Dominic Cameron

SOFTWARE DEVELOPMENT ASSESMENT- TESTING

Contents

[Introduction 2](#_Toc193039764)

[1. Objective 2](#_Toc193039765)

[2. Testing Scope 2](#_Toc193039766)

[3. Test Strategy 2](#_Toc193039767)

[3.1 Testing Approach 2](#_Toc193039768)

[4. Test Environment 3](#_Toc193039769)

[4.1 Hardware Requirements 3](#_Toc193039770)

[4.2 Software Requirements 3](#_Toc193039771)

[5. Test Cases and Execution Steps 3](#_Toc193039772)

[5.1 Application Launch and Closure 3](#_Toc193039773)

[5.2 Toolbar Functions 3](#_Toc193039774)

[5.3 Table Right-Click Functions 3](#_Toc193039775)

[5.4 Add Records (Database Validation) 4](#_Toc193039776)

[5.5 Search Records 4](#_Toc193039777)

[5.6 Edit Records 4](#_Toc193039778)

[5.7 Delete Records 5](#_Toc193039779)

[5.8 View Records 5](#_Toc193039780)

[6. Defect Reporting 5](#_Toc193039781)

[7. Conclusion 5](#_Toc193039782)

[Testing Strategies 6](#_Toc193039783)

[1. Functional Testing 6](#_Toc193039784)

[2. Black Box Testing 6](#_Toc193039785)

[3. Positive and Negative Testing 6](#_Toc193039786)

[4. User Interface (UI) Testing 6](#_Toc193039787)

[5. Database Testing 6](#_Toc193039788)

[6. Error Handling and Validation Testing 6](#_Toc193039789)

[7. Regression Testing 6](#_Toc193039790)

[8. Usability Testing 7](#_Toc193039791)

[Methodologies Used 7](#_Toc193039792)

[Test log sheets 8](#_Toc193039793)

[References 13](#_Toc193039794)

# Introduction

The following document describes and demonstrates the various different techniques used within the testing process of the Assessment software. The software in question is a simple Java shell connected to a MySQL database, designed to carry out basic crud operations, and present the data to the user in a GUI Format. In order to create a comprehensive testing

## **1. Objective**

The objective of this test plan is to evaluate the functionality, usability, and data integrity of the Hospital Database Application. The testing will focus on core features, ensuring that the application behaves as expected under normal, extreme, and exceptional scenarios.

## **2. Testing Scope**

The tests will cover the following key areas:

* Application launch and closure
* Toolbar functions (View/Edit, Search, Add Records)
* Table interactions (Right-click functions)
* CRUD (Create, Read, Update, Delete) operations
* Validation and error handling

## **3. Test Strategy**

### **3.1 Testing Approach**

The following testing methods will be used:

1. **Functional Testing** – Verify that each feature works correctly.
2. **Black Box Testing** – Test input and expected output without looking at code.
3. **Validation Testing** – Ensure user inputs are properly handled.
4. **Database Testing** – Ensure correct data is stored, retrieved, and deleted.
5. **User Interface Testing** – Check application navigation and layout.
6. **Performance Testing (Basic)** – Observe responsiveness, especially with table refreshes.

## **4. Test Environment**

### **4.1 Hardware Requirements**

* Laptop/PC with Java runtime installed
* Minimum 4GB RAM, 1.5GHz processor

### **4.2 Software Requirements**

* Java Development Kit (JDK)
* Database system (MySQL/PostgreSQL)
* Java IDE (e.g., Eclipse, NetBeans)
* The application source code (MainFrame.java)

## **5. Test Cases and Execution Steps**

### **5.1 Application Launch and Closure**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Launch App** | Run MainFrame.java | Window opens with a welcome message |
| **Close App** | Navigate to File > Exit | Application closes properly |

### **5.2 Toolbar Functions**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **View/Edit Table** | Click "View/Edit" > Select a table | Table appears in the main window |
| **Search Records** | Click "Search" > Select "Patient" | Search form appears with filters |
| **Add Records** | Click "Add Records" > Select a form | Form appears for new entry |

### **5.3 Table Right-Click Functions**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Right-click Menu** | Right-click a table record | Options appear: Add, Edit, Delete, Refresh |
| **Add Record via Right-click** | Right-click > "Add" | Form appears to enter data |
| **Edit Record via Right-click** | Right-click > "Edit" | Form appears pre-filled with record details |
| **Delete Record via Right-click** | Right-click > "Delete" | Confirmation dialog appears |
| **Refresh Table** | Right-click > "Refresh" | Table reloads, showing updated records |

### **5.4 Add Records (Database Validation)**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Normal Entry** | Add valid Insurance record | Record successfully added |
| **Extreme Case** | Enter a 50-character company name | Error message appears |
| **Exceptional Case** | Enter special characters in ID | Error message appears |

### **5.5 Search Records**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Normal Search** | Search for an existing Prescription ID | Record appears in results |
| **Extreme Case** | Enter negative duration value | "No records found" message |
| **Exceptional Case** | Search for a non-existing ID | "No records found" message |

### **5.6 Edit Records**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Normal Edit** | Change hospital for a doctor | Record updates successfully |
| **Extreme Case** | Delete email field and save | Error message appears |
| **Exceptional Case** | Enter special characters in specialization field | Error message appears |

### **5.7 Delete Records**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Normal Deletion** | Delete a patient with no dependencies | Record successfully removed |
| **Extreme Case** | Delete a patient linked to other records | System allows deletion (no foreign keys) |
| **Exceptional Case** | Attempt to delete without selecting a record | Action not possible |

### **5.8 View Records**

| **Test Case** | **Steps** | **Expected Result** |
| --- | --- | --- |
| **Open Drug Table** | Click "View/Edit" > Select "Drug" | Table loads correctly |
| **Full-Screen View** | Maximize window | Table adjusts for better viewing |
| **Search Drug Table** | Search for "Ib" | Shows records with "Ib" (e.g., Ibuprofen) |
| **View Patients primary care Doctor** | Right click on patients record, an select “View Primary Doctor” | A pop up appears on screen with the doctors information |
| **View the Patients, Insurance Company information** | Right click on patients record, an select “View Insurance Company” | A pop up appears on screen with the Insurance companies information |

## **6. Defect Reporting**

* If any test case fails, log the issue with a description, steps to reproduce, and screenshots (if applicable).
* Assign severity levels (Critical, High, Medium, Low).
* Track issues until resolution.

## **7. Conclusion**

This test plan ensures that the **Hospital Database Application** is fully functional, user-friendly, and handles data correctly. By following these structured test cases, potential issues will be identified and addressed before deployment.

# Testing Strategies

## 1. Functional Testing

This ensures that the hospital database application performs as expected. The test cases evaluate different functionalities such as launching the app, searching for records, adding records, editing, and deleting data. Each function's expected output is compared with the actual result.

## 2. Black Box Testing

The tests focus on the application's behaviour rather than internal code structures. Testers interact with the software by providing inputs and verifying outputs without knowledge of the underlying implementation.

## 3. Positive and Negative Testing

* **Normal Cases:** Expected user actions that follow the intended use of the system, such as adding a record correctly.
* **Extreme Cases:** Testing boundary values, such as entering extremely long text fields or resizing the application.
* **Exceptional Cases:** Invalid inputs, such as special characters in an ID field, checking if error handling is implemented correctly.

## 4. User Interface (UI) Testing

Tests evaluate how the system presents data and whether users can navigate menus, toolbars, forms, and tables effectively. UI responsiveness (resizing the screen, right-click menus) is also tested.

## 5. Database Testing

Tests check whether operations such as adding, updating, deleting, and searching records correctly interact with the database. This includes checking referential integrity (foreign key constraints) and ensuring data persistence.

## 6. Error Handling and Validation Testing

The system's ability to reject invalid data is tested, including incorrect formatting, missing required fields, and exceeding character limits. The tests verify whether appropriate error messages appear when expected.

## 7. Regression Testing

Ensuring that previously working functionalities continue to operate correctly after new tests, particularly when editing or deleting records.

## 8. Usability Testing

While not explicitly stated, the tests indirectly assess the ease of navigation and clarity of the system’s messages, forms, and table interactions.

# Methodologies Used

The **Agile methodology** played a crucial role in the development of the **Hospital Database Application**, allowing for an iterative and flexible approach. Instead of following a rigid, linear development process, Agile enabled me to continuously refine the system by moving back and forth between different stages, such as design, implementation, and testing. This adaptability ensured that any issues discovered during testing—such as validation errors, UI inconsistencies, or database integrity problems—could be addressed promptly without delaying progress. Frequent feedback loops and incremental improvements helped enhance functionality, usability, and overall system reliability, ensuring the final product met user needs effectively.

# Test log sheets

**Candidate:**  Dominic Cameron **Project title:** Hospital Database Assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case** | **Test data** | **Expected result** | **Actual result** | **Comments** |
| Launch the app | User will launch the app by running the MainFrame.java file | The app will open up with a welcome message in the centre of the Window. | The file opens in a window, with a size of 800x600, as expected | Image[1] *See References at the bottom of the file to view the images* |
| Close the app | The User will close the window by navigating to File in the top left of the toolbar, and selecting the only choice there “Exit” | Upon clicking of Exit, the application will close the window along with all application processes. | The program terminates the window, as expected | Image[2] User navigating to the File toolbar item |
| Toolbar Function – View / Edit | User will navigate to the toolbar item “View / Edit” and select a table they would like to view i.e “Visit” | The selected table will open up in the main frame and the user can now view or edit the records. | The program displays the menu items in the toolbar, and can navigate to a table. As expected | Image[3] demonstrates the toolbar menu, Image[4] demonstrates the table. |
| Toolbar Function – Search | User will navigate to the toolbar item “Search” and select one of the Search forms i.e “Patient” | Upon selecting “Patient” a form will appear that will allow the user to filter the search using Radio buttons to select the search parameter. | The program displays the menu items in the toolbar, and opens the correct form. As expected | Image[5] displays the behaviour of toolbar menu, Image [6] Shows the search form. |
| Toolbar Function – Add Records | User will navigate to the Toolbar Item “Add records” select a desired form I.e “Doctor form” | The system will bring up a form that will allow the user to add a new record to the Doctor Table. | The program displays the Toolbar menu items correctly, and pressing on an item returns the correct form. | Image[7] Depicts the behaviour of the toolbar menu, Image [8] shows the add records form. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case** | **Test data** | **Expected result** | **Actual result** | **Comments** |
| Table Right Click function | In the Table view, users can right click on a record to bring up, option for managing the record | A record is highlighted, and a small menu appears which allows the user to either “Add ” a new record, “Edit” the selected record, “Delete” the selected record, or “Refresh Table” | The right click option brings up a menu with all of the correct items. | Image [9] Shows the layout of the right click menu. |
| Table Right Click – Add | User clicks on the “Add” option in the Right click menu, while viewing Prescription table | The system will open up a form matching the current table. And wait for user to input details and save, or cancel. | In the prescription table, right clicking, and pressing add brings up ‘Add new prescription form’ | Image[10] shows the new prescription form after clicking add in the right click menu. In the prescription table. |
| Table Right Click - Edit | User right clicks desired record, chooses “Edit” option | The system will open up a form preloaded with the current records information. To allow seamless editing. | As expected the form shows up and displays the current records info | Image [11] |
| Table Right Click - Delete | User right clicks desired record, chooses “Delete” option | The System will bring up a confirmation box that will ask the user if they’re sure they want to delete this record. | As expected the system shows a confirmation box to ensure the user definitely wants to delete that record. | Image [12] |
| Table Right Click – Refresh Table | User right clicks and selects “Refresh Table” | The system will reload the table with the results (Only noticeable when a change doesn’t show immediately) | As expected the system refreshes the table, this can be seen when a record is highlighted. Refreshing will deselect the current item. | No comment |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case** | **Test data** | **Expected result** | **Actual result** | **Comments** |
| Add record on Insurance Table | Normal: Add insurance record. Insurance ID: HGND36P Company: Cherish Insurance Scotland  address: 25 Gallowgate Phone: 964-698-2871 | A new insurance record with the Insurance ID: HGND36P should be added to the Database. | As Expected a new record is added to the database. If viewing the Insurance table user can see that the record was added to the database. | Image [13] [14] |
|  | Extreme: Add insurance record featuring a very long company name. Insurance ID: HGND37P Company Name: X(50 Character long)  address: 26 Gallowgate Phone: 964-698-2872 | The system should reject the input of a company name, as it would exceed the allowed field length. It will display an error message that explains to the user that the field length allowed has a limit. | As expected a record cannot be submitted with a longer than allowed company name. Error pops up to inform the user as can be seen in Image [16] | Image [15] & [16] |
|  | Exceptional: Add Insurance record with an ID Featuring Special characters. ID: “H$ND37!” Company name: Enable Insurance Scotland  address: 45 Gallowgate Phone: 964-698-2873 | The system should reject the input of the special characters, and which upon submission of the form will prompt the user to amend that field. | As expected the system can not accept an insurance id that features special characters [17], as seen in image [18] the System displays a dialog box that informs the user of the issue. | Image [17] & [18] |
| Search Record on the Prescription Table | Normal: Search a prescription record with the id=”1634654226” | The table search box should close and the table should retrieve the record with that ID. | As expected, the program finds the Prescription ID searched and display’s it in a Results Table. Image[20] | Image [19] & [20] |
|  | Extreme: Search a prescription record with a negative value typed into duration. Value: -1 | The program should display a message that says, no records found. | As expected the program returns no records found | Image[21] & Image[22] |
|  | Exceptional: Search a prescription record that doesn’t exist id=”984991153” | The program should display a message that says, no records found. | Unexpected, the program returns a non-existing record in the results table, featuring empty values | Not an actual match, but lack of “No records found message” Image[23] |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case** | **Test data** | **Expected result** | **Actual result** | **Comments** |
| Edit Record in the Doctor Table | Normal: Find the record with the Doctor ID: “1724” and change the Hospital to “Aberdeen Royal Infirmary” | User either manually browses the table to find the record to right click, or uses the search option to find the listing, edit the necessary field, and submit to update the record. Table refreshes upon submission | As expected user was able to find the Doctor Boothe Pavlol and add Aberdeen Royal Infirmary to the hospital field. | Image[24] [25][26] |
|  | Extreme: Find the record with the DoctorID: “9947” and change the email to “” | Once the email has been edited to contain nothing, the form will not allow the user to submit. | As expected the system will not allow the user to submit a record without an email address. And returns an error message to inform the user. | Image[27] |
|  | Exceptional: Find the record with the DoctorID: “7763” and edit the Specialization field with Invalid characters = “$p3c!4L!zed” | System should display a message to inform the user that the Specialization field has invalid characters. User won’t be able to submit the form. | As expected the system will alert the user that the specialization field must only contain letters | Image[28] |
| Delete a Record in the Patient Table | Normal: Delete a patient record PatientID = “30087242” | System will delete the user record from Database | As expected the user located the record, right clicked, pressed delete, confirmed to delete the user | Image[29][30][31][32] It took a refresh of page to display the change to the database on gui end |
|  | Extreme: Delete a patient record which has relational integrity with records in another table PatientID: “9018414” | The System will allow the record to be deleted due to no foreign keys being implemented in the database. | As expected the after finding and locating the record, deleting the record removed the record from the patient table due to no foreign key constraints in the data dictionary | Image [33][34][35] SQL queries to show the presence and deletion of the record from database. |
|  | Exceptional: Delete a record with no PatientID | Impossible to achieve, user needs to right click on the selected record to remove a record. | As expected there is no way to demonstrate this. | No commnet |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case** | **Test data** | **Expected result** | **Actual result** | **Comments** |
| View Record in the Drug Table | Normal: Open the Drug table via toolbar, “View/Edit” & select “Drug” | The Drug table will open in the main frame and allow the user to manage the records | As expected the Drug table opens without issues | Image [36] & [37] |
|  | Extreme: View all details in drug table on screen, by making the page full screen. Resize columns if necessary. | The screen will resize from the standard size to allow the user to view the records easier. | As expected after making the window full screen, the benefits were easier to read, and sorting of name column was available, resizing columns as necessary was also available | Image[38] |
|  | Exceptional: Find the records in Drug table by using the search bar to see all drug name’s containing the string “Ib” | The system will return all drugs that contain the String “Ib” within the drug name | As expected the system displayed the records containing “Ib” in the Drug name, returning one record, Ibuprofen Kids. | Image[39][40][41] |
| View Primary Care Doctor | Normal: Select a user who has had a visit in the visit records and right click the record to select “View Primary Doctor” | The system will retrieve the primary care doctor information, and display them on screen using a pop up window. | As expected the selected patients primary care doctor information is displayed on screen in a pop up format. | Image [42] [43] |
|  | Extreme: No record is selected | Impossible to achieve, the right click menu only appears after a record is selected and highlighted in blue. | As expected, user is unable to bring up the right click menu without clicking on a record. |  |
|  | Exceptional: Selected user has no visits in the visit table. | The pop up will appear to let user know that no primary care doctor has been found for the patient. | As expected the user sees a pop up that says “No primary care doctor found for this patient” | Image [44] |
| View Insurance company information | Normal: Select a user that has an Insurance ID present, and right click to retrieve more information about their insurance company. | The system will retrieve the Insurance Company information, and display them on screen using a pop up window. | As expected the selected users insurance company information appears on the screen in a pop up window | Image [45] [46] |
|  | Extreme: No record is selected | Impossible to achieve, the right click menu only appears after a record is selected and highlighted in blue. | As expected, user is unable to bring up the right click menu without clicking on a record. |  |
|  | Exceptional: Selected user has no insurance ID. | The pop up will still appear, but it will not contain any of the insurance company information. | As expected the user sees a pop up that says “No Insurance Company found for this patient” | Image [47] |

# References

|  |  |
| --- | --- |
| 1 | A screenshot of a computer  AI-generated content may be incorrect. |
| 2 | A screenshot of a computer  AI-generated content may be incorrect. |
| 3 | A screenshot of a computer  AI-generated content may be incorrect. |
| 4 | A screenshot of a computer  AI-generated content may be incorrect. |
| 5 | A screenshot of a computer  AI-generated content may be incorrect. |
| 6 | A screenshot of a computer  AI-generated content may be incorrect. |
| 7 | A screenshot of a computer  AI-generated content may be incorrect. |
| 8 | A screenshot of a computer  AI-generated content may be incorrect. |
| 9 | A screenshot of a computer  AI-generated content may be incorrect. |
| 10 | A screenshot of a computer  AI-generated content may be incorrect. |
| 11 | A screenshot of a computer  AI-generated content may be incorrect. |
| 12 | A screenshot of a computer  AI-generated content may be incorrect. |
| 13 | A screenshot of a computer screen  AI-generated content may be incorrect. |
| 14 | A screenshot of a computer  AI-generated content may be incorrect. |
| 15 | A screenshot of a computer screen  AI-generated content may be incorrect. |
| 16 | A screenshot of a computer error  AI-generated content may be incorrect. |
| 17 | A screenshot of a computer  AI-generated content may be incorrect. |
| 18 | A screenshot of a computer error  AI-generated content may be incorrect. |
| 19 | A screenshot of a computer  AI-generated content may be incorrect. |
| 20 | A screenshot of a computer  AI-generated content may be incorrect. |
| 21 | A screenshot of a computer  AI-generated content may be incorrect. |
| 22 | A screenshot of a computer error message  AI-generated content may be incorrect. |
| 23 | A screenshot of a computer  AI-generated content may be incorrect. |
| 24 | A screenshot of a computer  AI-generated content may be incorrect. |
| 25 | A screenshot of a computer  AI-generated content may be incorrect. |
| 26 | A screenshot of a computer  AI-generated content may be incorrect. |
| 27 | A screenshot of a computer  AI-generated content may be incorrect. |
| 28 | A screenshot of a computer screen  AI-generated content may be incorrect. |
| 29 | A screenshot of a computer  AI-generated content may be incorrect. |
| 30 | A screenshot of a computer  AI-generated content may be incorrect.A screenshot of a computer error  AI-generated content may be incorrect. |
| 31 | A screenshot of a computer  AI-generated content may be incorrect.  A screenshot of a computer  AI-generated content may be incorrect. |
| 32 | A black background with white text  AI-generated content may be incorrect. |
| 33 | A black screen with white text  AI-generated content may be incorrect. |
| 34 | A screenshot of a computer  AI-generated content may be incorrect. |
| 35 | A computer screen with white text  AI-generated content may be incorrect. |
| 36 | A screenshot of a computer  AI-generated content may be incorrect. |
| 37 | A screenshot of a computer  AI-generated content may be incorrect. |
| 38 | A screenshot of a computer  AI-generated content may be incorrect. |
| 39 | A screenshot of a computer  AI-generated content may be incorrect. |
| 40 | A screenshot of a computer  AI-generated content may be incorrect. |
| 41 | A computer screen shot of a browser  AI-generated content may be incorrect. |
| 42 |  |
| 43 |  |
| 44 |  |
| 45 |  |
| 46 |  |
| 47 |  |
| 48 |  |
| 49 |  |