Change request log 2

1. Concept Location

Step	Description	Rationale
1	Ran the system.	-
2	Logged in and navigated to the home screen.	Identified the relevant UI elements and screens that needed modification.
3i	Interacted with the mute button in the header and attempted to mute/unmute.	Verified its existing functionality.
4	Opened the Network tab in the browser's developer tools and examined API calls.	Traced the backend request responsible for updating the mute state.
5	Found API call dwr/call/plaincall/MiscDwr.toggleUserMuted.dwr initiated from common.js.	Led to setUserMuted() in common.js, which managed the mute state.
6	Identified the soundPlayer() object in header.js, called from setUserMuted().	-
7	Noted that modifying the default mute value in header.js had no effect.	Led us to investigate how user settings were stored.
8	Checked User.java, where user preferences were initialized.	Confirmed this as the location to implement the default mute behavior.

Time spent (in minutes): 30

Classes and methods inspected:

- /mango/mangoSource/war/resources/header.js
  - Function SoundPlayer()
- /mango/mangoSource/war/resources/common.js
  - function setUserMuted(muted)
- /mango/mangoSource/src/com/serotonin/mango/web/dwr/MiscDwr.java
  - public boolean toggleUserMuted()
- /mango/mangoSource/src/com/serotonin/mango/vo/User.java
  - public class User implements SetPointSource, HttpSessionBindingListener, JsonSerializable{}

2. Impact Analysis

Step	Description	Rationale
1	We made a list of methods related to the mute functionality, starting with toggleUserMuted() in MiscDwr.java.	To track the classes and functions impacted by the change.
2	We inspected setUserMuted(muted) in common.js.	This function directly interacts with the UI and controls the mute functionality.
3	We analyzed the SoundPlayer() function in header.js.	This function is responsible for playing notification sounds, so modifying its default behavior is crucial.
4	We checked how the User class in User.java stores and retrieves the mute preference.	Ensuring that user preferences persist correctly across sessions
5	We traced the API call dwr/call/plaincall/MiscDwr.toggleUserMuted.dwr.	To verify how the backend processes the mute toggle request.
6	We examined how the UI retrieves and displays the mute state.	Ensuring that the mute button state remains consistent with the backend setting.

Time spent (in minutes): 20

Classes and methods inspected:

- /mango/mangoSource/war/resources/header.js
  - Function SoundPlayer()
- /mango/mangoSource/war/resources/common.js

    - function setUserMuted(muted)
- /mango/mangoSource/src/com/serotonin/mango/web/dwr/MiscDwr.java
    - public boolean toggleUserMuted()
- /mango/mangoSource/src/com/serotonin/mango/vo/User.java
    - public class User implements SetPointSource, HttpSessionBindingListener, JsonSerializable{}

3. Prefactoring (optional)

Prefactoring was skipped as the required changes were minimal and did not necessitate significant restructuring of the existing code.

4. Actualization

Step	**Description**	**Rationale**
1	Modified private transient boolean muted = true; in User.java.	Ensures that the notification sound is muted by default when the system starts.
2	Verified setUserMuted(muted) in common.js to confirm that toggling the mute button updates the state correctly.	Ensures the frontend correctly reflects the new default behavior.
3	Tested SoundPlayer() in header.js to ensure it respects the default muted state.	Confirms that sound does not play unless explicitly enabled.
4	Conducted unit tests and functional tests to validate the behavior across different scenarios.	Ensures correctness and prevents regressions.
5	Committed changes with a clear message documenting the modification.	Maintains version history and allows rollback if necessary.

Time spent (in minutes): 30

Classes and Methods Changed:

- User.java
    - private transient boolean muted = true;

5. Postfactoring (optional)

Since our changes were minimal and focused on fixing a bug rather than restructuring the code, a major postfactoring step is not strictly necessary.

6. Validation

Step	**Description**	**Rationale**
1	Test case defined: Tested the regular data flow by toggling the mute button on the navigation bar. Inputs: Clicking the mute/unmute button. Expected Output: The notification sound should toggle between on and off states.	Ensures that the mute button functionality works as expected. The test passed.. The test passed.
2	Test case defined: Tested the functionality of the mute button after toggling the sound state. Inputs: Muting and unmuting multiple times. Expected Output: The sound should toggle correctly each time.	Ensures the button correctly changes the sound state. The test passed.
3	Test case defined: Checked API response structure in the network tab. Inputs: Clicking the mute button and checking the network call. Expected Output: The API response should reflect the updated mute state.	Confirms that the backend properly updates the mute state. The test passed.

Time spent (in minutes): 50

7. Summary of the change request

Phase	Time (minutes)	No. of classes inspected	No. of classes changed	No. of methods inspected	No. of methods changes
Concept location	30	4	0	4	0
Impact Analysis	20	4	0	4	0
Prefactoring	-	-	-	-	-
Actualization	30	1	1	1	1
Postfactoring	-	-	-	-	-
Verification	50	0	0	0	0
Total	130	4	1	4	1

8. Conclusion

The change request to mute the notification sound by default in the Mango system was successfully implemented. After identifying the relevant UI elements and backend functions, we modified the default mute state in User.java. We then validated the changes through unit and functional tests to ensure proper functionality.

The mute button now defaults to muted, and users can toggle it as needed, without affecting other system behaviors. The implementation was efficient, with minimal impact on the existing system, and the change was delivered as expected.