Change request log

1. Concept Location

Step #	Description	Rationale
1	We ran the system	
2	We interacted with the system: after logging in the system with admin credentials.	To get familiar with some of the features of the system, and identify the screens or graphical elements we had to change.
3	We went to the password change page and looked into the URL.	To understand which is the jsp page, showing the user details.
4	We inspected users.jsp file, where saveUser() method lead us to UserDwr class.	The UserDwr class has the main implementation of saveUserAdmin().
5	We inspected UserDwr, which lead us to UserDao class, and inspected saveUser() method of that class.	Because we need to reach to the method which is interacting with the database.
6	The saveUser() method lead us to the updateUser() method and where we put debugging point and printed all the values coming from the frontend part.	To understand and find out if any of the values coming to the saveUser() method null or not. In result some of the values were printing null values.
7	We marked UserDao class as concept location.	We confirmed this class had to be modified.

Time spent (in minutes): 27

Classes and methods inspected:

- *\mangoSource\war\WEB-INF\jsp\users.jsp*
  - *saveUser()*
- *\mangoSource\src\com\serotonin\mango\web\dwr\UsersDwr.java*
  - *DwrResponseI18n saveUserAdmin()*
- *\mangoSource\src\com\serotonin\mango\db\dao\UserDao.java*
  - *void saveUser()*
  - *void updateUser()*

2. Impact Analysis

Step #	Description	Rationale
1	We used IDE's search tool in order to identify the classes or methods using the UserDao's updateUser() method,. We used regular expressions to search for classes.	To track the classes that could be impacted by the change.
2	We marked UserDao class changed and the classes which came as a result are marked as To be inspected, hence we marked UserDwr class as to be inspected.	We realized that UserDao class's updateUser() method is getting some null values from the front-end. So, null values need to be handled properly in updateUser() method, which might impact UserDwr class which is making use of UserDao class's method.
3	We inspected UserDwr class as a result and marked as inspected and unchanged.	Because, the change in UserDao() class's updateUser() method is not returning anything, and not having any impact on UserDwr's saveUserAdmin() method.

Time spent (in minutes): 18

- *\mangoSource\src\com\serotonin\mango\web\dwr\UsersDwr.java*

    - *DwrResponseI18n saveUserAdmin()*
- *\mangoSource\src\com\serotonin\mango\db\dao\UserDao.java*
    - *void saveUser()*
    - *void updateUser()*

3. Actualization

Step #	Description	Rationale
1	We added a debug statement in UsersDwr.java to print the saved password.	To aid in debugging by confirming the value of the saved password. This helped verify that the password was being saved correctly after encryption.
2	We modified UserDao.java to handle null values for phone and homeUrl.	These fields were receiving null values from the frontend, causing errors during database updates. Converting them to empty strings ensured data integrity and prevented potential null pointer exceptions.
3	We tested the changes to ensure they worked as expected.	To ensure the changes worked as expected

Time spent (in minutes): 35

Inspected: -

- \mangoSource\src\com\serotonin\mango\web\dwr\UsersDwr.java
    - DwrResponseI18n saveUserAdmin()
- \mangoSource\src\com\serotonin\mango\db\dao\UserDao.java
    - void saveUser()
    - void updateUser()

Changed: -

- \mangoSource\src\com\serotonin\mango\web\dwr\UsersDwr.java
    - DwrResponseI18n saveUserAdmin()
- \mangoSource\src\com\serotonin\mango\db\dao\UserDao.java
    - void updateUser()
-

4. Validation

Step #	Description	Rationale
1	**Test case 1**: Verify the system handles valid data correctly. **Inputs**: Valid user data (e.g., valid username, password, phone, and homeUrl). **Expected output**: User data is saved successfully without errors.	This is the regular expected behavior. The test passed.
2	**Test case 2**: Verify the system handles null values for phone and homeUrl.	This is an exceptional behavior (when values are null).

	Inputs: User data with null values for phone and homeUrl. **Expected output**: User data is saved successfully, and null values are handled gracefully.	The test passed.
3	**Test case 3:** Verify the system handles long strings for phone and homeUrl. **Inputs:** User data with long strings (e.g., >100 characters) for phone and homeUrl. **Expected output:** User data is saved successfully, and long strings are handled correctly.	This is an exceptional behavior (when values are larger than 100 characters). The test passed.

Time spent (in minutes): 60

5. 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	27	3	0	2	0
Impact Analysis	18	2	0	3	0
Actualization	35	1	2	2	2
Verification	60	1	0	2	0
Total	140	7	2	9	2

6. Conclusions

For this change request, concept location was relatively straightforward as the issue was clearly related to the phone and homeUrl fields in UserDao.java and the password-saving functionality in UserDwr.java. The impact analysis was smooth since the changes were localized to these two classes, with minimal dependencies. The actual implementation involved handling null values and adding debug statements, which were validated through manual and functional testing. The testing confirmed that the changes worked as expected, and the system handled both regular and exceptional cases correctly. Overall, the process was efficient, with the main focus on ensuring no regressions were introduced.