JustHealth Test Cases

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Authored By:** | Stephen Tate |
| **Date Authored:** | 2nd October 2014 |
| **Version #:** | 1.0 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Date**  (DD-MMM-YYYY) | **Signed** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.1. Client Table: Inserting legitimate data into the client table 5](#_Toc400690474)

[1.2. Client Table: Inserting data that is too long into the client table 6](#_Toc400690475)

[1.2. Client Table: Inserting data that is too long into the client table (Cont’d) 7](#_Toc400690476)

[1.3. Client Table: Executing an insert command with data missing 8](#_Toc400690477)

[1.3. Client Table: Executing an insert command with data missing (Cont’d) 9](#_Toc400690478)

[1.4. Client Table: Deleting records from the table 10](#_Toc400690479)

[1.5. Client Table: Update the data for a given record 11](#_Toc400690480)

[1.5. Client Table: Update the data for a given record (Cont’d) 12](#_Toc400690481)

[1.5. Client Table: Update the data for a given record (Cont’d) 13](#_Toc400690482)

[1.5. Client Table: Update the data for a given record (Cont’d) 14](#_Toc400690483)

[1.5. Client Table: Update the data for a given record (Cont’d) 15](#_Toc400690484)

[1.6. Password Table: Inserting legitimate data into the password table 16](#_Toc400690485)

[1.7. Password Table: Inserting illegitimate data into the password table 17](#_Toc400690486)

[1.8. Password Table: Deleting records from the password table 18](#_Toc400690487)

[1.9. Password Table: Testing the Constraint on Foreign key 19](#_Toc400690488)

[1.10. Web Application: User Registration 20](#_Toc400690489)

[1.10. Web Application: User Registration (Cont’d) 21](#_Toc400690490)

[1.11. Web Application: Log In Functionality 22](#_Toc400690491)

[1.12. Web Application: Account Locked 23](#_Toc400690492)

[1.12. Web Application: Account Locked (Cont’d) 24](#_Toc400690493)

[1.13. Web Application: Forgot Password 25](#_Toc400690494)

[1.13. Web Application: Forgot Password (Cont’d) 26](#_Toc400690495)

[1.14. ORM Testing: Inserting legitimate data into the client table 27](#_Toc400690496)

[1.15. ORM Testing: Inserting data that is too long into the client table 28](#_Toc400690497)

[1.15. ORM Testing: Inserting data that is too long into the client table (Cont’d) 29](#_Toc400690498)

[1.16. ORM Testing: Executing an insert command on the client table with data missing 30](#_Toc400690499)

[1.16. ORM Testing: Executing an insert command on the client table with data missing (Cont’d) 31](#_Toc400690500)

[1.17. ORM Testing: Deleting records from the client table 32](#_Toc400690501)

[1.18. ORM Testing: Update the data for a given record in the client table 33](#_Toc400690502)

[1.18. ORM Testing: Update the data for a given record in the client table (Cont’d) 34](#_Toc400690503)

[1.19. ORM Testing: Inserting legitimate data into the password table 35](#_Toc400690504)

[1.20. ORM Testing: Inserting illegitimate data into the password table 36](#_Toc400690505)

[1.21. ORM Testing: Deleting records from the password table 37](#_Toc400690506)

[1.22. ORM Testing: Testing the Constraint on Foreign key 38](#_Toc400690507)

[Appendix 39](#_Toc400690508)

[1.1. Client Table: Inserting legitimate data into the client table 39](#_Toc400690509)

[1.1.1. 39](#_Toc400690510)

[1.2. Client Table: Inserting data that is too long into the client table 40](#_Toc400690511)

[1.2.1. 40](#_Toc400690512)

[1.2.2. 40](#_Toc400690513)

[1.2.3. 40](#_Toc400690514)

[1.2.4. 40](#_Toc400690515)

[1.2.5. 40](#_Toc400690516)

[1.2.6. 40](#_Toc400690517)

[1.2.7. 40](#_Toc400690518)

[1.3. Client Table: Executing an insert command with data missing 41](#_Toc400690519)

[1.3.1. 41](#_Toc400690520)

[1.3.2. 41](#_Toc400690521)

[1.3.3. 41](#_Toc400690522)

[1.3.4. 41](#_Toc400690523)

[1.3.5. 41](#_Toc400690524)

[1.3.6. 41](#_Toc400690525)

[1.3.7. 41](#_Toc400690526)

[1.4. Client Table: Deleting records from the table 42](#_Toc400690527)

[1.4.1. 42](#_Toc400690528)

[1.4.2. 42](#_Toc400690529)

[1.5. Client Table: Update the data for a given record 43](#_Toc400690530)

[1.5.1. 43](#_Toc400690531)

[1.5.2. 43](#_Toc400690532)

[1.5.3. 43](#_Toc400690533)

[1.5.4. 43](#_Toc400690534)

[1.5.5. 44](#_Toc400690535)

[1.5.6. 44](#_Toc400690536)

[1.5.7. 44](#_Toc400690537)

[1.5.8. 44](#_Toc400690538)

[1.5.9. 45](#_Toc400690539)

[1.6. Password Table: Inserting legitimate data into the password table 46](#_Toc400690540)

[1.6.1. 46](#_Toc400690541)

[1.7. Password Table: Inserting illegitimate data into the password table 47](#_Toc400690542)

[1.7.1. 47](#_Toc400690543)

[1.7.2. 47](#_Toc400690544)

[1.7.3. 47](#_Toc400690545)

[1.7.4. 47](#_Toc400690546)

[1.8. Password Table: Deleting records from the password table 48](#_Toc400690547)

[1.8.1. 48](#_Toc400690548)

[1.9. Password Table: Testing the Constraint on Foreign key 49](#_Toc400690549)

[1.9.1. 49](#_Toc400690550)

# Client Table: Inserting legitimate data into the client table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.1.1 | 1. Run the SQL found in Appendix 1.1.1. 2. Run the following SQL on the database:   SELECT \* FROM client | There is a record in the database with username = test, email = [test@test.com](mailto:test@test.com) and verified = FALSE. |  |

# Client Table: Inserting data that is too long into the client table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.2.1 | 1. Run the SQL found in Appendix 1.2.1. | The insert statement should fail to execute. |  |
| 1.2.2 | 1. Run the SQL found in Appendix 1.2.2. | The insert statement should fail to execute. |  |
| 1.2.3 | 1. Run the SQL found in Appendix 1.2.3. | The insert statement should fail to execute. |  |
| 1.2.4 | 1. Run the SQL found in Appendix 1.2.4. | The insert statement should fail to execute. |  |

# Client Table: Inserting data that is too long into the client table (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.2.5 | 1. Run the SQL found in Appendix 1.2.5. | The insert statement should fail to execute. |  |
| 1.2.6 | 1. Run the SQL found in Appendix 1.2.6. | The insert statement should fail to execute. |  |
| 1.2.7 | 1. Run the SQL found in Appendix 1.2.7. | The insert statement should fail to execute. |  |

# Client Table: Executing an insert command with data missing

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.3.1 | 1. Run the SQL found in Appendix 1.3.1. | The insert statement should fail to execute. |  |
| 1.3.2 | 1. Run the SQL found in Appendix 1.3.2. | The insert statement should fail to execute. |  |
| 1.3.3 | 1. Run the SQL found in Appendix 1.3.3. | The insert statement should fail to execute. |  |
| 1.3.4 | 1. Run the SQL found in Appendix 1.3.4. | The insert statement should fail to execute. |  |

# Client Table: Executing an insert command with data missing (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.3.5 | 1. Run the SQL found in Appendix 1.3.5. | The insert statement should fail to execute. |  |
| 1.3.6 | 1. Run the SQL found in Appendix 1.3.6. | The insert statement should fail to execute. |  |
| 1.3.7 | 1. Run the SQL found in Appendix 1.3.7. | The insert statement should fail to execute. |  |

# Client Table: Deleting records from the table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.4.1 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.4.1. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.4.1. 2. Run SELECT \* FROM client, password; | The record that was previously added should be deleted. There should be no records in the table. |  |
| 1.4.2 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.4.2. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.4.2. 2. Run SELECT \* FROM client, password; | The record that was previously added should be deleted. There should be no records in the table. |  |

# Client Table: Update the data for a given record

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.5.1 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.1. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.1. 2. Run SELECT \* FROM client; | The username field should have updated and now display ‘testingUpdate’. |  |
| 1.5.2 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.2. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.2. 2. Run SELECT \* FROM client; | The firstName field should have updated and now display ‘testingFirstName’ |  |

# Client Table: Update the data for a given record (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.5.3 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.3. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.3. 2. Run SELECT \* FROM client; | The surname field should have updated and now display ‘testingSurname’. |  |
| 1.5.4 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.4. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.4. 2. Run SELECT \* FROM client; | The dob field should have updated and now display ‘03/03/1993’. |  |

# Client Table: Update the data for a given record (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.5.5 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.3. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.3. 2. Run SELECT \* FROM client; | The isMale field should have updated and now display FALSE. |  |
| 1.5.6 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.4. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.4. 2. Run SELECT \* FROM client; | The isCarer field should have updated and now display FALSE. |  |

# Client Table: Update the data for a given record (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.5.7 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.3. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.5.3. 2. Run SELECT \* FROM client; | The email field should have updated and now display ‘testingUpdate@testingUpdate.com’. |  |
| 1.5.8 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.3. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement. Verified should be set to FALSE.*   1. Run the second SQL statement found in Appendix 1.5.3. 2. Run SELECT \* FROM client; | The verified field should have updated to TRUE. |  |

# Client Table: Update the data for a given record (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.5.9 | 1. Run DELETE \* FROM client 2. Run the first SQL statement found in Appendix 1.5.4. 3. Run SELECT \* FROM client;   *you should see the record that you just added with the previous statement. Verified should be set to FALSE.*   1. Run the second SQL statement found in Appendix 1.5.4. 2. Run SELECT \* FROM client; | The accountLocked field should have updated to TRUE. |  |

# Password Table: Inserting legitimate data into the password table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.6.1 | 1. Run the two SQL statement found in Appendix 1.6.1. 2. Run SELECT \* FROM uq8LnAWi7D | The record that Appendix 1.6.1 inserts should be seen in the table. |  |

# Password Table: Inserting illegitimate data into the password table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.7.1 | 1. DELETE \* FROM uq8LnAWi7D 2. Run the SQL statement found in Appendix 1.7.1. | The SQL statement should not execute successfully because no username has been provided. |  |
| 1.7.2 | 1. Run the SQL statement found in Appendix 1.7.2. | The SQL statement should not execute successfully as the user ‘notInTable’ does not exist in the client table. |  |
| 1.7.3 | 1. Run the SQL statement found in Appendix 1.7.3. | The SQL statement should not execute successfully because the password field is too long. |  |
| 1.7.4 | 1. Run the SQL statement found in Appendix 1.7.4. | The SQL statement should not execute successfully because the isCurrent value passed is not a legitimate Boolean value. |  |
| 1.7.5 | 1. Run the SQL statement found in Appendix 1.7.5. | The SQL statement should not execute successfully because the expiryDate is not valid. |  |

# Password Table: Deleting records from the password table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.8.1 | 1. Run the first SQL statement found in Appendix 1.8.1. 2. Run SELECT \* FROM uq8LnAWi7D;   *You should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.8.1. 2. Run SELECT \* FROM uq8LnAWi7D; | The record that was previously added should be deleted. There should be no records in the table. |  |

# Password Table: Testing the Constraint on Foreign key

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Database |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.9.1 | 1. Run DELETE \* FROM uq8LnAWi7D; 2. Run DELETE \* FROM client; 3. Run the first SQL statements found in Appendix 1.9.1. 4. Run SELECT \* FROM client;   *You should see the record that you just added with the previous statement.*   1. Run SELECT \* FROM uq8LnAWi7D;   *You should see the record that you just added with the previous statement.*   1. Run the second SQL statement found in Appendix 1.9.1. 2. Run SELECT \* FROM client; 3. Run SELECT \* FROM uq8LnAWi7D; | The records in both the client and the uq8LnAWi7D should have been removed. |  |

# Web Application: User Registration

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Navigate to the registration screen. 2. Fill in the username with ‘registrationTest’ 3. Fill in the email address with *tester’s email address* 4. Fill in the First Name, Surname, Date of Birth, Gender, Carer/Patient fields with ‘test’ data. 5. Fill in the two password fields with ‘registrationPassword’ 6. Press Submit. 7. Connect to the database, run SELECT \* FROM client;   **Take a screenshot**   1. Connect to the database, run SELECT \* FROM uq8LnAWi7D;   **Take a screenshot** | The SELECT \* FROM client query should show the record created in the client table with the following attributes:   * Username = registrationTest * Email = *tester’s email address* * Verified = false   The SELECT \* FROM uq8LnAWi7D query should show the record created in the ‘password’ table with the following attributes:   * Username = registrationTest * Password = SHA value * isCurrent = True * expiryDate = 45 days from day of test |  |

# Web Application: User Registration (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Check your email to check for the verification email from JustHealth.   **Take a screenshot.** | Email to have been received asking for email verification after your recent registration. |  |
|  | 1. Click the link in the email. | The link takes you to a page to verify your email address. |  |
|  | 1. Connect to the database and run, SELECT \* FROM client   **Take a screenshot.** | The verified field has switched to TRUE. |  |

# Web Application: Log In Functionality

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Navigate to the login screen. 2. Use the username: ‘registrationTest’ and password: ‘registrationPassword’. 3. Press login. | The login is authorised. |  |
|  | 1. Navigate to the login screen. 2. Use the username: ‘registrationTest’ and password: ‘registrationPasswords’. 3. Press login. | The login is not authorised – incorrect password. |  |
|  | 1. Navigate to login screen. 2. Use the username: ‘registrationTest’ and password: ‘registrationPassword’. 3. Press login. | Login count in database updated by 1 |  |

# Web Application: Account Locked

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Navigate to the login screen. 2. Use the username: ‘registrationTest’ and password: ‘registrationPasswords’. 3. Press Login. 4. Attempt this another 4 times.   **Take a screenshot.** | A message should appear stating that the account is locked. |  |
|  | 1. Connect to the database, run SELECT \* FROM client   **Capture a screenshot** | The attribute accountLocked for the record with username ‘registrationTest’ has switched to TRUE. |  |
|  | 1. Check your email to find the email informing you of your account being locked. | Email to have been received informing you that your account is locked and containing a link to reset your password. |  |
|  | 1. Click the link in the email from JustHealth that is informing you that your account is locked. | Directs you to a site that allows you to reset your password. |  |

# Web Application: Account Locked (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Type in both of the password fields: ‘resetPassword’. 2. Click ‘Reset Password’. | You are informed that your account has been unlocked within the web browser page. |  |
|  | 1. Connect to the database and run, SELECT \* FROM uq8LnAWi7D;   **Capture a screenshot** | There are two records for username: ‘registrationTest’. The record with the lower number recordId, has its attribute isCurrent set to FALSE and the other has its isCurrent attribute set to TRUE. |  |
|  | 1. Navigate to the login screen. 2. Use the username: ‘registrationTest’ and password: ‘resetPassword’. 3. Press Login. | The login is authorised. |  |

# Web Application: Forgot Password

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Navigate to the login screen. 2. Press the ‘forgot password’ button/link. | Directed to a screen that asks the user to enter their username, email address and dob that is associated with their JustHealth account. |  |
|  | 1. Enter username: ‘registrationTest’ , email, dob, new pass, confirm new pass 2. Press the ‘Confirm’ button. | Directed to a screen that informs the user that an email has been sent to their email to click to reactivate their account. |  |
|  | 1. Check your email to find the email containing a link to reset your password. | Email to have been received informing you have reset your password |  |
|  | 1. Click the link in the email from JustHealth verifying your reset password 2. Log in with new passwords | Allows user to log in with new password |  |
|  | 1. Type in both of the password fields: ‘forgotPassword’. 2. Click ‘Reset Password’. | You are informed that your password has been reset within the web browser page. |  |

# Web Application: Forgot Password (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1. Connect to the database and run, SELECT \* FROM uq8LnAWi7D;   **Capture a screenshot** | There are four records for username: ‘registrationTest’. All but the record with the highest number recordID, have their attribute isCurrent set to FALSE and the highest number recordId has its isCurrent attribute set to TRUE. |  |
|  | 1. Navigate to the login screen. 2. Use the username: ‘registrationTest’ and password: ‘forgotPassword. 3. Press Login. | The login is authorised. |  |

# Test for confirming current password

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | Web |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
|  | 1.Navigate to the login screen.  2.Press the ‘forgot password’ button/link.  3.Enter username: ‘registrationTest’  4.Press the ‘Confirm’ button.  5.Check your email to find the email containing a link to reset your password.  6.Click the link in the email from JustHealth that is informing you that your account is locked.  7. Change your password  8.navigate back to the login screen  9. try logging in with the old password | Unable to login with old password |  |

# ORM Testing: Inserting legitimate data into the client table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.14.1 | 1. Run function test\_1\_14\_1 in test.py. | Pass |  |

# ORM Testing: Inserting data that is too long into the client table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.15.1 | 1. Run function test\_1\_15\_1 in test.py. | Pass |  |
| 1.15.2 | 1. Run function test\_1\_15\_2 in test.py. | Pass |  |
| 1.15.3 | 1. Run function test\_1\_15\_3 in test.py. | Pass |  |
| 1.15.4 | 1. Run function test\_1\_15\_4 in test.py. | Pass |  |
| 1.15.5 | 1. Run function test\_1\_15\_5 in test.py. | Pass |  |

# ORM Testing: Inserting data that is too long into the client table (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.15.6 | 1. Run function test\_1\_15\_6 in test.py. | Pass |  |
| 1.15.7 | 1. Run function test\_1\_15\_7 in test.py. | Pass |  |

# ORM Testing: Executing an insert command on the client table with data missing

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.16.1 | 1. Run function test\_1\_16\_1 in test.py. | Pass |  |
| 1.16.2 | 1. Run function test\_1\_16\_2 in test.py. | Pass |  |
| 1.16.3 | 1. Run function test\_1\_16\_3 in test.py. | Pass |  |
| 1.16.4 | 1. Run function test\_1\_16\_4 in test.py. | Pass |  |
| 1.16.5 | 1. Run function test\_1\_16\_5 in test.py. | Pass |  |

# ORM Testing: Executing an insert command on the client table with data missing (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.16.6 | 1. Run function test\_1\_15\_6 in test.py. | Pass |  |
| 1.16.7 | 1. Run function test\_1\_15\_7 in test.py. | Pass |  |

# ORM Testing: Deleting records from the client table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.17.1 | 1. Run function test\_1\_17\_1 in test.py. | Pass |  |
| 1.17.2 | 1. Run function test\_1\_17\_2 in test.py. | Pass |  |

# ORM Testing: Update the data for a given record in the client table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.18.1 | 1. Run function test\_1\_18\_1 in test.py. | Pass |  |
| 1.18.2 | 1. Run function test\_1\_18\_2 in test.py. | Pass |  |
| 1.18.3 | 1. Run function test\_1\_18\_3 in test.py. | Pass |  |
| 1.18.4 | 1. Run function test\_1\_18\_4 in test.py. | Pass |  |
| 1.18.5 | 1. Run function test\_1\_18\_5 in test.py. | Pass |  |

# ORM Testing: Update the data for a given record in the client table (Cont’d)

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.18.6 | 1. Run function test\_1\_18\_6 in test.py. | Pass |  |
| 1.18.7 | 1. Run function test\_1\_18\_7 in test.py. | Pass |  |
| 1.18.8 | 1. Run function test\_1\_18\_8 in test.py. | Pass |  |
| 1.18.9 | 1. Run function test\_1\_18\_9 in test.py. | Pass |  |

# ORM Testing: Inserting legitimate data into the password table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.19.1 | 1. Run function test\_1\_19\_1 in test.py. | Pass |  |

# ORM Testing: Inserting illegitimate data into the password table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.20.1 | 1. Run function test\_1\_20\_1 in test.py. | Pass |  |
| 1.20.2 | 1. Run function test\_1\_20\_2 in test.py. | Pass |  |
| 1.20.3 | 1. Run function test\_1\_20\_3 in test.py. | Pass |  |
| 1.20.4 | 1. Run function test\_1\_20\_4 in test.py. | Pass |  |

# ORM Testing: Deleting records from the password table

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.21.1 | 1. Run function test\_1\_21\_1 in test.py. | Pass |  |

# ORM Testing: Testing the Constraint on Foreign key

|  |  |
| --- | --- |
| **Iteration #:** | 1 |
| **Application Type:** | ORM (Server Side) |
| **Date/Time**  **(DD-MMM-YYYY / HH:MM):** |  |
| **Tester Name:** |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ID** | **Test Steps** | **Expected Result** | **Pass/Fail (Initials)** |
| 1.22.1 | 1. Run function test\_1\_22\_1 in test.py. | Pass |  |

# Appendix

## Client Table: Inserting legitimate data into the client table

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

## Client Table: Inserting data that is too long into the client table

### 1.2.1.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘testtesttesttesttesttestte’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

### 1.2.2.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘testtesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttestt’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

### 1.2.3.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘testtesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttestt’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

### 1.2.4.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’test’, TRUE, TRUE, ’test@test.com’);

### 1.2.5.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, 01/01/2001, ‘test’, TRUE, ’test@test.com’);

### 1.2.6.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, 01/01/2001, TRUE, ‘test’, ’test@test.com’);

### 1.2.7.

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, 01/01/2001, TRUE, TRUE, ’testesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttesttestt@test.com’);

## Client Table: Executing an insert command with data missing

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’’, TRUE, TRUE, ’test@test.com’);

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, , TRUE, ’test@test.com’);

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, , ’test@test.com’);

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’’);

## Client Table: Deleting records from the table

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

DELETE FROM client

WHERE (username = ‘test’);

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

DELETE FROM client

WHERE (email = ‘test@test.com’);

## Client Table: Update the data for a given record

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET username = ‘testingUpdate’

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET firstName = ‘testingFirstName’

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET surname = ‘testingSurname’

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET dob = ‘03/03/1993’

WHERE username = ’test’;

1.5.5.

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET isMale = FALSE

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET isCarer = FALSE

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

SQL statement two

UPDATE client

SET email = ‘testingUpdate@testingUpdate.com’

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, email)

VALUES (‘test’, ’test@test.com’);

SQL statement two

UPDATE client

SET verified = TRUE

WHERE username = ’test’;

### 

SQL statement one

INSERT INTO client (username, email)

VALUES (‘test’, ’test@test.com’);

SQL statement two

UPDATE client

SET accountLocked = TRUE

WHERE username = ’test’;

## Password Table: Inserting legitimate data into the password table

### 

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (‘test’, crypt(‘password’, gen\_salt(‘md5’)), TRUE, 10/10/2014);

## Password Table: Inserting illegitimate data into the password table

### 

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (‘notInTable’, crypt(‘password’, gen\_salt(‘md5’)), TRUE, 10/10/2014);

### 

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (‘test’, crypt(‘passwordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpasswordpassword’, gen\_salt(‘md5’)), TRUE, 10/10/2014);

### 

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (test, crypt(‘password’, gen\_salt(‘md5’)), ‘test’, 10/10/2014);

### 

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (test, crypt(‘password’, gen\_salt(‘md5’)), TRUE, 19/17/1993);

## Password Table: Deleting records from the password table

### 

SQL statement one

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (‘test’, crypt(‘password’, gen\_salt(‘md5’)), TRUE, 10/10/2014);

SQL statement two

DELETE FROM uq8LnAWi7D

WHERE (username = ‘test’);

## Password Table: Testing the Constraint on Foreign key

### 

SQL statement one

INSERT INTO client (username, firstName, surname, dob, isMale, isCarer, email)

VALUES (‘test’, ‘test’, ‘test’, ’01/01/2001’, TRUE, TRUE, ’test@test.com’);

INSERT INTO uq8LnAWi7D (username, password, isCurrent, expiryDate)

VALUES (‘test’, crypt(‘password’, gen\_salt(‘md5’)), TRUE, 10/10/2014);

SQL statement two

DELETE FROM client

WHERE (username = ‘test’);