndicatorField	Active step indicator parameter field is required for the Macro	Boolean	1

Table 4.4: Macro Entity

5 - Glossary

Active Directory: A database that stores all of an organization's user login credentials.

Admin User: A user with access to the same functionality as a General User, with additional permissions.

Changelog: A table stored in the Internal Liberty Dash Database. This table documents the changes made through the system on the External Liberty Mutual Database. The Changelog table includes information on: what was run (Macro and parameters), who peer reviewed (if applicable), and a comment regarding why this Macro was invoked.

External Liberty Mutual Database: The database from Liberty Mutual that holds Liberty Mutual's metadata. The Liberty Dash system will interact with this database, as it is external to our system.

General User: A user with access to all basic functionalities of the system.

Internal Liberty Dash Database: The database internal to our system that stores the Changelog, processes information, and Peer Reviews that have been requested.

Peer Review: The process by which submitted tasks are reviewed and either approved or rejected by other user in the system. By default for all macros Peer Review will be selected. However, if an administrator allows modifies the settings to allow developers to toggle it, when attempting to invoke a macro the developers will be unable to uncheck the "Peer Review" button to bypass this system in emergencies.

User: A General User or Admin User who accesses our system by logging in through the Microsoft Azure Active Directory library.