# HealthConnect Test Plan Document

**Version:** 1.0
**Date:** February 5, 2026
**Author:** Sarath Sasidharan
**Status:** Approved

---

## Table of Contents



---

## 1. Document Control & Version History

### Document Information

| Field | Value |
|---|---|
| **Document Title** | HealthConnect Test Plan |
| **Project Name** | HealthConnect Telemedicine Platform |
| **Version** | 1.0 |
| **Date** | February 5, 2026 |
| **Author** | Sarath Sasidharan |
| **Reviewers** | QA Lead, Product Owner |
| **Status** | Approved |

### Revision History

| Version | Date | Author | Changes | Approved By |
|---|---|---|---|---|
| 1.0 | Feb 5, 2026 | Sarath Sasidharan | Initial version | QA Lead |

---

## 2. Executive Summary

### 2.1 Project Overview

HealthConnect is a comprehensive telemedicine platform that enables patients to connect with healthcare providers, book appointments, manage medical records, and process payments online. The platform serves two primary user types: **Patients** and **Doctors**, each with distinct functionalities and access levels.

**Application URL:** http://18.142.250.249:5000/

### 2.2 Testing Objectives

The primary objectives of this testing effort are:

1. **Functional Validation:** Ensure all features work as per requirements
2. **Quality Assurance:** Identify and report defects before production release
3. **User Experience:** Validate usability and user interface consistency
4. **Security:** Verify authentication, authorization, and data protection
5. **Performance:** Ensure acceptable response times and system stability
6. **Compliance:** Validate data handling and privacy requirements

### 2.3 Scope Summary

**In-Scope:**

- User Management (Registration, Login, Profile)
- Doctor Discovery (Search, Filter, Profile View)
- Appointment Management (Book, View, Cancel)
- Payment Processing (Multiple payment methods)
- Medical Records Management (Upload, View, Delete)
- Dashboard & Reporting (Patient/Doctor dashboards)

**Out-of-Scope:**

- Database testing (no direct database access)
- API testing (if APIs not accessible)
- Load/Stress testing (limited to response time validation)
- Third-party payment gateway backend testing
- Mobile app testing (web application only)
- Email notification testing (if not testable)

### 2.4 Key Stakeholders

| Role | Name/Department | Responsibility |
|---|---|---|
| QA Engineer | Sarath Sasidharan | Test design, execution, automation |
| QA Lead | TBD | Test plan approval, review |
| Product Owner | TBD | Requirements clarification, UAT |
| Development Team | TBD | Bug fixes, build deployment |
| Project Manager | TBD | Project coordination |

---

## 3. Test Strategy & Approach

## 3.1 Testing Methodology

This project follows an **Agile testing methodology** with iterative test cycles. Testing activities are integrated into the development lifecycle, allowing for continuous feedback and early defect detection.

## 3.2 Test Levels

### 3.2.1 Unit Testing

- **Responsibility:** Development Team
- **Coverage:** Individual components and functions
- **Tools:** Jest, Mocha (as applicable)

### 3.2.2 Integration Testing

- **Responsibility:** QA Team
- **Coverage:** Module interactions, API integrations
- **Approach:** Bottom-up integration testing

### 3.2.3 System Testing

- **Responsibility:** QA Team
- **Coverage:** End-to-end functionality, user workflows
- **Approach:** Functional, regression, security, performance testing

### 3.2.4 User Acceptance Testing (UAT)

- **Responsibility:** Product Owner, Business Users
- **Coverage:** Business scenarios, user workflows
- **Approach:** Real-world scenarios, user stories

## 3.3 Test Types

### 3.3.1 Functional Testing

- **Purpose:** Validate functional requirements
- **Coverage:** All features and user workflows
- **Priority:** High

### 3.3.2 Regression Testing

- **Purpose:** Ensure existing functionality remains intact
- **Coverage:** 130 test cases covering all modules
- **Frequency:** After each build/release

### 3.3.3 Security Testing

- **Purpose:** Validate authentication, authorization, data protection
- **Coverage:** Login security, SQL injection, XSS prevention
- **Priority:** High

### 3.3.4 Performance Testing

- **Purpose:** Validate response times and system stability
- **Coverage:** Page load times, transaction processing times
- **Target:** < 3 seconds page load, < 5 seconds payment processing

### 3.3.5 Usability Testing

- **Purpose:** Validate user interface and user experience
- **Coverage:** Navigation, form layouts, error messages
- **Approach:** Manual testing with real user scenarios

### 3.4 Test Techniques

#### 3.4.1 Equivalence Partitioning

- **Application:** Input field validation (email, password, phone)
- **Example:** Valid email formats, invalid email formats, boundary values

#### 3.4.2 Boundary Value Analysis

- **Application:** Password length (minimum 6 characters), date ranges
- **Example:** Password with 5, 6, 7 characters; dates at boundaries

#### 3.4.3 Decision Table Testing

- **Application:** Complex business rules (appointment booking, payment processing)
- **Example:** Appointment booking with different date/time combinations

#### 3.4.4 State Transition Testing

- **Application:** Appointment status transitions (Scheduled → Completed/Cancelled)
- **Example:** Valid and invalid state transitions

#### 3.4.5 Error Guessing

- **Application:** Negative testing scenarios based on experience
- **Example:** SQL injection, XSS attacks, invalid data formats

### 3.5 AI Tool Utilization Strategy

#### 3.5.1 Cursor IDE

- **Purpose:** Test automation framework development
- **Usage:** Code generation, refactoring, debugging
- **Benefits:** Faster development, code consistency

#### 3.5.2 ChatGPT/Claude

- **Purpose:** Test case design, test data generation
- **Usage:** Scenario brainstorming, edge case identification
- **Benefits:** Comprehensive coverage, creative test scenarios

### 3.6 Defect Management Approach

- **Tool:** Manual tracking (Excel/Spreadsheet) or Jira (if available)
- **Severity Levels:** Critical (P0), High (P1), Medium (P2), Low (P3)
- **Priority:** Based on business impact
- **Lifecycle:** New → Assigned → In Progress → Fixed → Retest → Closed
- **Reporting:** Daily defect reports, weekly summary reports

---

## 4. Scope

### 4.1 In-Scope Features

#### 4.1.1 User Management

- **Patient Registration:** All fields validation, role selection
- **Doctor Registration:** Patient fields + doctor-specific fields (specialty, license, qualification, experience, fee, bio)
- **Login:** Patient and doctor login with credential validation
- **Profile Management:** View and update user profile information

- **Logout:** Session termination

#### 4.1.2 Doctor Discovery

- **Doctor Search:** Search by name or specialty
- **Filtering:** Filter by specialty, city
- **Doctor Profile:** View doctor details, availability, reviews
- **Doctor Listing:** Display doctor cards with key information

#### 4.1.3 Appointment Management

- **Book Appointment:** Select date, reason, additional notes
- **View Appointments:** List all appointments with filters (All, Scheduled, Completed, Cancelled)
- **Cancel Appointment:** Cancel scheduled appointments
- **Appointment Details:** View complete appointment information

#### 4.1.4 Payment Processing

- **Payment Methods:** Credit Card, Debit Card, Net Banking, UPI
- **Payment Form:** Card details, validation
- **Payment Confirmation:** Success/failure handling
- **Payment History:** View payment status for appointments

#### 4.1.5 Medical Records Management

- **Upload Records:** Upload medical documents with metadata
- **View Records:** List all records with filters (All, Lab Reports, Prescriptions, Scans)
- **Delete Records:** Remove medical records with confirmation
- **Record Details:** View record information and attached files

#### 4.1.6 Dashboard & Reporting

- **Patient Dashboard:** Widgets (Total Appointments, Upcoming Appointments, Medical Records), Quick Actions
- **Doctor Dashboard:** Widgets (Total Appointments, Today's Appointments, Total Patients, Average Rating), Quick Actions

### 4.2 Out-of-Scope Features

1. **Database Testing:** No direct database access for testing
2. **API Testing:** Backend API testing (if APIs not accessible)
3. **Load/Stress Testing:** Large-scale performance testing (limited to response time validation)
4. **Third-Party Integrations:** Payment gateway backend testing
5. **Mobile App Testing:** Only web application testing
6. **Email Notifications:** Email delivery testing (if not testable)
7. **Video Consultation:** Video call functionality (if not implemented)
8. **Prescription Management:** Doctor prescription writing (if not available)

---

## 5. Test Environment

### 5.1 Application Details

| Component | Details |
|---|---|
| **Application URL** | http://18.142.250.249:5000/ |
| **Environment Type** | Test/Staging |

| | |
|---|---|
| **Access** | Public (no VPN required) |

### 5.2 Browser Compatibility

| Browser | Version | OS | Priority |
|---|---|---|---|
| Google Chrome | Latest | Windows 10/11, macOS | High |
| Mozilla Firefox | Latest | Windows 10/11, macOS | High |
| Microsoft Edge | Latest | Windows 10/11 | Medium |
| Safari | Latest | macOS | Medium |

### 5.3 Operating Systems

- **Windows 10/11:** Primary testing environment
- **macOS:** Secondary testing environment

### 5.4 Test Tools

| Tool Category | Tool Name | Version | Purpose |
|---|---|---|---|
| **Automation Framework** | Playwright | 1.40.0 | Test automation |
| **Programming Language** | JavaScript (Node.js) | 16+ | Test script development |
| **Test Management** | Excel/CSV | - | Test case documentation |
| **Defect Tracking** | Excel/Manual | - | Bug tracking |
| **AI Tools** | Cursor IDE, ChatGPT/Claude | - | Code generation, test design |
| **Version Control** | Git | - | Code repository |

### 5.5 Test Data

#### 5.5.1 Demo Credentials

**Patient:**

- Email: john.doe@example.com
- Password: password123

**Doctor:**

- Email: dr.emily.carter@healthcare.com (Note: May not work)
- Password: password123

#### 5.5.2 Test Data Strategy

- **New Test Accounts:** Create new patient/doctor accounts for testing
- **Test Data Management:** Use JSON files for test data
- **Data Cleanup:** Manual cleanup of test data after test execution
- **Data Privacy:** Ensure no real patient data is used

---

## 6. Risk Analysis

### 6.1 Risk Matrix

| Risk ID | Risk Description | Impact | Probability | Severity | Mitigation Strategy |
|---|---|---|---|---|---|
| R001 | Payment processing failures | High | Medium | High | Comprehensive payment testing, test with multiple payment methods |
| R002 | Unauthorized access to patient data | High | Low | High | Security testing, authentication/authorization validation |
| R003 | Appointment booking conflicts/double booking | High | Medium | High | Business logic validation, concurrent booking tests |
| R004 | Data loss in medical records | High | Low | High | Data integrity validation, backup verification |
| R005 | Search functionality returns incorrect results | Medium | Medium | Medium | Search validation, filter testing |
| R006 | Email/password validation bypass | Medium | Low | Medium | Input validation testing, security testing |
| R007 | Session management issues | Medium | Medium | Medium | Session timeout testing, concurrent session validation |
| R008 | UI rendering issues across browsers | Low | Medium | Low | Cross-browser testing, responsive design validation |
| R009 | Minor validation errors | Low | Low | Low | Comprehensive negative testing |

### 6.2 Risk Mitigation Strategies

1. **Comprehensive Test Coverage:** Ensure all critical paths are tested
2. **Security Testing:** Validate authentication, authorization, input validation
3. **Data Integrity Validation:** Verify data persistence and accuracy
4. **Regular Regression Testing:** Ensure existing functionality remains intact
5. **Early Testing:** Start testing as soon as features are available
6. **Defect Tracking:** Track and prioritize defects based on severity
7. **Communication:** Regular updates to stakeholders on testing progress

---

## 7. Test Schedule & Milestones

### 7.1 Test Timeline

| Phase | Start Date | End Date | Duration | Status |
|---|---|---|---|---|
| **Test Planning** | Jan 22, 2026 | Jan 23, 2026 | 2 days | Completed |
| **Test Design** | Jan 24, 2026 | Jan 28, 2026 | 5 days | Completed |
| **Test Execution** | Jan 29, 2026 | Feb 3, 2026 | 6 days | In Progress |
| **Defect Reporting & Retesting** | Feb 4, 2026 | Feb 5, 2026 | 2 days | Pending |
| **Test Closure** | Feb 6, 2026 | Feb 6, 2026 | 1 day | Pending |

### 7.2 Milestones

1. **M1 - Test Plan Approval:** Jan 23, 2026 ✓
2. **M2 - Test Cases Ready:** Jan 28, 2026 ✓
3. **M3 - Automation Framework Ready:** Feb 1, 2026 ✓
4. **M4 - Sanity Testing Complete:** Feb 2, 2026
5. **M5 - Regression Testing Complete:** Feb 3, 2026
6. **M6 - All Defects Fixed & Retested:** Feb 5, 2026
7. **M7 - Test Closure Report:** Feb 6, 2026

---

## 8. Entry & Exit Criteria

### 8.1 Entry Criteria

The following criteria must be met before test execution begins:

1. ✅ **Test Environment Available:** Application is accessible at http://18.142.250.249:5000/
2. ✅ **Application Build Deployed:** Latest build is deployed and accessible
3. ✅ **Test Data Prepared:** Demo credentials and test data are available
4. ✅ **Test Cases Reviewed and Approved:** All test cases are reviewed and approved
5. ✅ **Required Tools Set Up:** Playwright, Node.js, and other tools are installed
6. ✅ **Test Plan Approved:** Test plan document is reviewed and approved

### 8.2 Exit Criteria

The following criteria must be met before test closure:

1. **All Planned Test Cases Executed:** 100% of planned test cases are executed
2. **95% Pass Rate Achieved:** At least 95% of test cases pass
3. **No Critical (P0) Defects Open:** All critical defects are fixed and retested
4. **No High (P1) Defects Open:** All high-priority defects are fixed or approved for next release
5. **Test Summary Report Completed:** Final test summary report is prepared
6. **All Deliverables Submitted:** Test plan, test cases, automation scripts, and reports are submitted

---

## 9. Roles & Responsibilities

### 9.1 QA Engineer (Sarath Sasidharan)

**Responsibilities:**

- Test case design and documentation
- Test execution (manual and automated)

- Test automation framework development
- Defect identification and reporting
- Test data preparation
- Test execution reports

### 9.2 QA Lead

**Responsibilities:**

- Test plan review and approval
- Test strategy guidance
- Test case review
- Defect triage and prioritization
- Quality metrics tracking
- Stakeholder communication

### 9.3 Development Team

**Responsibilities:**

- Bug fixes and code changes
- Build deployment
- Technical clarifications
- Test environment maintenance

### 9.4 Product Owner

**Responsibilities:**

- Requirements clarification
- User Acceptance Testing (UAT)
- Business scenario validation
- Feature prioritization

---

## 10. Defect Management Process

### 10.1 Defect Lifecycle

```
New → Assigned → In Progress → Fixed → Retest → Closed
                      ↓
                  Rejected/Deferred
```

### 10.2 Severity Levels

| Severity | Description | Example | SLA |
|---|---|---|---|
| **P0 - Critical** | System crash, data loss, security breach | Payment failure, login not working | 24 hours |
| **P1 - High** | Major functionality broken, workaround available | Appointment booking fails, search not working | 48 hours |
| **P2 - Medium** | Minor functionality issue, workaround available | UI alignment issue, validation message unclear | 1 week |

| | | | |
|---|---|---|---|
| **P3 - Low** | Cosmetic issue, enhancement | Typo, color mismatch | Next release |

### 10.3 Defect Reporting Template

| Field | Description |
|---|---|
| **Defect ID** | Unique identifier |
| **Title** | Brief description |
| **Severity** | P0/P1/P2/P3 |
| **Priority** | High/Medium/Low |
| **Module** | Affected module |
| **Steps to Reproduce** | Detailed steps |
| **Expected Result** | What should happen |
| **Actual Result** | What actually happens |
| **Screenshots** | Attach if applicable |
| **Environment** | Browser, OS, version |

### 10.4 Defect Triage

- **Daily Triage:** Review new defects daily
- **Priority Assignment:** Assign priority based on business impact
- **Assignment:** Assign to development team
- **Tracking:** Track defect status and resolution

---

## 11. Test Deliverables

### 11.1 Test Documentation

1. **Test Plan Document** (This document)
    - Format: Word/PDF
    - Pages: 15-20
    - Status: ✅ Completed
2. **Sanity Test Suite**
    - Format: Excel/CSV
    - Test Cases: 35
    - Status: ✅ Completed
3. **Regression Test Suite**
    - Format: Excel/CSV
    - Test Cases: 130
    - Status: Pending

### 11.2 Test Automation

4. **Automation Framework**
    - Framework: Playwright + JavaScript
    - Pattern: Page Object Model (POM)
    - Test Cases: 35 sanity tests automated
    - Status: ✅ Completed
5. **Test Scripts**
    - Location: `tests/` directory
    - Coverage: All 35 sanity test cases
    - Status: ✅ Completed

### 11.3 Test Execution Reports

6. **Test Execution Reports**
    - Format: HTML, JSON, JUnit
    - Location: `reports/` directory
    - Status: Generated after execution
7. **Defect Reports**
    - Format: Excel/Spreadsheet
    - Status: Generated during execution

### 11.4 Test Summary

8. **Test Summary Report**
    - Format: Word/PDF
    - Content: Test metrics, defect summary, recommendations
    - Status: Pending (after execution)

---

## 12. Assumptions & Dependencies

### 12.1 Assumptions

1. **Stable Test Environment:** Test environment remains stable and accessible throughout testing
2. **Timely Bug Fixes:** Development team provides timely bug fixes for retesting
3. **No Major Changes:** No major requirement changes during test execution
4. **Test Data Availability:** Test data remains available and accessible
5. **Tool Availability:** All required tools and licenses are available

### 12.2 Dependencies

1. **Development Team:** Availability for bug fixes and clarifications
2. **Test Environment:** Stable and accessible test environment
3. **Test Data:** Access to demo credentials and ability to create test accounts
4. **Requirements:** Clear and documented requirements
5. **Build Deployment:** Regular builds deployed to test environment

### 12.3 Constraints

1. **Time Constraints:** Limited time for comprehensive testing

2. **Resource Constraints:** Single QA engineer for execution
3. **Environment Constraints:** No direct database or API access
4. **Tool Constraints:** Limited to available tools and licenses

---

## 13. Traceability Matrix

### 13.1 Requirements to Test Cases Mapping

| Requirement ID | Requirement Description | Test Case IDs | Status |
|---|---|---|---|
| REQ-001 | Patient Registration | SAN_PR_001 to SAN_PR_005, REG_PR_* | Covered |
| REQ-002 | Doctor Registration | SAN_DR_001 to SAN_DR_005, REG_DR_* | Covered |
| REQ-003 | Patient Login | SAN_LOGIN_001, SAN_LOGIN_003, SAN_LOGIN_004 | Covered |
| REQ-004 | Doctor Login | SAN_LOGIN_002 | Covered |
| REQ-005 | Doctor Search | SAN_DS_001 to SAN_DS_004, REG_DS_* | Covered |
| REQ-006 | Book Appointment | SAN_AB_001 to SAN_AB_006, REG_AB_* | Covered |
| REQ-007 | View Appointments | SAN_VA_001 to SAN_VA_003, REG_VA_* | Covered |
| REQ-008 | Payment Processing | SAN_PAY_001 to SAN_PAY_005, REG_PAY_* | Covered |
| REQ-009 | Medical Records | SAN_MR_001 to SAN_MR_003, REG_MR_* | Covered |
| REQ-010 | Patient Dashboard | REG_DASH_* | Covered |
| REQ-011 | Doctor Dashboard | REG_DASH_* | Covered |

### 13.2 Test Coverage Summary

| Module | Sanity Tests | Regression Tests | Total | Coverage |
|---|---|---|---|---|
| Patient Registration | 5 | 8 | 13 | 100% |
| Doctor Registration | 5 | 10 | 15 | 100% |
| Login | 4 | 6 | 10 | 100% |
| Doctor Search | 4 | 10 | 14 | 100% |
| Appointment Booking | 6 | 12 | 18 | 100% |
| View Appointments | 3 | 8 | 11 | 100% |
| Payment Processing | 5 | 10 | 15 | 100% |
| Medical Records | 3 | 9 | 12 | 100% |
| Dashboard | 0 | 8 | 8 | 100% |

| | | | | |
|---|---|---|---|---|
| **Total** | **35** | **81** | **116** | **100%** |

---

## Appendix A: Test Case Naming Convention

- **Sanity Tests:** `SAN_<MODULE>_<NUMBER>` (e.g., SAN_PR_001)
- **Regression Tests:** `REG_<MODULE>_<TYPE>_<NUMBER>` (e.g., REG_PR_NEG_001)

## Appendix B: Abbreviations

- **POM:** Page Object Model
- **UAT:** User Acceptance Testing
- **API:** Application Programming Interface
- **SQL:** Structured Query Language
- **XSS:** Cross-Site Scripting
- **CVV:** Card Verification Value
- **UPI:** Unified Payments Interface

---

**Document End**

---

*This test plan document follows IEEE 829 standard for software test documentation and industry best practices for QA testing.*