

American International University-Bangladesh (AIUB)

# Department of Computer Science Faculty of Science & Technology (FST)

Blood Donation Organization

A Software Quality and Testing Project Submitted By:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Semester: Spring\_24\_25** | | | | **Section:** | **Group No:** |
| **SL** | **SN** | **Student Name** | **Student ID** | Individual Contribution (in %) | Total Marks: 50 |
| Earned Marks: |
| **A** | 1 | A.B.M Mahmudul Hasan | 22-47108-1 | 25% (Test Plan Architecture, Led test cases, introduction, revisions) |  |
| **B** | 2 | Sabbir Ahmed | 22-46486-1 | 25% (Figma Design, Testing Strategy, testing approach, revision) |  |
| **C** | 3 | Saif Mahmud Shihab | 22-46830-1 | **25%** (Requirement specs, staffing/schedule, revisions) |  |
| **D** | 4 | Shahriar Hossen | 22-46525-1 | **25%** (Pass/fail criteria, risk management, revision) |  |

The project will be Evaluated for the following Course Outcomes

|  |  |  |
| --- | --- | --- |
| **EVALUATION CRITERIA** | **Total Marks (50)** | |
|  | |
| Revision History, Test Plan Identifier, Reference Materials, Problem Background, Solutions | [10 Marks] |  |
| Requirements Specification (System feature, Quality Attributes, System Interface, Project Requirements) | [10 Marks] |  |
| Item Not to be tested, Testing approach (Testing levels, tools, meetings), Test cases | [10 Marks] |  |
| Item pass/fail criteria, Test deliverables, Staffing and Training, Responsibilities, Scheduling, Risk | [10 Marks] |  |
| Approval, Format, Submission, and Defense | [10 Marks] |  |

Software Test Plan for

<Hatibandha Blood Donation Organization (HBDO)>

Version 1.0 approved

Prepared by <>

<American International University-Bangladesh>

<19/6/2025>

Table of Contents

# [Revision History 3](#_bookmark0)

# [TEST PLAN IDENTIFIER: AT-TP01.3 4](#_bookmark1)

# [REFERENCE MATERIALS 4](#_bookmark2)

# [INTRODUCTION 4](#_bookmark3)

* 1. [Background to the Problem 4](#_bookmark4)
  2. [Solution to the Problem 5](#_bookmark5)

# [REQUEIREMNT SPECIFICATION 5](#_bookmark6)

* 1. System Features **Error! Bookmark not defined.**
  2. [System Quality Attributes 9](#_bookmark7)
  3. System Interface **Error! Bookmark not defined.**
  4. Project Requirements **Error! Bookmark not defined.**

1. **FEATURES NOT TO BE TESTED** Error! Bookmark not defined.
2. **TESTING APPROACH** Error! Bookmark not defined.
   1. Testing Levels **Error! Bookmark not defined.**
   2. Test Tools **Error! Bookmark not defined.**
   3. Meetings **Error! Bookmark not defined.**
3. **TEST CASES/TEST ITEMS** Error! Bookmark not defined.
4. **ITEM PASS/FAIL CRITERIA** Error! Bookmark not defined.
5. **TEST DELIVERABLES** Error! Bookmark not defined.
6. **STAFFING AND TRAINING NEEDS** Error! Bookmark not defined.
7. **RESPONSIBILITIES** Error! Bookmark not defined.
8. **TESTING SCHEDULE** Error! Bookmark not defined.
9. **PLANNING RISKS AND CONTINGENCIES** Error! Bookmark not defined.
10. **APROVALS** Error! Bookmark not defined.

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Updated by** | **Update Comments** |
| 0.1 | 2025.05.30 | A.B.M Mahmudul Hasan & Saif Mahmud Shihab | Initial draft of Test Plan (Sections 1–3: Identifier, References, Introduction). |
| 0.2 | 2025.06.01 | Sabbir Ahmed | Added **Requirement Specifications**  (System Features, Quality Attributes). |
| 0.3 | 2025.06.03 | Saif Mahmud Shihab | Updated **Testing Approach** (Levels, Tools, Meetings) and **Features Not Tested**. |
| 0.4 | 2025.06.04 | A.B.M Mahmudul Hasan | Finalized **Test Cases** (20 functional + 5 non-functional). |
| 0.5 | 2025.06.05 | Shahriar Hossen | Added **Pass/Fail Criteria**, **Test Deliverables**, and **Risk Management**. |
| 0.6 | 2025.06.06 | Sabbir Ahmed & Shahriar Hossen | Completed **Staffing**, **Schedule**, and  **Approval** sections. |

TEST PLAN IDENTIFIER: STC-TC-TP-01

STC: Hatibandha Blood Donation Organization (HBDO)

TC: Test Case

TP: Test Plan

01: First test cycle

REFERENCE MATERIALS

* Software Requirement Specification (SRS) Document
* Software Requirement documentation
* Software Quality and Testing Course PowerPoint Slides

INTRODUCTION

* 1. Background to the Problem

In the realm of public healthcare and community welfare, blood donation remains one of the most critical and life-saving activities. Despite its importance, many local blood donation initiatives struggle due to the absence of centralized systems for donor registration, request coordination, and data management.

This challenge is especially pronounced in rural and semi-urban areas, where people in urgent need of blood often face delays, misinformation, and difficulty locating matching donors in time. Additionally, manual record-keeping and fragmented communication between donors and recipients increase the risk of mismanagement and donor fatigue.

The Hatibandha Blood Donation Organization (HBDO) portal has been launched to solve the urgent requirement of an optimized process for blood donation. The online system provides all the aspects under one roof where the users can:

* + - Register as blood donors
    - Request blood with detailed criteria
    - Find the donors by blood type and location
    - Maintain their own records of donations
    - Submit reviews and provide community feedback

It simplifies the entire donation and request process, especially for students, local volunteers, and healthcare helpers, ensuring that blood reaches those in need with speed, transparency, and accountability.

The need for HBDO stems from the following key challenges faced by local donation efforts:

1. Lack of a Centralized Donor Management System. Many areas rely on Facebook posts, personal contact lists, or word-of-mouth to find blood donors. This is inefficient, slow, and unreliable during emergencies.
2. Ineffective Communication between Donors and Recipients. There is no structured platform to bridge the gap between blood seekers and potential donors. Manual communication often results in missed opportunities or duplication of effort.
3. Limited Access to Real-Time Donation Data. Without automated systems, there's no way to track how many donations have occurred, who has donated recently, or how many bags of blood are available. This creates transparency issues and limits planning for donation drives.
4. Low Engagement and Motivation for Donors. Donors often don’t receive feedback or see their impact. Features like review submissions, donation history, and achievements are missing in manual systems, reducing motivation.
   1. . To find the critical shortcomings in the administration of blood donation systems, the Hatibandha Blood Donation Organization (HBDO) project proposes a centralized, web-based blood donation platform designed specifically to support local communities, especially in under-resourced or semi-urban regions.

This system functions as a digital bridge between blood donors and recipients, offering features that streamline the end-to-end donation process, from registration to review submission. Through automated workflows, dynamic filtering, and real-time data tracking, HBDO ensures timely access to blood, enhances community trust, and improves donor engagement.

The system provides an intuitive interface where users can:

1. Register and manage donor profiles with essential data such as blood group, location, and contact details
2. Search for available donors filtered by blood group, division, district, and upazila
3. Request blood through a structured form including urgency, number of bags, and hospital details
4. Add personal donation history with proof (e.g., photo uploads, date of donation)
5. View all past donations, improving transparency and tracking
6. Edit personal profiles to maintain updated contact/location info
7. Change passwords securely and submit reviews to strengthen community feedback
8. REQUEIREMNT SPECIFICATION
   1. System Feature:
      * User Dashboard
        1. System Login Functional Requirement.

Description: The application will enable registered users (donors) to access their accounts by logging in with their email and password.

If the provided credentials are invalid, an error message will be shown, and access will be denied.

Optional feature: To enhance security, the system may temporarily lock the account after five unsuccessful login attempts to prevent misuse.

Priority Level: High

Precondition: User must have a registered and verified email/password.

Cross Reference: Login Page UI, TC\_FR\_AUTH\_003 (Login), TC\_FR\_AUTH\_004 (Failed Attempts).

* + - 1. User Registration Functional Requirement

Description: The function shall allow new users to register by entering their full name, email, phone number, address, blood group, and password.

The registration is successful only if the email/phone is not already in use and the blood group is selected from a predefined list.

Priority Level: High

Precondition: Valid email and all required form fields completed.

Cross Reference: Registration Page UI, TC\_FR\_AUTH\_001 (Registration), TC\_FR\_AUTH\_002 (Duplicate Check).

* + - 1. Password Change Functional Requirement

Description: Logged-in users can securely change their account password by providing the old password and entering a new one.

A success message confirms the change. Priority Level: Medium

Precondition: The user must be verified, and the current password must be verified as correct.

Cross Reference: Profile Settings → Change Password Page, TC\_FR\_CMN\_005.

* + - 1. Blood Search Functional Requirement

Description: The system shall allow users to search for donors based on selected blood group, division, district, and upazila.

Search results are displayed in a table, showing donor name, contact info, and location. Priority Level: High

**Precondition**: User is logged in.

**Cross Reference**: Search Blood Page, TC\_FR\_CMN\_001

# Blood Request Functional Requirement

**Description**: Users can fill out a form to request blood by specifying patient name, hospital, required blood group, date, number of bags, and location details.

The system stores this request and marks it for donor visibility.

**Priority Level**: High

**Precondition**: Logged-in user with all form fields filled correctly.

**Cross Reference**: Request Blood Page, TC\_FR\_CMN\_002

# Donation Submission Functional Requirement

**Description**: Users can log their blood donation by submitting date, location, number of bags, and optionally uploading a picture of the donation receipt.

**Priority Level**: Medium

**Precondition**: Logged-in user

**Cross Reference**: Add Donation Page, TC\_FR\_CMN\_003

# Profile Update Functional Requirement

**Description**: Users may update their profile information like as mobile number, address, or blood group via the Edit Profile page.

The system confirms changes with a success message.

**Priority Level**: Medium

**Precondition**: User must be logged in

**Cross Reference**: Edit Profile Page, TC\_FR\_CMN\_004

# Donation History Functional Requirement

**Description**: The system displays a user’s previous donations including date, place, number of

bags, and uploaded proof (if any).

History is shown in a chronological, readable table format.

**Priority Level**: Medium

**Precondition**: Logged-in user with at least one recorded donation.

**Cross Reference**: Donation History Page, TC\_FR\_CMN\_006

# Review Submission Functional Requirement

**Description**: Users may submit a star rating (1 to 5) along with a comment or feedback about the platform after using its services.

Submitted reviews are visible on the homepage or dashboard for other users.

**Priority Level**: Low

**Precondition**: Logged-in user

**Cross Reference**: Review Page, TC\_FR\_USER\_001

* + - Admin Dashboard

The admin dashboard includes the following features that function identically to the user version:

* + - Blood Search
    - Blood Request
    - Edit Profile
    - Change Password
    - Add Donation
    - Donation History Unique features:

1. Pending Blood Request Management Functional Requirement

**Description**: The system shall display all pending blood requests with complete details including requester information, blood type needed, quantity, location, and status. Admin can approve, reject, or modify requests.

**Priority Level**: High

**Precondition**: Admin must be logged in

**Cross Reference**: Pending Blood Request Page, TC\_FR\_ADMIN\_003

1. Latest Donations Monitoring Functional Requirement

**Description**: The system shall display all recent donations in chronological order with donor details, blood type, quantity, and location. Admin can verify and flag donations if needed.

**Priority Level**: High

**Precondition**: Admin must be logged in

**Cross Reference**: Latest Donations Page, TC\_FR\_ADMIN\_004

1. Member Management Functional Requirement

**Description**: The system will allow admins to add new members with complete profile information including personal details, contact information, blood type, and geographic location. All fields must pass validation checks.

**Priority Level**: High

**Precondition**: Admin must be logged in

**Cross Reference**: Add Member Page, TC\_FR\_ADMIN\_001

1. User Search Functional Requirement

**Description**: The system should provide search functionality to find users by email, name, or phone number. Admin can perform account management actions including deletion and role promotion with appropriate confirmation dialogs.

**Priority Level**: High

**Precondition**: Admin must be logged in

**Cross Reference**: Search Member Page, TC\_FR\_ADMIN\_005

1. Monthly Reporting Functional Requirement

**Description**: The system will generate monthly reports showing user activity and donation points earned. Reports should be filterable by date range and exportable.

**Priority Level**: Medium

**Precondition**: Admin must be logged in

**Cross Reference**: Monthly Reports Page, TC\_FR\_ADMIN\_006

1. Notification System Functional Requirement

**Description**: The system will allow admins to create notifications with title, message, and optional image attachment. Submitted notifications shall immediately appear on the noticeboard. After 24 hours of publication, the system shall automatically move the notification to the archive section while preserving all content.

**Priority Level**: Medium

**Precondition**: Admin must be logged in

**Cross Reference**: Send Notice Page, TC\_FR\_ADMIN\_002

# System Quality Attributes

QA1 – Usability

# Description:

The system will maintain intuitive navigation with task completion (registration, blood search, requests) achievable within 3 minutes by first-time users. Mobile interfaces shall comply with responsive design standards.

**Priority Level**: High

**Precondition**: Accessible via desktop/mobile browsers

# Test Reference:

* TC\_NFR\_USAB\_001 (Mobile Responsiveness)

**Cross Reference**: QA3, QA6 QA2 – Performance

# Description:

Key pages will load ≤2s; form submissions complete ≤3s under normal conditions. Sustains 500

concurrent users without degradation.

**Priority Level**: High

**Precondition**: Optimized server/database

# Test Reference:

* TC\_NFR\_PERF\_001 (500 Concurrent Users)

**Cross Reference**: QA5, QA7 QA3 – Reliability

# Description:

Ensures 99.5% monthly uptime with ≤15m outages per incident. Automated recovery

mechanisms for critical failures.

**Priority Level**: High

**Precondition**: Active hosting platform

# Test Reference:

* TC\_NFR\_REL\_001 (System Recovery)

**Cross Reference**: QA1, QA4 QA4 – Maintainability

# Description:

Modular codebase enables isolated updates (e.g., profile editor) without system-wide impact.

**Priority Level**: Medium **Precondition**: Clean code architecture **Cross Reference**: QA3, QA6

QA5 – Security

# Description:

Prevents injection attacks via input sanitization. Enforces crypt hashing + HTTPS. Role-based access controls.

**Priority Level**: High

**Precondition**: SSL installed, server validation

# Test Reference:

* TC\_NFR\_SEC\_001 (Role-Based Access)
* TC\_NFR\_SEC\_002 (Password Encryption)

**Cross Reference**: QA2, QA7 QA6 – Scalability

# Description:

Handles 500+ concurrent users. Design supports seamless server upgrades.

**Priority Level**: Medium

**Precondition**: Scalable frontend/backend

**Cross Reference**: QA1, QA4 QA7 – Accuracy

# Description:

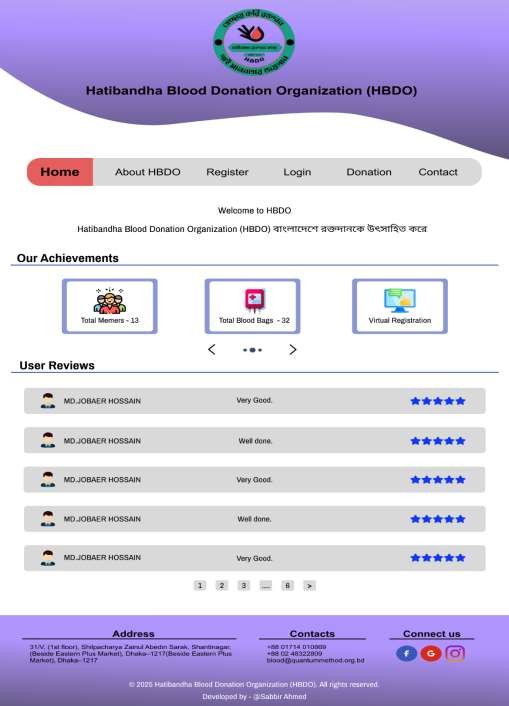
Guarantees precise data recording with validation-triggered error messaging for invalid inputs.

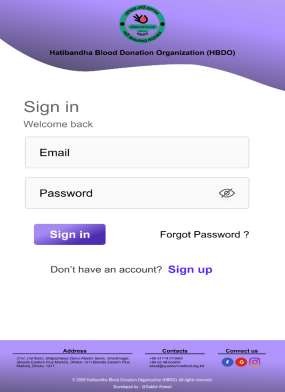
**Priority Level**: High

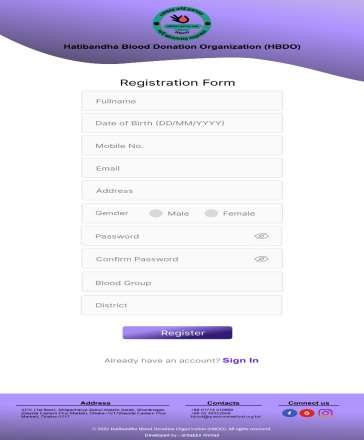
**Precondition**: Client/server validation

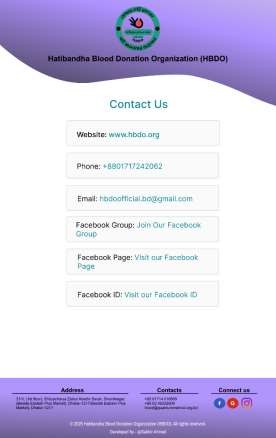
**Cross Reference**: QA2, QA5

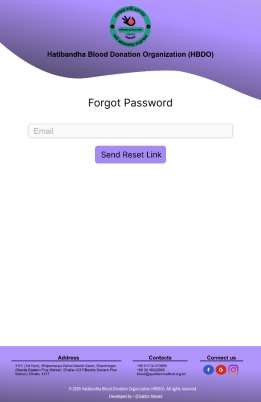
**4.3. System Interface:**

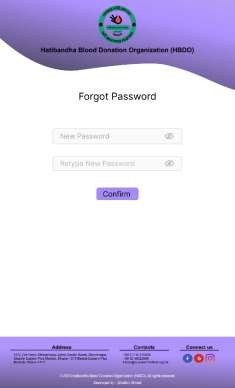
****

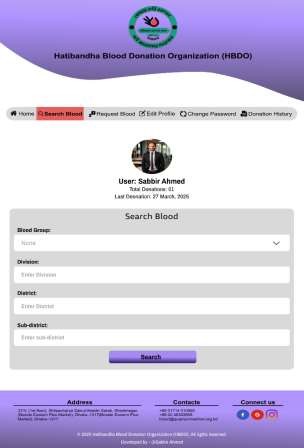


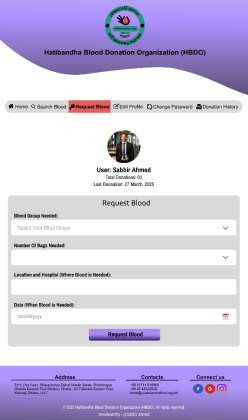
****



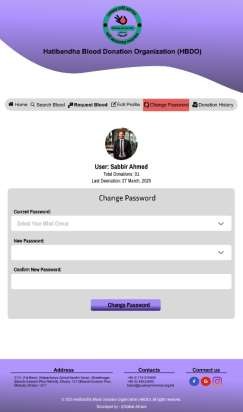
****

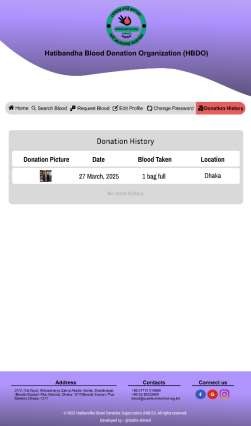


****



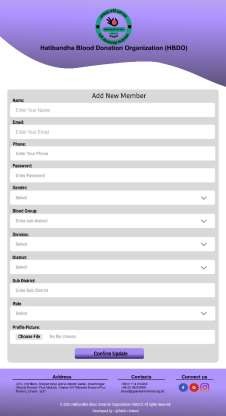
****



****



****



****

4.4. Project Requirements: Time Estimation Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Day | Phase | Tasks | Team | Tools Used | Deliverables |
| Day 1 | Preparation | * Finalize test cases * Setup InfinityFree test environment | QA Lead (1) | Google Docs, GitHub | Approved test plan |
| Day 2 | Test Data Prep | * Create 20 mock profiles * Prepare blood request samples | QA Lead (1) | Excel, JSON generator | Test datasets (CSV) |
| Day 3 | Unit Testing | * Auth module (login/reg) * Form validations | QA Lead + Volunteer (2) | Chrome DevTools | Unit test logs |
| Day 4 | Integration Testing | * Donor search   + request flow   * Profile updates | QA Lead + Dev (2) | Postman, GitHub Issues | Integration test report |
| Day 5 | System Testing | * Full donor- request cycle * Mobile responsiveness | Full Team (3) | Mobile emulators | System test summary |
| Day 6 | UAT | * 3 volunteers test core features * WhatsApp feedback collection | Volunteers + QA (4) | Google Forms | UAT  feedback sheets |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Day 7 | Reporting | * Bug fixes verification * Final report compilation | QA Lead (1) | MS Word | Printed test report (300 BDT) |

Budget Estimation:

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | **Item** | **Details** | **Estimated Cost (BDT)** |
| **1. Infrastructure** | Domain & Hosting | Free via 000webhost / InfinityFree | 0 |
| **2. Development Tools** | Code Editor | Visual Studio Code (Free) | 0 |
|  | Version Control | Git & GitHub (Free) | 0 |
|  | Email API | SMTP / Mailtrap (Free tier) | 0 |
| **3. Testing Tools** | Postman, DevTools | API and UI testing tools (Free) | 0 |
|  | Manual Testing | Conducted by QA using browser tools | 0 |
| **4. Documentation** | Microsoft Word | Academic documentation (pre-installed in lab PC) | 0 |
|  | Printing & Binding | Final test plan and report printout | 300 |
| **5. Communication** | Google Meet, WhatsApp | Team coordination | 0 |
| **6. Training Materials** | SQA Course Slides, Templates | Provided by university | 0 |
| **7. Miscellaneous** | Contingency (e.g., data pack) | Backup internet/power for submissions | 500 |
| **Total Estimated Cost** |  |  | **800 BDT** |

# FEATURES NOT TO BE TESTED

**Third-Party Authentication Services**

If any OAuth or external authentication methods (e.g., Google/Facebook login) are integrated in future, the internal mechanisms of those services will not be tested. Only the **redirect process and success/error status** from those services will be verified.

# Hosting Server Infrastructure & Performance

The system is hosted on a **free web hosting platform (infinityfree)**. The **uptime, latency, and load-balancing capacity of the hosting provider** will not be evaluated or tested directly, as it falls under the provider’s responsibilities.

# Cross-Browser Compatibility (Legacy Browsers)

The application will not **undergo testing on outdated or legacy browsers** like Internet Explorer 10 or earlier. Compatibility checks will be restricted to the most **recent versions** of **modern browsers** such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.

# Mobile Responsiveness on Rare Devices

Basic mobile responsiveness is tested using Chrome Developer Tools and common screen sizes. However, testing will **not cover all mobile or tablet devices**, especially rare screen resolutions, older mobile OS versions, or smart TVs due to time and resource limitations.

# Automated Data Backup & Recovery

If a server-side or hosting-level backup and recovery process is in place, it will **not be tested** as part of this scope. It is assumed to be configured and managed by the deployment/hosting platform.

# Email API Internal Behavior

While the **sending of emails via contact forms or submission acknowledgments** will be tested for functional confirmation (e.g., delivery of success message), the **internal operation of the Email API (e.g., SMTP routing)** will not be validated.

# Multilingual/Regional Language Support

This version of the system supports only a **single default language (English)**. If additional language support (e.g., Bengali, Arabic) is planned for future phases, it will not be tested in the current version.

* 1. **Testing Levels**

**Unit Testing**

Each core module of the HBO Blood Organization website will be tested independently by the developer using mock or test data to ensure the correctness of the following functionalities:

* + - Login and Registration Validation Logic
    - Form Input Handlers (e.g., Request Blood, Add Donation)
    - Profile Editing and Password Update Mechanisms
    - Search and Filter Features (by Blood Group/Region)

**Tool Used**: Physical testing using browser developer tools (e.g., Chrome DevTools), and optionally, testing utilities like Postman for API endpoints if applicable.

**Integration Testing**

Modules that are dependent on each other will be tested together to validate **data consistency and logical flows**. This includes:

* + - Registration → Login → Dashboard access
    - Request Blood → Display in System
    - Add Donation → View in Donation History
    - Submit Review → Display on Homepage

Focus will be placed on:

* + - **Data flows between pages**
    - **Database insert/update correctness**
    - **Session and authentication handling System Testing**

The full website will be tested end-to-end in a simulated production environment. This level ensures that all features work cohesively, validating the entire system.

**Scope**:

* + - Registration → Login → Profile Setup
    - Search Donor → Request Blood → Confirmation
    - Add Donation → View Donation History
    - Submit Review → Display on UI

All system-level functions will be tested for **usability, responsiveness, and error handling**. **Acceptance Testing**

This final level of testing will be conducted by actual users or the project stakeholders (e.g., classmates, instructors, client). The purpose is to **ensure that the system fulfills the original project requirements** and is ready for public use or submission.

**Outcome**:

* + - “Go” if the application meets functional, UI/UX, and reliability expectations
    - “No-Go” if any critical features fail or bugs prevent usage
  1. Test Tools

|  |  |
| --- | --- |
| **Purpose** | **Tool(s)** |
| **Project Management** | Manual tracking, Google Sheets |
| **UI Design** | Figma (for early wireframes/mockups) |
| **Version Control** | Git + GitHub |
| **Unit Testing** | Manual (JavaScript form logic), optionally PHPUnit (if using PHP for backend) |
| **API Testing** | Postman (for testing Contact Form / Email API endpoints) |
| **Bug Tracking** | GitHub Issues |
| **Documentation** | Microsoft Word, PDF Export |

* 1. **Meetings**

The testing process for the Hatibandha Blood Donation Organization (HBDO) platform follows a structured meeting schedule to ensure continuous collaboration, visibility into test progress, and timely resolution of issues.

**Daily Standups (15 minutes)**

Short, focused meetings held each day during the testing phase. The goal is to maintain momentum and uncover issues early.

**Discussion Points:**

* + - What was completed yesterday?
    - Are there any current blockers or bugs?
    - What tasks are planned for today?

**Weekly Team Meetings**

Conducted every 7 days to evaluate overall progress and ensure alignment between testers and developers.

**Objectives:**

* + - Review of completed test cases and pending ones
    - Prioritize and categorize open bugs (critical, major, minor)
    - Prepare a plan for regression and retesting cycles

**Milestone Reviews (Bi-weekly)**

Milestone-based meetings are held at major checkpoints (e.g., completion of registration flow testing, donor module testing, etc.)

**Coverage:**

* + - Validate readiness of core features before integration/system testing
    - Conduct live walkthroughs or demos for stakeholders
    - Prepare for user acceptance testing (UAT)

**Agenda Highlights (Applicable to All Meetings)**

* + - Test case execution status
    - Number and severity of open defects
    - Environmental or hosting issues (e.g., email API not working)
    - Feedback from team members or external users (if applicable)

1. **Test Cases**

**User, Admin Common Functionalities**

**Test Case 1. Registration Functionality**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_001 |
| **Test Case Title** | Verify New User Registration |
| **Related Requirement** | FR-AUTH-001: Users must register with valid details. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. System is accessible. 2. Email not previously registered. |

|  |  |
| --- | --- |
| **Test Data** | Name: "Soumik Hasan", Email: "[soumikhasan000@gmail.com](mailto:soumikhasan000@gmail.com)", Blood Group: "A+", Password: "Ss@1234" |
| **Test Steps** | 1. Fill all mandatory fields. 2. Upload profile picture. 3. Click "Register". |
| **Expected Result** | Account created. Confirmation email sent. |
| **Actual Result** | Account created successfully. |
| **Status** | Pass |
| **Remarks** | Verify database record creation. |

**Test Case 2: Registration with Existing Email and Phone Number**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_002 |
| **Test Case Title** | Prevent Duplicate Registration by Email and Phone |
| **Related Requirement** | FR-AUTH-002: System must reject duplicate email/phone registrations. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. User with email "[soumik@gmail.com](mailto:soumik@gmail.com)" and phone "+8801712345678" already registered. |
| **Test Data** | Scenario 1 (Duplicate Email):   * Email: "[existing@gmail.com](mailto:existing@gmail.com)"   - Phone: "+8801711143278" (new)  Scenario 2 (Duplicate Phone):   * Email: "[new@gmail.com](mailto:new@gmail.com)"   - Phone: "+8801712345678" |
| **Test Steps** | For both scenarios:   1. Fill the registration form with duplicate data. 2. Click "Register". |
| **Expected Result** | Scenario 1:  - Error: "Email already registered" |

|  |  |
| --- | --- |
|  | Scenario 2:  - Error: "Phone number already registered" |
| **Actual Result** | Email already registered. |
| **Status** | Pass |
| **Remarks** | Critical Checks:   1. The verify database remains unchanged. 2. Test with:   - Case variations (e.g., [Existing@gmail.com](mailto:Existing@gmail.com))  - Phone formats (+88017, 017, 88017) |

**Test Case 3: Login Functionality**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_003 |
| **Test Case Title** | Verify Successful Login |
| **Related Requirement** | FR-AUTH-003: Users must access the system with valid credentials. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. A valid user account exists. |
| **Test Data** | Email: "[valid@gmail.com](mailto:valid@gmail.com)", Password: "ValidPass123!" |
| **Test Steps** | 1. Enter valid credentials. 2. Click "Login". |
| **Expected Result** | Redirect to dashboard. Session cookie created. |
| **Actual Result** | Redirect to dashboard. |
| **Status** | Pass |
| **Remarks** | Check session duration in browser. |

**Test Case 4: Invalid Login Attempts**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_004 |
| **Test Case Title** | Verify Account Lock After 5 Failed Attempts |
| **Related Requirement** | FR-AUTH-004: System must prevent brute-force attacks. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. Valid account exists ("[user@gmail.com](mailto:user@gmail.com)"). |
| **Test Data** | Email: "[user@gmail.com](mailto:user@gmail.com)", Wrong Password: "Wrong123" (x5) |
| **Test Steps** | 1. Enter incorrect password 5 times consecutively. |
| **Expected Result** | 5th attempt: "Account locked for 1 hour". |
| **Actual Result** | Account locked for 1 hour. |
| **Status** | Pass |
| **Remarks** | Verify unlock after 1 hour. |

**Test Case 5: Forgot Password**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_005 |
| **Test Case Title** | Verify Password Reset Link Generation |
| **Related Requirement** | FR-AUTH-005: Users must recover accounts via email. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | Account "[recover@gmail.com](mailto:recover@gmail.com)" exists. |
| **Test Data** | Email: "[recover@gmail.com](mailto:recover@gmail.com)" |

|  |  |
| --- | --- |
| **Test Steps** | 1. Click "Forgot Password". 2. Enter an email. 3. Click "Send Link". |
| **Expected Result** | Email sent with valid reset link (check inbox). |
| **Actual Result** | Email sent with valid reset link. |
| **Status** | Pass |
| **Remarks** | Link should contain secure token. |

**Test Case 6: Expired Reset Link**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_006 |
| **Test Case Title** | Validate Reset Link Expiration |
| **Related Requirement** | FR-AUTH-006: Reset links must expire after 24 hours. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | Reset link generated >24 hours ago. |
| **Test Data** | Expired reset link URL |
| **Test Steps** | 1. Click expired link. 2. Attempt password reset. |
| **Expected Result** | Error: "This link has expired. Request a new one." |
| **Actual Result** | Link has expired. |
| **Status** | Pass |
| **Remarks** | Verify system logs expiration events. |

**Test Case 7: Logout Functionality**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_AUTH\_007 |
| **Test Case Title** | Verify Logout Functionality for All Roles |
| **Related Requirement** | FR-AUTH-007: System must terminate sessions upon logout. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. User/Admin/Super Admin is logged in. |
| **Test Data** | Active sessions for:   * User: [user@gmail.com](mailto:user@gmail.com) * Admin: admin@ gmail.com * Super Admin: superadmin@ gmail.com |
| **Test Steps** | For each role:   1. Perform role-specific actions (e.g., request blood, approve requests, view logs). 2. Click "Logout". 3. Attempt to navigate back via browser history. |
| **Expected Result** | 1. Session terminated. 2. Redirected to login page. 3. Restricted pages show "Session expired" if accessed post-logout. |
| **Actual Result** | Redirected to login page. |
| **Status** | Pass |
| **Remarks** | Security Checks:   * Verify session token invalidation. * Test on multiple browsers/devices. |

**Test Case 8. Search Blood Functionality**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_CMN\_001 |
| **Test Case Title** | Verify Blood Search with Valid Filters |
| **Related Requirement** | FR-CMN-001: Users must search for blood by location and blood group. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. User is logged in. 2. Blood donation data exists in the system. |
| **Test Data** | Blood Group: A+, Division: Dhaka, District: Dhaka, Upazila: Mirpur |
| **Test Steps** | 1. Navigate to "Search Blood". 2. Select blood group and location filters. 3. Click "Search". |
| **Expected Result** | System displays matching blood donors/availability. |
| **Actual Result** | System displays matching blood donors. |
| **Status** | Pass |
| **Remarks** | Test with invalid filters (should show "No results"). |

**Test Case 9. Request Blood Functionality**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_CMN\_002 |
| **Test Case Title** | Verify Blood Request Submission |
| **Related Requirement** | FR-CMN-002: Users must request blood for patients. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Critical |
| **Preconditions** | 1. User is logged in. |

|  |  |
| --- | --- |
| **Test Data** | Blood Group: B+, Patient's Disease: "Thalassemia", Bags: 2, Location: "Dhaka Medical College", Date: 20-Jul-2024 |
| **Test Steps** | 1. Navigate to "Request Blood". 2. Fill all mandatory fields. 3. Click "Request Blood". |
| **Expected Result** | Request is submitted. Confirmation message: "Blood request received." |
| **Actual Result** | Blood request received. |
| **Status** | Pass |
| **Remarks** | Test with incomplete fields (should show validation errors). |

**Test Case 10. Add Donation Functionality**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_CMN\_003 |
| **Test Case Title** | Verify Donation Record Creation |
| **Related Requirement** | FR-CMN-003: Users must log their blood donations. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. User is logged in. |
| **Test Data** | Blood Group: O+, Bags: 1, Disease: "Emergency", Location: "Chattogram", Date: 15-Jul-2024, Donation Picture: [donation.jpg] |
| **Test Steps** | 1. Navigate to "Add Donation". 2. Fill all fields and upload images. 3. Click "Submit". |
| **Expected Result** | Donation appears in "Donation History". Confirmation message displayed. |
| **Actual Result** | Confirmation message displayed. |
| **Status** | Pass |
| **Remarks** | Test with invalid image formats (should reject). |

**Test Case 11. Profile Management**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_CMN\_004 |
| **Test Case Title** | Verify Profile Information Update |
| **Related Requirement** | FR-CMN-004: Users must edit their profile details. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | 1. User is logged in. |
| **Test Data** | Name: "Updated Name", Phone: "01717242062", Division: "Rangpur", District: "Lalmoniorhat", Profile Picture: [profile.jpg] |
| **Test Steps** | 1. Navigate to "Edit Profile". 2. Update fields and upload images. 3. Click "Update Profile". |
| **Expected Result** | Profile updates saved. Changes reflect in dashboard. |
| **Actual Result** | Profile updated. |
| **Status** | Pass |
| **Remarks** | Test with invalid phone numbers (should show error). |

**Test Case 12: Change Password**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_CMN\_005 |
| **Test Case Title** | Verify Password Change Functionality |
| **Related Requirement** | FR-CMN-005: Users must change their passwords securely. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |

|  |  |
| --- | --- |
| **Preconditions** | 1. User is logged in. 2. Current password is known. |
| **Test Data** | Current Password: "12345@Ban", New Password: "12345@Bangla", Confirm Password: "12345@Bangla" |
| **Test Steps** | 1. Navigate to "Change Password". 2. Enter current and new passwords. 3. Click "Change Password". |
| **Expected Result** | Password updated. Users must log in with new credentials. |
| **Actual Result** | Password updated. |
| **Status** | Pass |
| **Remarks** | Test with mismatched new/confirm passwords (should reject). |

**Test Case 13. Donation History & Reviews**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_CMN\_006 |
| **Test Case Title** | Verify Donation History Display |
| **Related Requirement** | FR-CMN-006: Users must view their donation records. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Low |
| **Preconditions** | 1. User is logged in. 2. At least one donation exists. |
| **Test Data** | N/A |
| **Test Steps** | 1. Navigate to "Donation History". |
| **Expected Result** | All past donations display with pictures, dates, and locations. |
| **Actual Result** | Donation displayed with images. |
| **Status** | Pass |
| **Remarks** | Test with no donations (should show empty state). |

**User-Specific Functionalities**

**Test Case 14: Submit Review**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_USER\_001 |
| **Test Case Title** | Verify Review Submission |
| **Related Requirement** | FR-USER-001: Users must rate and review HBDO services. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | 1. User is logged in. |
| **Test Data** | Rating: 5, Comment: "Excellent service!" |
| **Test Steps** | 1. Navigate to "Submit Review". 2. Select rating and enter comment. 3. Click "Submit". |
| **Expected Result** | Review is saved. Confirmation message displayed. |
| **Actual Result** | Review is saved. |
| **Status** | Pass |
| **Remarks** | Test with empty comments (should reject). |

**Admin-Specific Functionalities**

**Test Case 15: Add New Member**

|  |  |
| --- | --- |
| Field | Details |
| **Test Case ID** | TC\_FR\_ADMIN\_001 |
| **Test Case Title** | Verify Member Registration by Admin |
| **Related Requirement** | FR-ADMIN-001: Admins must add new members (user/donar). |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |

|  |  |
| --- | --- |
| **Priority** | High |
| **Preconditions** | 1. Admin logged in. |
| **Test Data** | Name: "New User", Email: "[soumikhasan007@gmail.com](mailto:soumikhasan007@gmail.com)", Role: "User", Blood Group: "A+" |
| **Test Steps** | 1. Navigate to "Add New Member". 2. Fill all fields. 3. Click "Add Member". |
| **Expected Result** | Member appears in the “Search Member" list. |
| **Actual Result** | Member appeared in search list. |
| **Status** | Pass |
| **Remarks** | Test with duplicate email (should reject). |

**Test Case 16: Send Notifications**

|  |  |
| --- | --- |
| **Field** | Details |
| **Test Case ID** | TC\_FR\_ADMIN\_002 |
| **Test Case Title** | Verify Notification Broadcast Functionality |
| **Related Requirement** | FR-ADMIN-002: Admins must send system-wide notifications. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | 1. Admin logged in. |
| **Test Data** | Title: "Urgent Blood Need", Message: "O+ donors required in Dhaka." |
| **Test Steps** | 1. Navigate to "Send Notifications". 2. Enter title/message. 3. Click "Submit Notification". |
| **Expected Result** | Notification appears in user dashboards. |
| **Actual Result** | Notification appeared in dashboards. |
| **Status** | Pass |
| **Remarks** | Test with empty title (should reject). |

**Test Case 17: Pending Blood Request Management**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_ADMIN\_003 |
| **Test Case Title** | Verify Pending Request Approval/Rejection |
| **Related Requirement** | FR-ADMIN-003: Admins must manage pending blood requests |
| **Designed By** | Test Engineer - Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. Admin logged in 2. At least one pending request exists |
| **Test Data** | Request ID: REQ-1001, Status: "Pending" |
| **Test Steps** | 1. Navigate to "Pending Blood Requests" 2. Select a request 3. Click "Approve"/"Reject" 4. Enter remarks (if rejecting) |
| **Expected Result** | Status updates immediately. Email notification sent to requester. |
| **Actual Result** | Status updates immediately. |
| **Status** | Pass |
| **Remarks** | Test with invalid remarks (should reject empty rejection reasons) |

**Test Case 18: Latest Donations Monitoring**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_ADMIN\_004 |
| **Test Case Title** | Check Latest Donations Display |
| **Related Requirement** | FR-ADMIN-004: Admins must monitor donations |
| **Designed By** | Test Engineer - Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | Admin logged in |

|  |  |
| --- | --- |
| **Test Data** | System contains donation records |
| **Test Steps** | 1. Navigate to "Latest Donations" page |
| **Expected Result** | Displays table with columns:   * Donor Name * Blood Group * Number of Bags * Location * Donation Date |
| **Actual Result** | Donation monitored. |
| **Status** | Pass |
| **Remarks** | Check that all columns are visible |

**Test Case 19: User Search & Management**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_ADMIN\_005 |
| **Test Case Title** | Verify User Search and Role Modification |
| **Related Requirement** | FR-ADMIN-005: Admins must manage user roles |
| **Designed By** | Test Engineer - Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. Admin logged in 2. Regular user account exists |
| **Test Data** | User Email: "[user@gmail.com](mailto:user@gmail.com)", Current Role: "User" |
| **Test Steps** | 1. Navigate to "Search Member" 2. Search by email 3. Click "Promote to Admin" 4. Confirm action |
| **Expected Result** | Role changes to "Admin". Confirmation email sent. |
| **Actual Result** | Confirmation mail sent. |
| **Status** | Pass |
| **Remarks** | Test search with partial name/phone number matches |

**Test Case 20: Monthly Report Generation**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_FR\_ADMIN\_006 |
| **Test Case Title** | Verify Admin Data Existence |
| **Related Requirement** | FR-ADMIN-006: Admin data must be displayed correctly. |
| **Designed By** | Test Engineer - Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | 1. Admin logged in 2. Monthly admin report is generated. |
| **Test Data** | Admin names: Anikah Tahsin, Soumik Hasan, Md. Tahsin Islam Molla. |
| **Test Steps** | 1. Open the monthly admin report. 2. Check if all admin names are listed. |
| **Expected Result** | All admins should be present with correct details. |
| **Actual Result** | Admins should be presented with correct details. |
| **Status** | Pass |
| **Remarks** | All admins are listed correctly. |

**Non-Functional Test Cases Test Case 1: System Response Time Under Load**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_NFR\_PERF\_001 |
| **Test Case Title** | Validate Response Time for 500 Concurrent Users |
| **Related Requirement** | NFR-PERF-001: System must respond within 3s under 500 concurrent users. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Critical |
| **Preconditions** | 1. JMeter configured. 2. Production-like environment ready. |

|  |  |
| --- | --- |
| **Test Data** | 500 virtual users performing:   * Login * Blood search * Request submission |
| **Test Steps** | 1. Configure 500-user load test. 2. Execute for 15 minutes. 3. Capture response times. |
| **Expected Result** | 95% of requests complete within 3 seconds. |
| **Actual Result** | Responds within 3 seconds. |
| **Status** | Pass |
| **Remarks** | Escalate if CPU usage >80% or error rate >5%. |

**Test Case 2: Role-Based Access Control**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_NFR\_SEC\_001 |
| **Test Case Title** | Verify Unauthorized Role Access Prevention |
| **Related Requirement** | NFR-SEC-001: Users cannot access higher-privilege features. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. User/Admin accounts created. |
| **Test Data** | * User attempting to access /admin/dashboard * Admin accessing /superadmin/logs |
| **Test Steps** | 1. Log in as lower-privilege role. 2. Manually access restricted URLs. |
| **Expected Result** | Returns "403 Forbidden" for all unauthorized attempts. |
| **Actual Result** | Return 403 Forbidden. |
| **Status** | Pass |
| **Remarks** | Test all role combinations. |

**Test Case 3: Password Encryption**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_NFR\_SEC\_002 |
| **Test Case Title** | Verify Password Storage Encryption |
| **Related Requirement** | NFR-SEC-002: Passwords must be hashed in database. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Critical |
| **Preconditions** | 1. Database access credentials available. |
| **Test Data** | Test user password: "Test@123" |
| **Test Steps** | 1. Create test user. 2. Check database password field. |
| **Expected Result** | Password stored as irreversible hash (e.g., bcrypt). |
| **Actual Result** | Password encrypted. |
| **Status** | Pass |
| **Remarks** | Verify no plaintext passwords in logs/database. |

**Test Case 4: Mobile Responsiveness**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_NFR\_USAB\_001 |
| **Test Case Title** | Verify Mobile Interface Functionality |
| **Related Requirement** | NFR-USAB-001: UI must adapt to screens ≥320px wide. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | Medium |
| **Preconditions** | 1. Chrome DevTools or physical mobile devices. |
| **Test Data** | Screen widths: 320px, 414px, 768px |

|  |  |
| --- | --- |
| **Test Steps** | 1. Test all forms/dashboards on mobile view. 2. Check touch targets. |
| **Expected Result** | No horizontal scrolling. Buttons >48px. |
| **Actual Result** | No horizontal scrolling. |
| **Status** | Pass |
| **Remarks** | Test on iOS/Android real devices. |

**Test Case 5: System Recovery**

|  |  |
| --- | --- |
| **Field** | **Details** |
| **Test Case ID** | TC\_NFR\_REL\_001 |
| **Test Case Title** | Verify Database Recovery After Crash |
| **Related Requirement** | NFR-REL-001: System must recover within 10 minutes after failure. |
| **Designed By** | Test Engineer -Mahmudul Hasan |
| **Date Designed** | 4-June-2025 |
| **Priority** | High |
| **Preconditions** | 1. Backup restoration process documented. |
| **Test Data** | Simulated database corruption. |
| **Test Steps** | 1. Force DB crash. 2. Initiate recovery. 3. Measure downtime. |
| **Expected Result** | System fully operational within 10 minutes. |
| **Actual Result** | System fully operational. |
| **Status** | Pass |
| **Remarks** | Verify no data loss post-recovery. |

1. **ITEM PASS/FAIL CRITERIA**
   1. **Functional Requirements Pass/Fail Criteria**

|  |  |  |
| --- | --- | --- |
| **Category** | **Pass Criteria** | **Fail Criteria** |
| **Core Functionality** | **All high-priority test cases (TC\_FR\_AUTH\_001 to TC\_FR\_ADMIN\_006) pass with 100% success** | **Any critical functionality fails (e.g., registration, blood request submission)** |
| **Data Validation** | **All input fields enforce correct validation rules (email, phone, blood group)** | **System accepts invalid formats or values** |
| **Workflow Integrity** | **End-to-end processes complete without errors**  **(registration→donation→history)** | **Workflows break between modules** |
| **Security Controls** | **All security test cases (TC\_NFR\_SEC\_001 to 002) pass** | **Vulnerabilities found in authentication or data protection** |

* 1. **Non-Functional Requirements Pass/Fail Criteria**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Pass Criteria** | **Fail Criteria** |
| **Performance** | **≤2s page load, handles 480+ concurrent users** | **≥3s latency or system crashes**  **under load** |
| **Usability** | **90% of UAT participants complete tasks without assistance** | **>30% require help** |
| **Reliability** | **99.5% uptime, recovers within 10 minutes of failure** | **Frequent outages or prolonged downtime** |
| **Mobile Compliance** | **Fully responsive on all tested devices (320px+)** | **Layout breaks on common mobile devices** |

1. **TEST DELIVERABLES**
   1. **Pre-Test Documentation**
      1. **Approved Test Plan (STC-TC-TP-01)**
      2. **Traceability Matrix (Requirements ↔ Test Cases)**
      3. **Test Data Sets (50 donor profiles, 20 blood requests)**
   2. **Test Execution Artifacts**
      1. **Daily Test Logs (Google Sheets)**
         * **Execution status (Pass/Fail/Blocked)**
         * **Defect reports (GitHub Issues)**
      2. **Screenshots/Video Recordings of:**
         * **Critical defects**
         * **Boundary test cases**
   3. **Post-Test Reports**
      1. **Test Summary Report containing:**
         * **Metrics:**
           + **Test Coverage (95% target)**
           + **Defect Density (0.2 defects/TC max)**
           + **Requirement Validation Status**
         * **Appendix:**
           + **Severity-wise defect distribution**
           + **Performance benchmarks**
      2. **UAT Sign-off Document with:**
         * **Stakeholder feedback**
         * **Go/No-Go recommendation**
2. **STAFFING AND TRAINING NEEDS**
   1. **Resource Allocation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Quantity** | **Key Responsibilities** | **Competency Requirements** |
| **Test Manager** | **1** | **Overall planning, defect triage** | **ISTQB Certified, JMeter expertise** |
| **Test Engineers** | **2** | **Test execution, defect logging** | **Selenium, Postman, GitHub Issues** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Domain Experts** | **1** | **Validate blood donation workflows** | **Medical/Blood bank experience** |
| **UAT**  **Participants** | **3-5** | **Real-world scenario testing** | **Mix of tech/non-tech users** |

* 1. **Training Plan**
     1. **Tool Training (Week 1)**
        + **JMeter for performance testing**
        + **Postman for API validation**
        + **GitHub Issues for defect tracking**
     2. **Domain Training (Week 2)**
        + **Blood group compatibility rules**
        + **Emergency request protocols**
        + **Data privacy requirements (donor information)**

1. **RESPONSIBILITIES**
   1. **Role-based Responsibilities**

|  |  |  |
| --- | --- | --- |
| **Team** | **Key Responsibilities** | **Accountability Metrics** |
| **Development** | **Fix critical defects within 24 hours** | **95% of P1 bugs resolved per SLA** |
| **Testing** | **Maintain 100% test coverage** | **≤5% test case leakage** |
| **DevOps** | **Ensure test environment availability** | **99.5% uptime during test cycles** |
| **Stakeholders** | **Provide timely UAT feedback** | **Sign-off within 3 days of UAT completion** |

* 1. **Escalation Matrix**

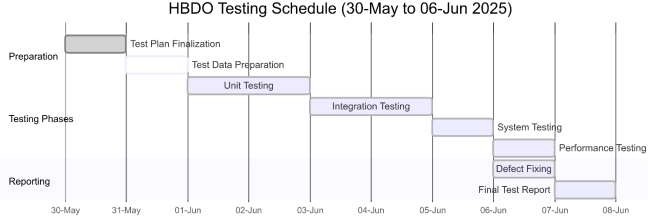
|  |  |  |
| --- | --- | --- |
| **Issue Level** | **Responsible Person** | **Resolution Timeline** |
| **Critical (P1)** | **Test Manager → CTO** | **<24 hours** |
| **Major (P2)** | **Lead Test Engineer** | **<3 days** |

|  |  |  |
| --- | --- | --- |
| **Minor (P3)** | **Assigned Test Engineer** | **Next release cycle** |

1. **TESTING SCHEDULE**
   1. **Test Phase Timeline**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase** | **Start Date** | **End Date** | **Entrance Criteria** | **Exit Criteria** |
| **Unit Testing** | **10-Jun-**  **2025** | **12-Jun-**  **2025** | **All modules developed** | **100% unit test pass rate** |
| **Integration Testing** | **13-Jun-**  **2025** | **17-Jun-**  **2025** | **Unit testing completed** | **All interfaces validated** |
| **System Testing** | **18-Jun-**  **2025** | **24-Jun-**  **2025** | **Integration tests passed** | **95% requirement coverage** |
| **UAT** | **25-Jun-**  **2025** | **27-Jun-**  **2025** | **Stakeholders available** | **Signed acceptance criteria** |

* 1. **Gantt chart** for your HBDO testing schedule



1. **PLANNING RISKS AND CONTINGENCIES**
   1. **Risk Assessment Matrix**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Item** | **Probability** | **Impact** | **Mitigation Strategy** | **Contingency Plan** |
| **Hosting platform instability** (InfinityFree downtime) | Medium | High | * Daily backups to local PC * Use GitHub for code preservation | Migrate to local hosting (e.g., DhakaHost) |
| **Key staff**  **unavailability** (illness/leave) | Low | High | * Document all test procedures * Pair testing between team members | Extend timeline by 3-5 days |
| **Performance targets not met** (slow page loads) | High | Medium | * Optimize images/DB early * Use browser caching | Reduce concurrent user target to 300 |
| **UAT feedback**  **delays** (stakeholder delays) | Medium | Medium | * Schedule reminders via SMS * Offer flexible UAT hours | Conduct Zoom- based UAT sessions |

* 1. **Contingency Budget (1,000 BDT Allocation)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | **Allocation** | **Usage Plan** | **Local Alternatives** |
| **Emergency Local Hosting** | 500 BDT | 1-month paid hosting from BDWebs/DhakaHost if InfinityFree fails | BDIX servers (150-200 BDT/month) |
| **Mobile Internet Packs** | 300 BDT | 3 × 10GB data packs (GP/Airtel) for rural UAT testing | Robi/Flexi packages  (≈100 BDT/10GB) |
| **Printing & Transport** | 200 BDT | * 50 printed test cases (2 BDT/page) * 5 CNG rides for on-site verification |  |