SOFTWARE REQUIREMENTS SPECIFICATION (SRS) FOR

Kenny's Krew

Digital Dash

Version 2.0
November 18, 2016

Prepared by:
Craig Norton, David Ter-Ovanesyan, Dylan Gelinas, Elvis Chen, Hoang Nguyen, Ian Torres, Joseph Geneva, Kavya Krishna, Siddharth Parasnis, Thanh Pham

Managed by: Kenneth Tsui

Table of Contents

1 Introduction

1.1 Purpose

The purpose of the project is to create a GUI that will automate the current manual work of running macros and MySQL statements for the Liberty Mutual team. This will make it more efficient for less technical users and professionals as well as making the application work faster and reducing risk.

1.2 Product Scope

The GUI should be able to manipulate the Liberty Mutual control tables as well as implement solutions for the Audit & Control (A&C) Manipulation Macros.

1.3 Intended Audience

The application is exclusive to the Liberty Mutual IT Department, which consists of the *admins* and *developers* that run and manage the macros.

1.4 User Characteristics

User Types	Description
1) Admin	Ability to sign in and execute queries without needing permission
2) Developer	Ability to sign in and submit queries to an admin to be looked over and executed

2 Functional Requirements

2.1 Login via Azure Active Directory

ID: 2.1

Title: Login via Azure Active Directory

Actor: *Admin, Developer*

Trigger: User clicks the "Login*"* [See Figure 1]

Precondition: Must be on the *Login* page

User's Goal: To login into their account via Azure Active Directory File System Authentication

Steps:

1. User fills out the username field
2. User fills out the password field
3. User chooses Azure Active Directory Environment (Development, Test, QA, and Production)
4. User presses "login"
5. Azure Active Directory authenticates user information
6. User is logged in and redirected to the homepage [See Figure 2]

Alternate Flow: None

Error Conditions:

4a. The User inputs incorrect username and/or password, which will cause the application to display an alert box that displays "incorrect username or password". Afterwards, the user will be redirected to the login page.

2.2 Update Driver Table

ID: 2.2

Title: Update Driver Table

Actor: *Admin, Developer*

Trigger: User selects Update option

Precondition: Must be logged in

User's Goal: Send a request or override for an Update to the *driver tables* and update the journal with relevant parameters

Steps:

1. User selects an update option from a list of options [See Figure 3]
2. User fills in displayed add macro parameters
    1. User fills in driver step id if it pertains to macro option
    2. User fills in audit id if it pertains to macro option
    3. User fills in schedule start time if it pertains to macro option
    4. User fills in run name if it pertains to macro option
    5. User fills in status if it pertains to macro option
    6. User fills in valuation start date time if it pertains to macro option
    7. User fills in valuation end date time if it pertains to macro option

    8. User fills in SLA date if it pertains to macro option
    9. User fills in SLA time if it pertains to macro option
    10. User fills in group number if it pertains to macro option
    11. User fills in active step indicator if it pertains to macro option
    12. User fills in driver step detail id if it pertains to macro option

3. User selects “Execute”
4. Prompt asks “Are you sure?”
5. User selects “yes”
6. Macro parameters are encapsulated into an update object
7. Update object is written to Peer Review file for pending approval
8. Journal entry is written to log file with specified data
   (Log file can be seen via “Log” option):
   “Peer Review, User Name, UserID, Macro Name, MacroID, Macro Parameters, Date”
9. User sees alert box saying that request has been processed. An email is sent to all *admins* of macro request.

<u>Alternate Flow</u>:

1a. User selects another option after selecting current option

3a. User selects “Urgent Exec”

3b. Prompt asks “Are you sure?”

3c. User selects “yes”

3d. Macro parameters are encapsulated into an update object

3e. Journal entry is written to log file with specified data
(Log file can be seen via “Log” option):
“Override, User Name, UserID, Macro Name, MacroID, Macro
Parameters, Date”

3f. Update object is written to the *driver tables*

5a. User selects “no” on the “Are you sure?” prompt. The user is then able to continue editing parameters.

<u>Error Condition</u>:

5a. User selects execute without inputting all relevant parameters. System displays error message with empty relevant parameters emphasized in a message with an example of the field input for empty parameters. HTML5 detects incorrect error based on inputs given pattern and process is stopped and is sent back to step 2.

5b. User selects execute but has input the wrong type in one or many of the input parameters. System displays error message with incorrect input parameters emphasized in a message with an example of the field input for incorrect parameters. HTML5 detects incorrect error based on the input's text pattern, the process ceases, and the user is sent back to step 2.

2.3 Delete Driver Entry

ID: 2.3

Title: Delete Driver Entry

Actor: *Admin, Developer*

Trigger: User selects *delete* option

Precondition: Must be logged in

User's Goal: Send a request, or override to delete from the *driver tables* and update the journal with relevant parameters

Steps:

1. User selects a delete option from a list of options [See Figure 4]
2. User fills in displayed delete macro parameters
    1. User fills in driver run name if it pertains to macro option
    2. User fills in group number if it pertains to macro option
    3. User fills in driver step id if it pertains to macro option
3. User selects "Execute"
4. Prompt asks "Are you sure?"
5. User selects "yes"
6. Macro parameters are encapsulated into a delete object
7. Delete object is written to Peer Review file for pending approval
8. Journal entry is written to log file with specified data (Log file can be seen via "Log" option):
   "Peer Review, User Name, UserID, Macro Name, MacroID, Macro Parameters, Date"
9. User sees alert box saying that request has been processed. An email is sent to all *admins* of macro request.

Alternate Flow:

1a. User selects another option after selecting current option

3a. User selects "Urgent Exec"

3b. Prompt asks “Are you sure?”

3c. User selects “yes”

3d. Macro parameters are encapsulated into a delete object

3e. Journal entry is written to log file with specified data
(Log file can be seen via “Log” option):
“Override, User Name, UserID, Macro Name, MacroID, Macro Parameters, Date”

3f. Delete object is written to the *driver tables*

5a. User selects “no” on the “Are you sure?” prompt. The user is then able to continue editing parameters

Error Condition:

3a. User selects execute without inputting all relevant parameters. System displays error message with empty relevant parameters emphasized in a message with an example of the field input for empty parameters. HTML5 detects incorrect error based on the input’s text pattern, the process ceases, and the user is sent back to step 2.

3b. User selects execute but has input the wrong type in one or many of the input parameters. System displays error message with incorrect input parameters emphasized in a message with an example of the field input for incorrect parameters. HTML5 detects incorrect error based on the input’s text pattern, the process ceases, and the user is sent back to step 2.

2.4 Add Driver to Table

ID: 2.4

Title: Add Driver to Table

Actor: *Admin*, *Developer*

Trigger: User selects an add option [See Figure 5]

Precondition: Must be logged in

User’s Goal: Send a request or override to add to the *driver tables* and update the journal with relevant parameters

Steps:

1. User selects a add option from a list of options [See Figure 5]
2. User fills in displayed add macro parameters
   1. User selects technology option
   2. User selects category option

3. User selects use case option
3. User selects CMD_text option
3. User selects “Execute”
4. Prompt asks “Are you sure?”
5. User selects “yes”
6. Macro parameters are encapsulated into an add object
7. Add object is written to Peer Review file for pending approval
8. Journal entry is written to log file with specified data
   (Log file can be seen via “Log” option):
   “Peer Review, User Name, UserID, Macro Name, MacroID, Macro Parameters, Date”
9. User sees alert box saying that request has been processed. An email is sent to all *admins* of macro request.

Alternate Flow:

1a. User selects another option after selecting current option
3a. User selects “Urgent Exec”
3b. Prompt asks “Are you sure?”
3c. User selects “yes”
3d. Macro parameters are encapsulated into an update object
3e. Journal entry is written to log file with specified data
(Log file can be seen via “Log” option):
“Override, User Name, UserID, Macro Name, MacroID, Macro
Parameters, Date”
3f. Add object is written to the *driver tables*
5a. User selects “no” on the “Are you sure?” prompt. The user is then able to
continue editing parameters

Error Condition:

4a. User selects execute without inputting all relevant parameters. System displays error message with empty relevant parameters emphasized in a message with an example of the field input for empty parameters. HTML5 detects incorrect error based on the input’s text pattern, the process ceases, and the user is sent back to step 2.
4b. User selects execute but has input the wrong type in one or many of the input parameters. System displays error message with incorrect input parameters emphasized in a message with an example of the field input for incorrect

Parameters. HTML5 detects incorrect error based on the input's text pattern, the process ceases, and the user is sent back to step 2.

2.5 View Log of Tasks

ID: 2.5

Title: View Log of Tasks

Actor: *Admin*, *Developer*

Trigger: User selects log option

Precondition: Must be logged in

User's Goal: To view the driver tasks that user has executed in the system

Steps:

1. User selects what they want to view [See Figure 11]
2. User is displayed a list or graph of what they have selected

Alternate Flow:

1a. User selects another option after selecting current option

Error Condition:

1b. User is logged in, however, if the connection to the database is down, the application sends error message that there is no connection to the database.

2.6 Peer Review of Pending Tasks

ID: 2.6

Title: Peer Review

Actor: *Admin, Developer*

Trigger: *Admin or Developer* selects peer review [See Figure 12]

Precondition: *Admin or Developer* must be logged in

User's Goal: To accept or decline a pending job request

Steps:

1. Pending tasks are displayed
2. *Admin* accepts pending process to run
3. Database is affected by having process shown as run
4. *Admin* is shown a message of action
5. *Admin* is shown a refreshed list of pending process

6. Journal entry is written to log file with specified data
   (Log file can be seen via “Log” option):
   “Review: Approval, Admin Name, AdminID, User Name, UserID, Macro Name, MacroID, Macro Parameters, Date”
7. *Admin and Initiator* receives a notification that the macro request has been approved via email.

Alternate Flow:

3a. *Admin* clicks decline and process shows as disabled in database

3b. Journal entry is written to log file with specified data
(Log file can be seen via “Log” option):
“Review: Denial, Admin Name, AdminID, User Name, UserID, Macro Name, MacroID, Macro Parameters, Date”

3c. *Admin and Initiator* receives a notification that the macro request has been declined via email.

Error Condition:

2a. User is logged in, however, if the connection to the database is down, the application sends an error message that there is no connection to the database.

2.7 Logout via Active Directory

ID: 2.7

Title: Logout via Active Directory

Actor: *Admin*, *Developer*

Trigger: User selects logout option [See Figure 2, Top Right]

Precondition: Must be logged in

User's Goal: To log user out of the application

Steps:

1. Remove user credential
2. User is redirected to the homepage

Error Condition:

1a. If the user selects logout when they are not logged in, the system will display the error message, “not currently logged in”.

3 Non-Functional Requirements

3.1 Software Quality Attributes

The design of this application would ensure simplicity and user-friendliness. All features are organized and composed in a drop-down menu with details of what the command does. IT department employees would be able to add additional modules at later times in order to improve the performance of the application, seamlessly. The applications interface and organizational structure could easily accommodate additional features.

3.2 Performance Requirements

This application's features run based on the given database, so the performance would rely on how big our database would be. But we only need to find and check the existence of the driver or macros before executing commands, the process will be simple and fast and should not use excessive computing resources.

3.3 Safety Requirements

The application will be compatible on all web browser and not affect any other software or hardware components on the user's pc.

3.4 Security Requirements

The application will require us to use active directory's user credentials in order to protect the application from outside threats. *Admins* and *developers* will have access to all functionals in the application. Only *admins* may be able to approve and run a macro.

4 Future Requirements

4.1 Runtime

The application will later offer a historical runtime module, which will keep track of average run times for all job levels (Batch, Job, Step) as well as overall historical trendings.

4.2 SLA

Tracking Service Level Agreements (SLAs) via the view module will be necessary after we can add and run them for instance seeing the progress of a pending SLA job.

4.3 Undo

This feature would be able to have such functionality in order to undo changes in the job tables in order to correct possible mistakes made in the driver schedule.

4.4 View in progress Macro

The ability to view in progress macros would help *developers* see the performance of their driver schedule table in order to help them resolve any runtime issues that they could encounter. Such a module would have to dynamically call to the tables in real time while the job is running in order to estimate the current job completion timings.

4.5 Stop in Progress Macro

Via the view feature the *admin* would have to ability to stop in progress macros. This functionality would be available to *dev* users on request via peer review.

4.6 Macro Definition

Using the add feature the *admin* would be able to define a new macro definition along with their active run dependencies. Error handling would be included in writing new macro definitions in order to prevent future data corruption (e.g. DELETE ROW *).

5 User Interface

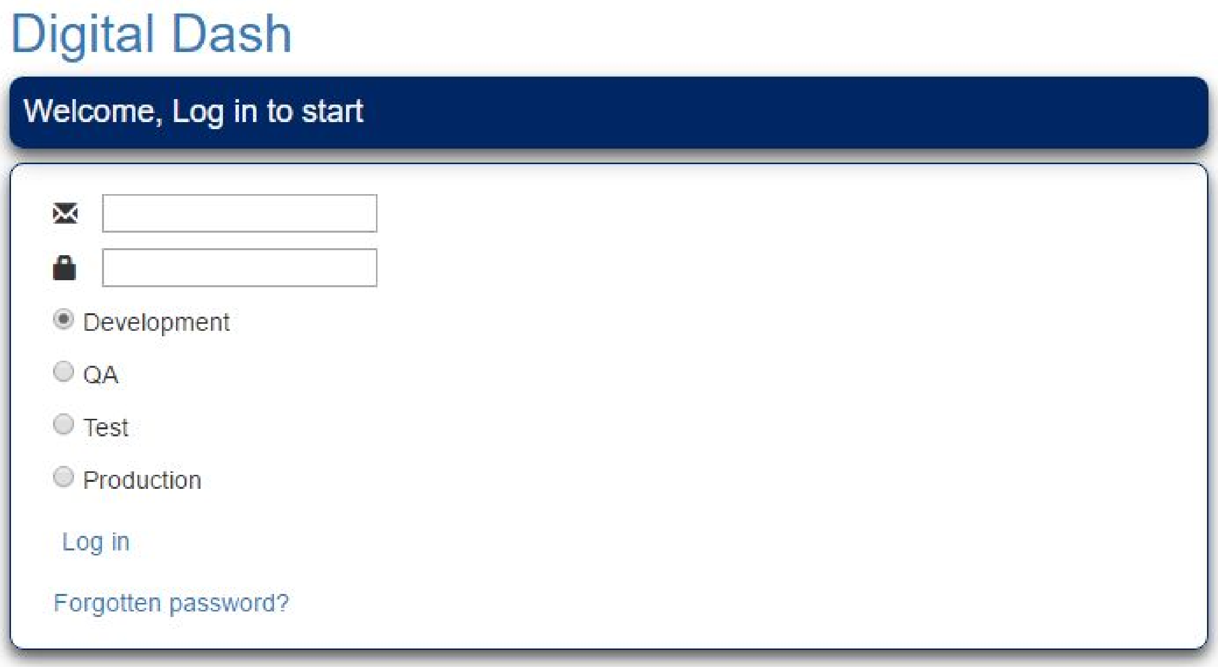
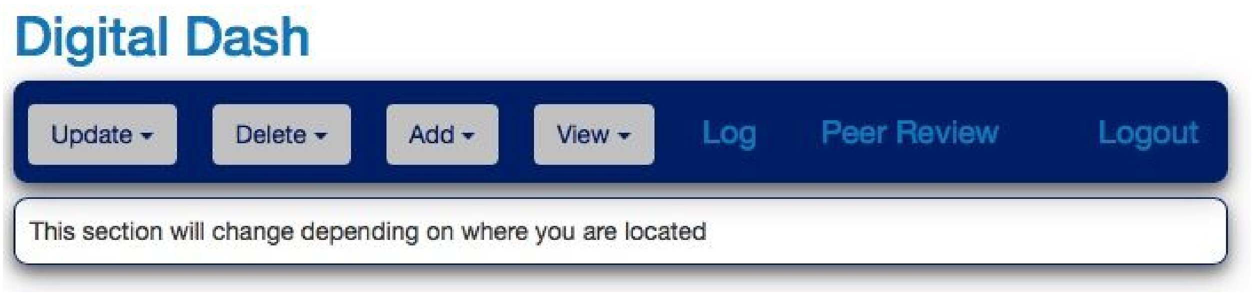


FIGURE 1 - Login via Active Directory
The display in detail of Login button in dynamic content section.



This section will change depending on where you are located

FIGURE 2 - Main Menu
The home screen, with Update, Delete, Add and View four drop down menus highlighted, consists of Log, Peer Review and Logout button and dynamically display section

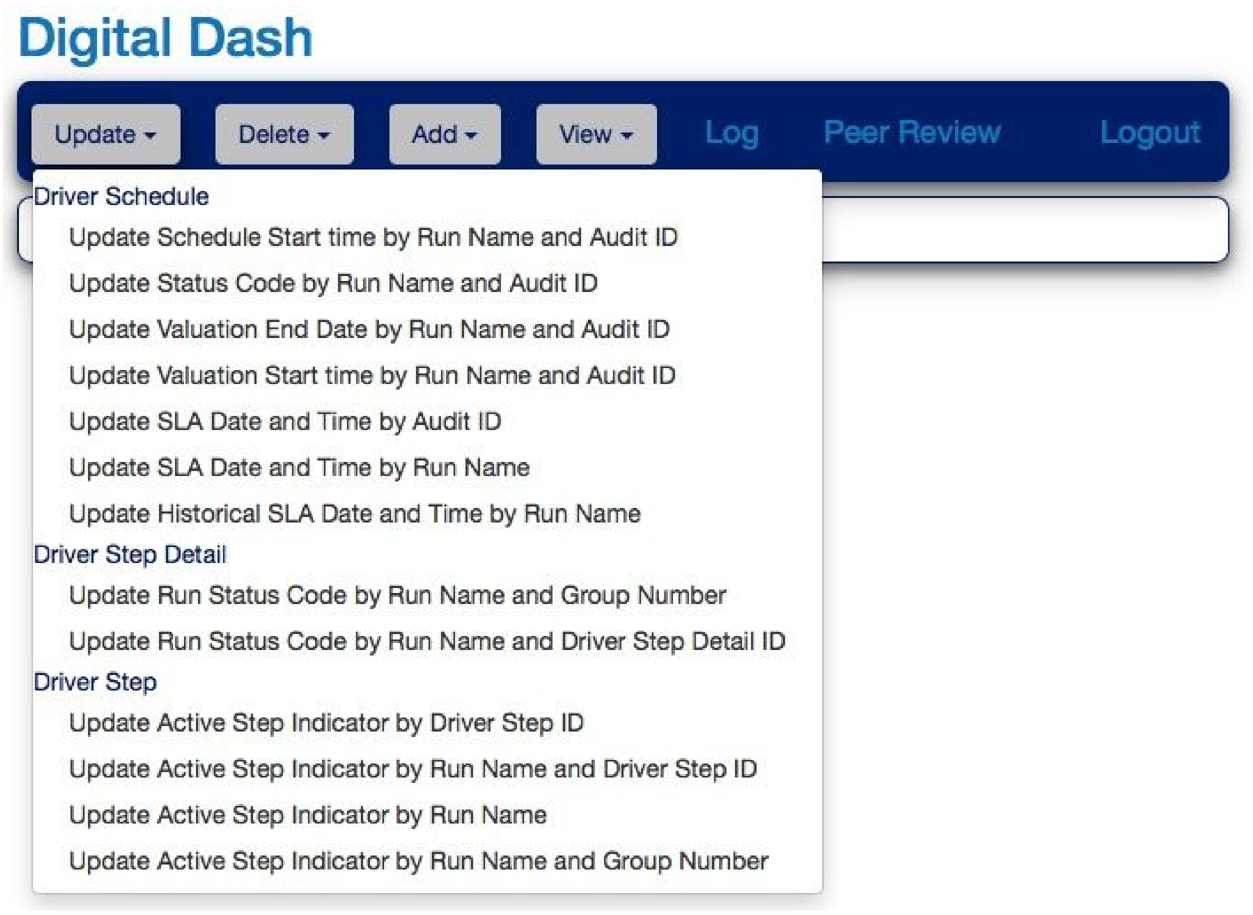

Digital Dash
Update
Delete
Add
View
Log
Peer Review
Logout
Driver Schedule
Update Schedule Start time by Run Name and Audit ID
Update Status Code by Run Name and Audit ID
Update Valuation End Date by Run Name and Audit ID
Update Valuation Start time by Run Name and Audit ID
Update SLA Date and Time by Audit ID
Update SLA Date and Time by Run Name
Update Historical SLA Date and Time by Run Name
Driver Step Detail
Update Run Status Code by Run Name and Group Number
Update Run Status Code by Run Name and Driver Step Detail ID
Driver Step
Update Active Step Indicator by Driver Step ID
Update Active Step Indicator by Run Name and Driver Step ID
Update Active Step Indicator by Run Name
Update Active Step Indicator by Run Name and Group Number

FIGURE 3 - Update Driver Table
The detail macros in Update drop down menu.

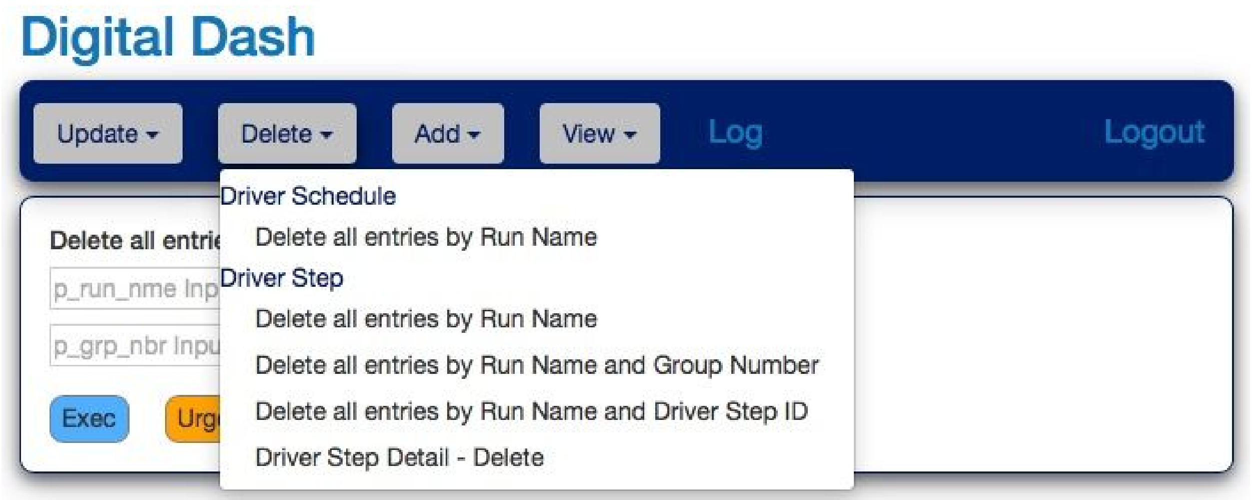
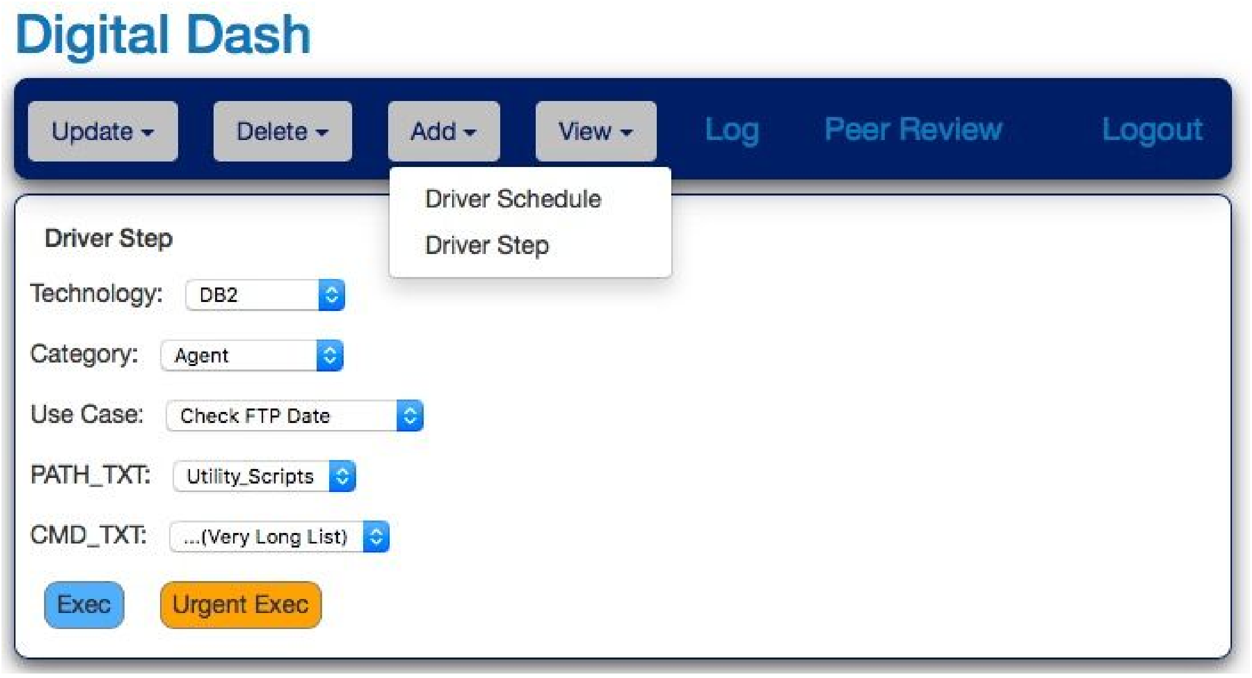

Digital Dash
Update
Delete
Add
View
Log
Logout
Delete all entrie
p_run_nme Inp
p_grp_nbr Inpu
Exec
Urg
Driver Schedule
Delete all entries by Run Name
Driver Step
Delete all entries by Run Name
Delete all entries by Run Name and Group Number
Delete all entries by Run Name and Driver Step ID
Driver Step Detail - Delete

FIGURE 4 - Delete Driver Entry
The detail macros in Delete drop down menu.



Technology: DB2

Category: Agent

Use Case: Check FTP Date

PATH_TXT: Utility_Scripts

CMD_TXT: ...(Very Long List)

FIGURE 5 - Add Driver to Table
The detail macros in Add drop down menu.

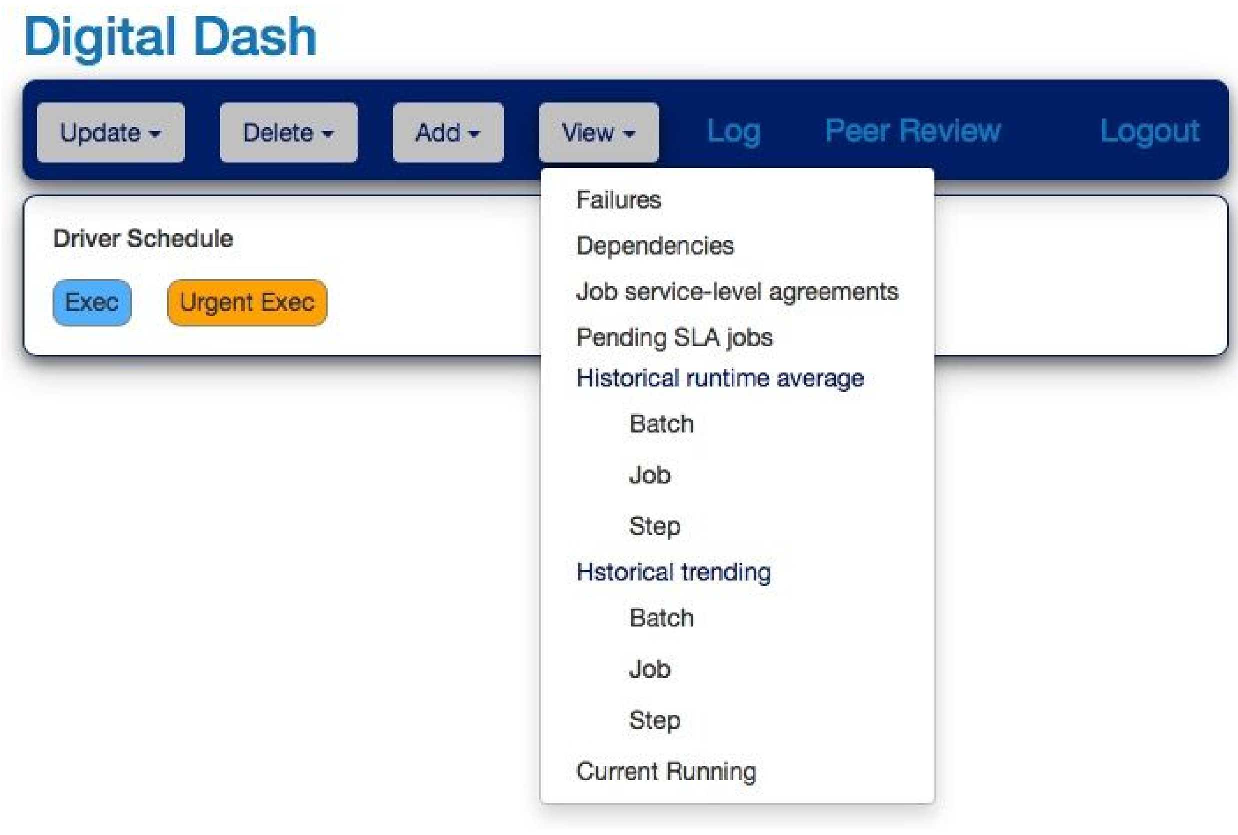


FIGURE 6 - View Feature
The detail view options in View drop down menu.

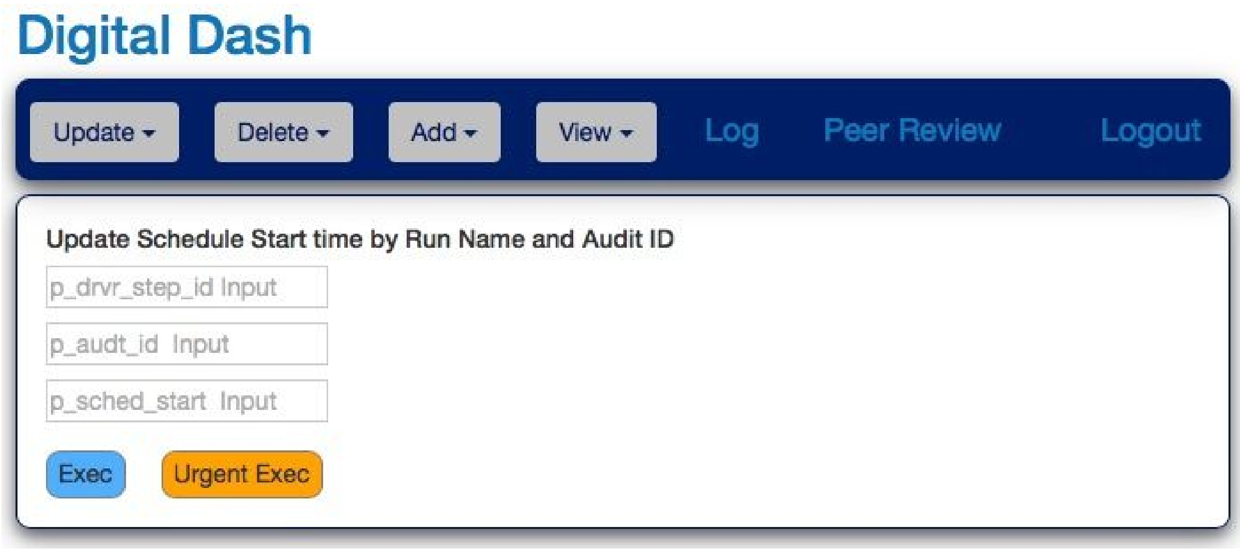


FIGURE 7 - Update Option Example
The display in detail of Update Schedule Start time by Run Name and Audit ID in dynamic content section.

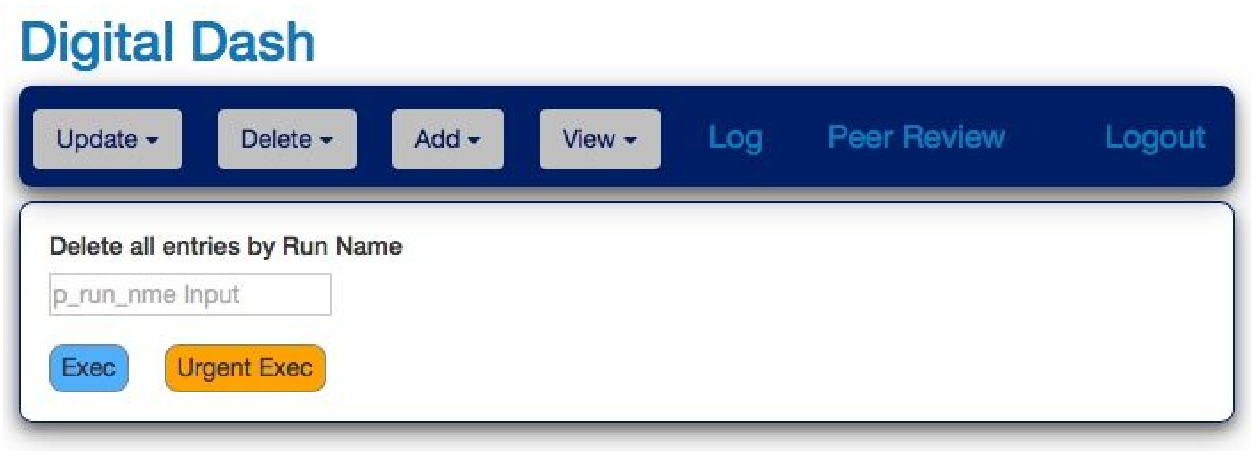


FIGURE 8 - Delete Option Example
The display in detail of Delete all entries by Run Name in dynamic content section.

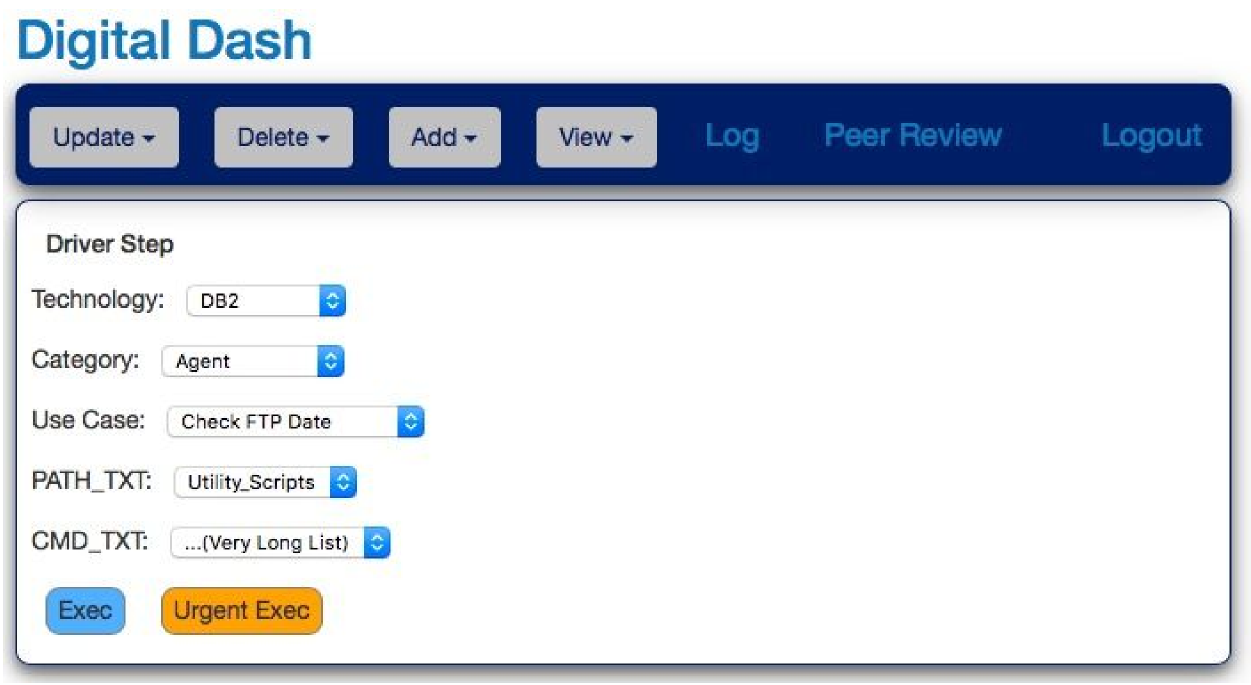


FIGURE 9 - Add Option Example
The display in detail of Add Driver Step in dynamic content section.

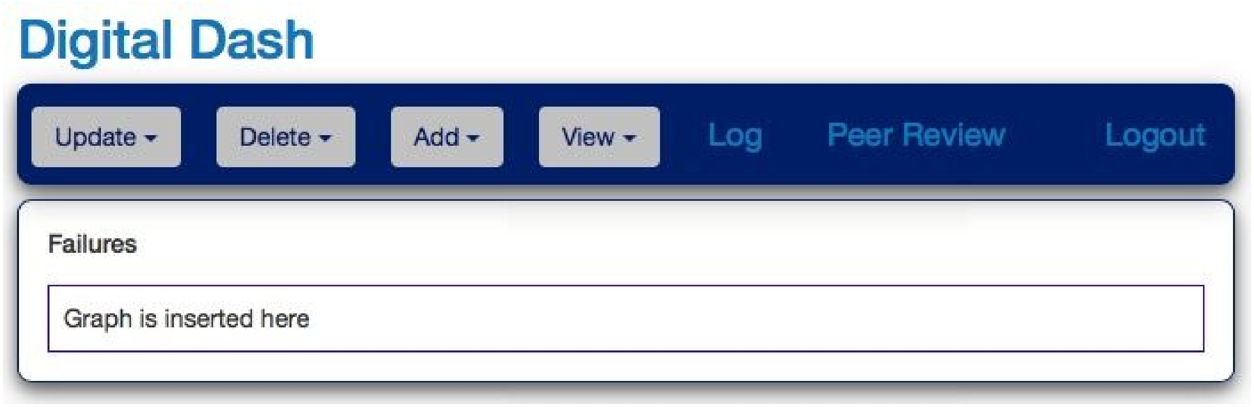
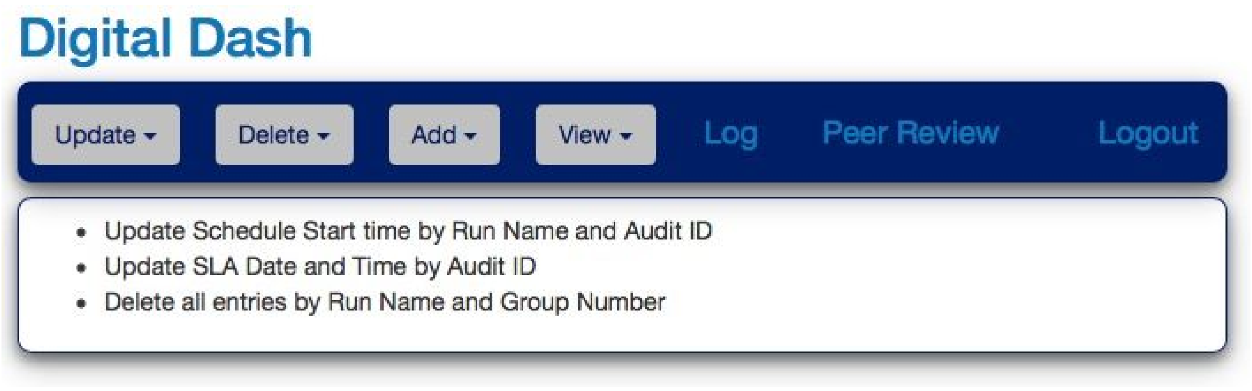


FIGURE 10 - View Option Example
The display in detail of viewing failures in dynamic content section.



- Update Schedule Start time by Run Name and Audit ID
- Update SLA Date and Time by Audit ID
- Delete all entries by Run Name and Group Number

FIGURE 11 - View Log of Tasks
The display in detail of Log content section.

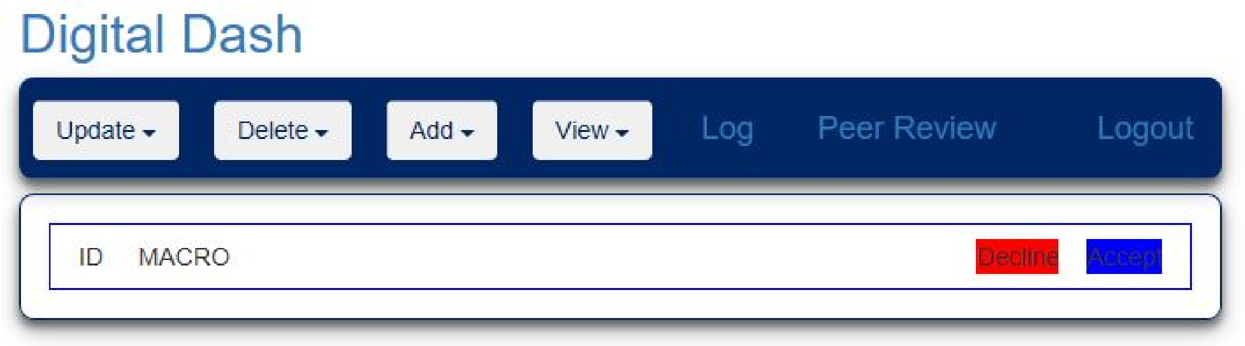


FIGURE 12 - Peer Review Pending Tasks
The display in detail of the Peer Review layout.

6 Glossary of Terms

1. Application - the software system
2. Macro - Contains pre-written SQL statements, can be called with parameters
3. SLA - Service Level Agreement
4. User - Driver Writer that is either an *Admin* or *Developer*

7 References

7.1 Documents Provided by Liberty Mutual

1. Create Run Name Document
2. Dictionary Categories Spreadsheet
3. Macros Document
4. Sample query

5. Universal Driver Control Document

7.2 Overview of project

1. UMass Amherst Capstone Project Charter

7.3 Sample data provided from tables (Liberty Mutual DB [7.5]):

1. C_DRIVER_STEP
2. C_DRIVER_STEP_DETAIL
3. C_DRIVER_STEP_DETAIL_H
4. C_DRIVER_SCHEDULE
5. C_APP_RUN_DEPENDENCY