Comprehensive Test Document for Blood Bank Management System

1. Introduction

The Blood Bank Management System is a web application designed to facilitate the efficient management of blood donations, requests, and inventory. This document outlines detailed test cases to verify the functionality, usability, and reliability of the system.

Objective

The primary objective of this testing process is to:

- Verify the accuracy of functionalities such as registration, login, and profile management.
- Ensure seamless integration of database operations with front-end features.
- Validate the end-to-end workflows from registration to request fulfillment.

Scope

The system under test includes:

- User interfaces for hospitals and receivers.
- Blood inventory management.
- Request and approval/rejection workflows.
- Profile management for both hospitals and receivers.

2. Pre-requisites

- Install a web server (e.g., Apache) and configure it with PHP.
- Import the provided ‘bloodbank.sql‘ database dump into MySQL.
- Ensure the server connection details in ‘connection.php‘ are correctly configured.
- Load the system files into the web server’s root directory.

3. Functional Test Cases

The following test cases focus on the core functionalities of the system:

3.1 Registration

Test ID	Description	Expected Result	Status
TC-001	Register a new hospital with valid inputs.	Registration is successful, and a success message is displayed.	Pass/Fail
TC-002	Attempt to register a hospital with an already registered email.	Registration fails, and an error message is displayed.	Pass/Fail
TC-003	Register a new receiver with valid inputs.	Registration is successful, and a success message is displayed.	Pass/Fail
TC-004	Attempt to register with missing required fields.	Registration fails with appropriate error messages.	Pass/Fail

3.2 Login

Test ID	Description	Expected Result	Status
TC-005	Login with correct credentials as a hospital.	Redirect to the hospital dashboard.	Pass/Fail
TC-006	Login with correct credentials as a receiver.	Redirect to the receiver dashboard.	Pass/Fail
TC-007	Attempt login with incorrect credentials.	Login fails, and an error message is displayed.	Pass/Fail
TC-008	Attempt login without providing credentials.	Login fails with appropriate error messages.	Pass/Fail

3.3 Blood Management

Test ID	Description	Expected Result	Status
TC-009	Add a new blood group as a hospital.	Blood group is successfully added and displayed in the hospital's inventory.	Pass/Fail
TC-010	Attempt to add a duplicate blood group.	Operation fails with a duplicate entry error.	Pass/Fail
TC-011	View available blood samples from the hospital's dashboard.	All blood samples are displayed accurately.	Pass/Fail

4. Database Validation

After performing the functional tests, validate the database for consistency:

- Ensure newly registered users appear in the ‘hospitals‘ or ‘receivers‘ table.
- Verify that blood groups added are stored in the ‘bloodinfo‘ table with accurate relationships to hospital IDs.
- Confirm request status updates (e.g., ‘Pending‘, ‘Accepted‘, ‘Rejected‘) are correctly reflected in the ‘bloodrequest‘ table.

5. Integration Testing

5.1 End-to-End Workflow: Blood Request Fulfillment

1. **Register:** A receiver registers and logs into the system.
2. **View Blood Samples:** The receiver browses available blood samples.
3. **Request Blood:** The receiver submits a request for a specific blood group.
4. **Respond to Request:** The hospital accepts or rejects the request.
5. **Database Validation:** Confirm the request status in the database.

6. Performance Testing

6.1 Load Testing

Simulate 100 concurrent users accessing the system:

- Measure response times for key operations (e.g., login, data retrieval).
- Ensure the system remains stable under load.

6.2 Stress Testing

Gradually increase user load beyond system capacity:

- Identify breaking points.
- Verify the system fails gracefully without data corruption.

7. Conclusion

The Blood Bank Management System testing ensures that the platform meets functional requirements, is robust under load, and provides a seamless user experience. Further improvements can be identified and implemented based on test results and user feedback.