**CS673 Software Engineering** 

**Team 3 - MyMedic**

**Software Test Document**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Adriel Domingo | Requirements | *Adriel Domingo* | 5/26/2025 |
| [John Gutierrez](mailto:johng13@bu.edu) | All | [*John Gutierrez*](mailto:johng13@bu.edu) | 6/16/2025 |
| Mengliang Tan | QA | *Mengliang Tan* | 5/26/2025 |
| Uzay Isin Alici | Security | *Uzay Isin Alici* | 5/26/2025 |
| Tyler Gonsalves | Configuration/ Design | *Tyler Gonsalves* | 5/26/2025 |
| Indra Sigicharla | Team/Configuration | *Indra Sigicharla* | 6/9/2025 |
| Hongcheng Ding | Design | *Hongcheng Ding* | 5/26/2025 |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| 1.0 | Team 3 | 5/26/2025 | N/A |
| 2.0 | Team 3 | 6/16/2025 | Download Medical Records Feature Tests Added |

[Testing Summary](#_sm5odwyvuk3j)

[Manuel Tests Reports](#_pqso2mbjyzx4)

[Automated Testing Reports](#_mtfbusfb0eq3)

[Testing Metrics](#_rijyjeu2ojqa)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

# Testing Summary

## Unit Testing

Unit testing will be performed upon every check-in and merge to the main branch to ensure that the master branch is always in a runnable state. Unit testing will also be run on push to any branch. The results of these tests will be recorded in the artefacts of every release for tracking. Unit tests will be incrementally added as new features are developed.

## Integration Testing

Integration testing will be performed to verify API functionality between the frontend and backend, and between backend components. These tests will be performed upon merges to the main branch to validate system functionality on the main branch.

## System Testing

System testing will be performed manually prior to each release of the code. The test results will be included in the release notes for tracking.

## Acceptance Testing

Acceptance testing will be performed manually on each merge of the code to the main branch to validate added features. The test results will be included in the pull request for merging the feature branch to document feature functionality.

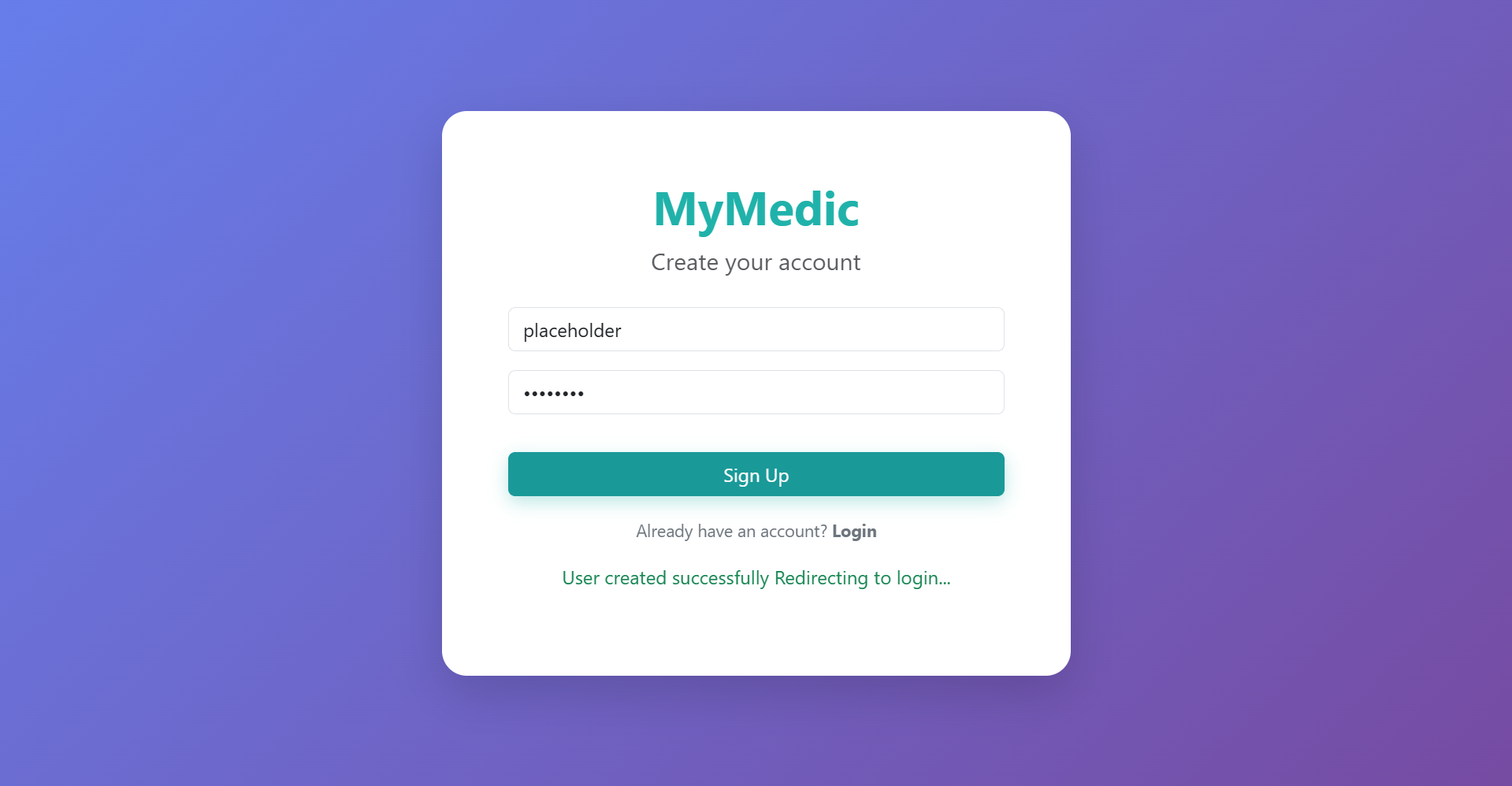
## Regression Testing

Regression testing will be performed on every push to every branch. As defects are added and fixed, tests to verify the bug is resolved will be added to the regression test suite. Regression test results will be recorded in pull requests and in releases of code.

# Manual Testing Report

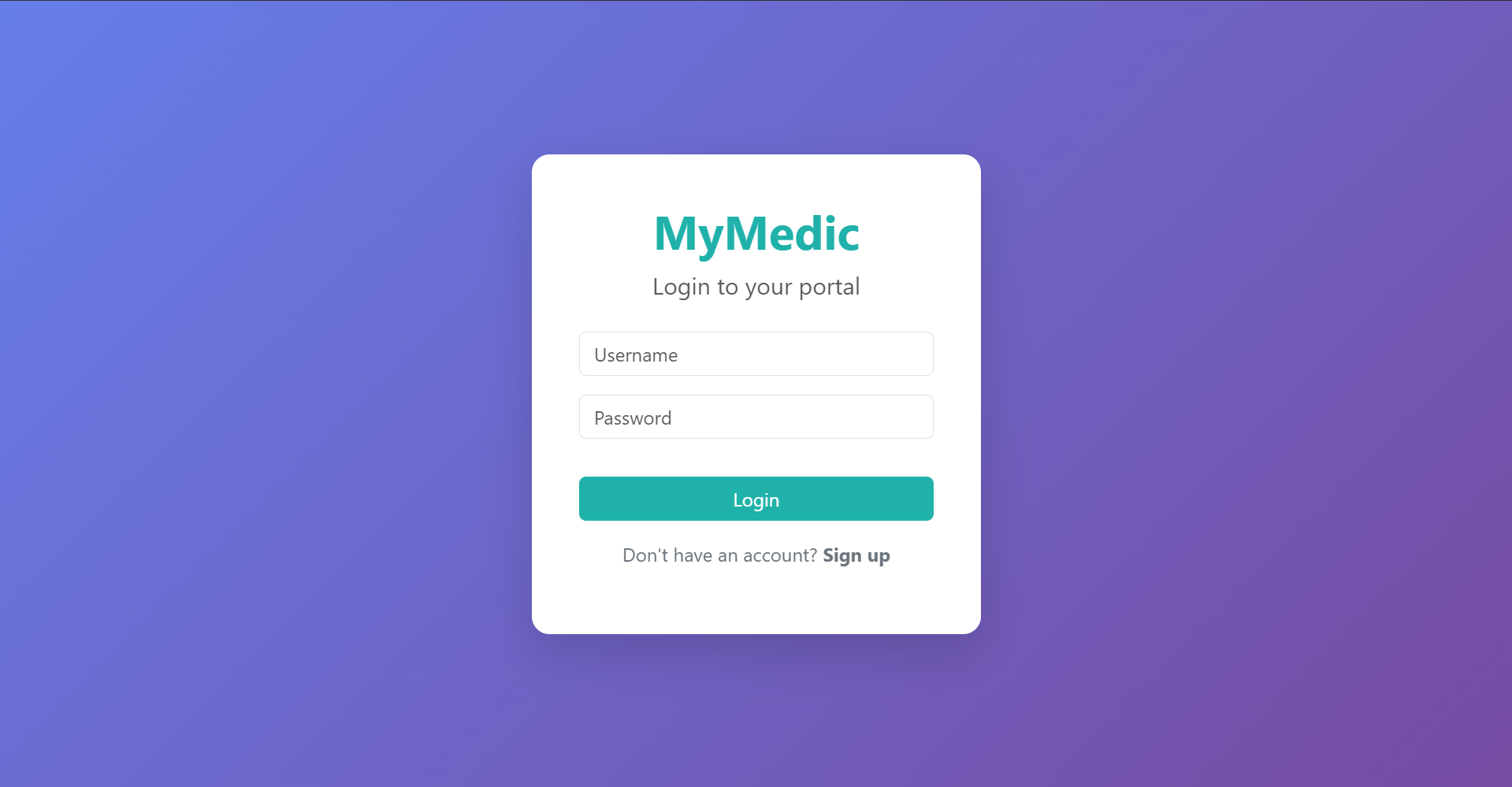
## Test Case 1: User Registration Updated by Indra Sigicharla

* New or old: New
* Test items: User Registration Page
* Test priority High
* Dependencies N/A
* Preconditions: Server is running
* input data:
  + Username: testuser
  + Password: 6uj9rD46!L#b
* Test steps:
  + Enter username in the username field of the registration form
  + Enter the password in the password field of the registration form
  + Click the “Sign Up” button
* Postconditions: Success message printed at the bottom of the form
* Expected output: User redirected to the login screen
* Actual output: User redirected to the login screen
* Pass or Fail: pass
* Bug id/link: <https://github.com/BUMETCS673/CS673OLSum25Team3/issues/11>
* Additional notes:



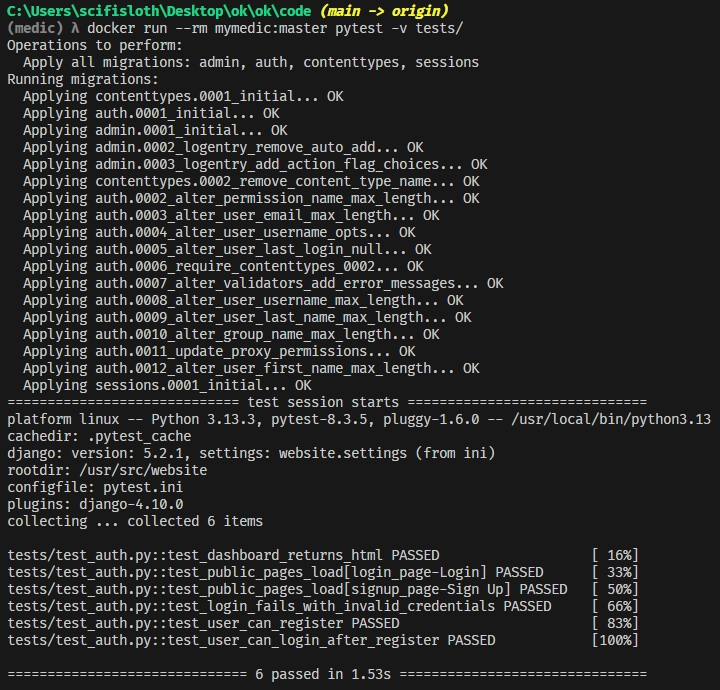
## Test Case 2: User Login

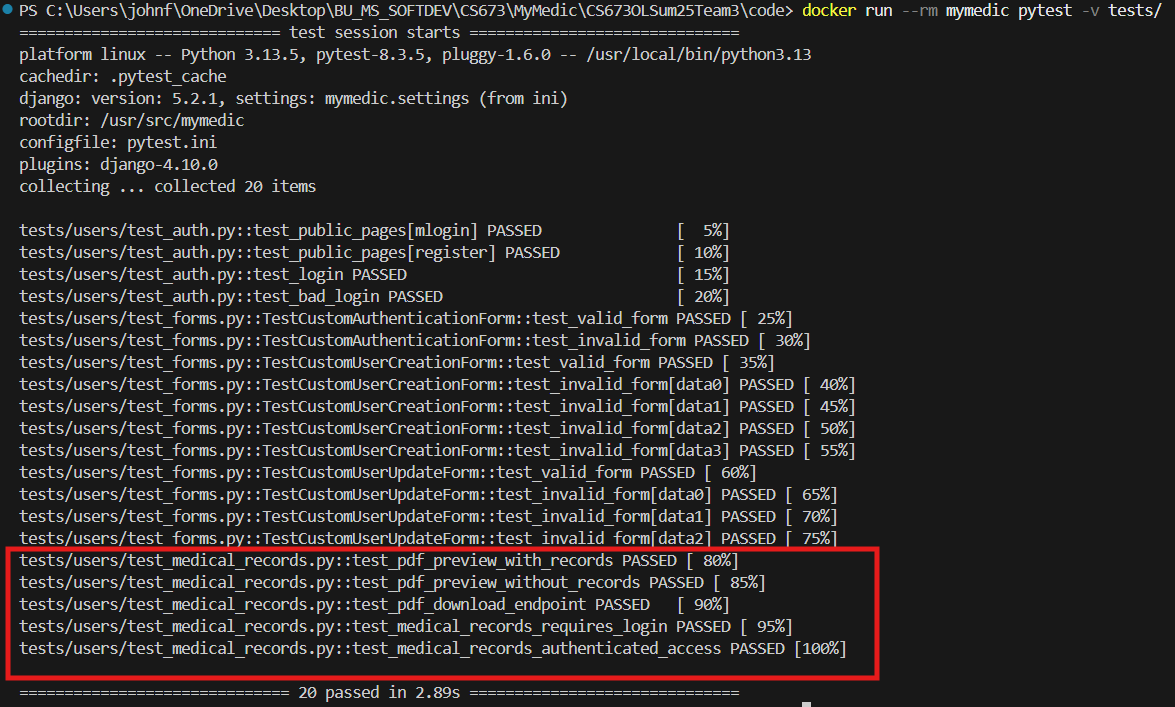
* New or old: New
* Test items: User Registration Page
* Test priority High
* Dependencies Success of Test Case 1
* Preconditions:
  + Server is running
  + User is registered
* input data:
  + Username: testuser
  + Password: 6uj9rD46!L#b
* Test steps:
  + Enter username in the username field of the registration form
  + Enter the password in the password field of the registration form
  + Click the “Login” button
* Postconditions: N/A
* Expected output: User redirected to the home page
* Actual output: User redirected to the home page
* Pass or Fail: pass
* Bug id/link: <https://github.com/BUMETCS673/CS673OLSum25Team3/issues/11>
* Additional notes:

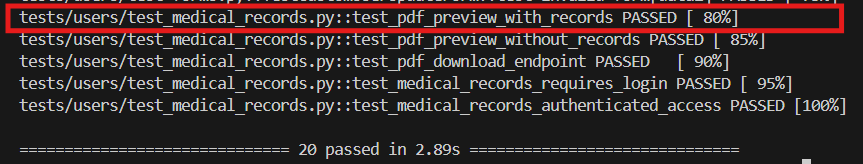


# Automated Testing Report

Automated unit and regression testing will be completed using pytest and jest for testing the python and javascript code, respectively. Cypress and Selenium will be used to complete integration, and system testing where appropriate. These tests will be orchestrated using github actions.



**Medical Records Automated Test Cases Updated by** [**John Gutierrez**](mailto:johng13@bu.edu)

**Test Case 1: Medical Records Access with Patient Data**

* New or old: New
* Test items: Medical Records Page Access for Authenticated User with Patient Data
* Test priority: High
* Dependencies:
  + User authentication system
  + Patient model
* Preconditions:
  + Server is running
  + User account exists
  + Patient data exists for user

# Input data:

# Username: testuser

# Password: testpass123

# Patient Data: John Doe, [test@example.com](mailto:test@example.com)

* Test steps:

1. Create user account with username 'testuser' and password 'testpass123'
2. Create associated patient record with first name 'John', last name 'Doe'
3. Authenticate user (force login)
4. Navigate to '/records/' endpoint
5. Verify response status and template rendering

* Postconditions: Medical records page displays successfully
* Expected output: HTTP 200 status code, 'users/medical\_records.html' template rendered
* Actual output: tests/users/test\_medical\_records.py::test\_pdf\_preview\_with\_records PASSED
* Pass or Fail: Pass
* Bug id/link: <https://github.com/BUMETCS673/CS673OLSum25Team3/issues/25>
* Additional notes: Tests core functionality of medical records access for valid authenticated users

## Test Case 2: Medical Records Access without Patient Data

## New or old: New

## Test items: Medical Records Page Access for User without Patient Data

## Test priority: Medium

## Dependencies: User authentication system

## Preconditions:

## Server is running

## User account exists without patient data

## Input data:

## Username: emptyuser

## Password: testpass123

## Patient Data: None

## Test steps:

# Create user account with username 'emptyuser' and password 'testpass123'

# Do not create associated patient record

# Authenticate user (force login)

# Navigate to '/records/' endpoint

# Verify response handles missing patient data appropriately

# Postconditions: System handles missing patient data gracefully

# Expected output: HTTP 200 or 404 status code (depending on business logic)

# Actual output: tests/users/test\_medical\_records.py::test\_pdf\_preview\_without\_records PASSED

# Pass or Fail: Passed

# Bug id/link:[https://github.com/BUMETCS673/CS673OLSum25Team3/issues/25](https://github.com/BUMETCS673/CS673OLSum25Team3/issues/25+)

# Additional notes: Tests edge case where user exists but patient data is missing

## Test Case 3: Medical Records PDF Download Functionality

## New or old: New

## Test items: PDF Download Endpoint for Medical Records

## Test priority: High

## Dependencies: User authentication system, Patient model, PDF generation

## Preconditions:

## Server is running

## User account exists

## Patient data exists

# Input data:

# Username: downloaduser

# Password: testpass123

# Patient Data: Jane Smith, download@example.com

# Test steps:

# Create user account with username 'downloaduser' and password 'testpass123'

# Create associated patient record with first name 'Jane', last name 'Smith'

# Authenticate user (force login)

# Navigate to '/records/' endpoint (placeholder for PDF download)

# Verify medical records page loads successfully

# Postconditions: Medical records page accessible for PDF functionality

# Expected output: HTTP 200 status code

# Actual output: tests/users/test\_medical\_records.py::test\_pdf\_download\_endpoint PASSED

# Pass or Fail: Passed

# Bug id/link: <https://github.com/BUMETCS673/CS673OLSum25Team3/issues/25>

# Additional notes: Currently tests page access; will be updated when PDF download functionality is implemented

## Test Case 4: Medical Records Authentication Requirement

## New or old: New

## Test items: Medical Records Page Authentication Enforcement

## Test priority: Critical

## Dependencies: User authentication system, login redirect functionality

## Preconditions: Server is running, no user logged in

## Input data:

## User session: Unauthenticated

## Target URL: '/records/'

# Test steps:

# Ensure no user is logged in (clear session)

# Attempt to navigate directly to '/records/' endpoint

# Verify system redirects to login page

# Check that redirect URL contains 'mlogin'

# Postconditions: Unauthenticated user redirected to login

# Expected output: HTTP 302 redirect status, redirect URL contains 'mlogin'

# Actual output: tests/users/test\_medical\_records.py::test\_medical\_records\_requires\_login PASSED

# Pass or Fail: Passed

# Bug id/link: <https://github.com/BUMETCS673/CS673OLSum25Team3/issues/25>

# Additional notes: Critical security test ensuring protected resources require authentication

## Test Case 5: Authenticated Medical Records Access

## New or old: New

## Test items: Medical Records Page Access for Authenticated User

## Test priority: High

## Dependencies: User authentication system, Patient model, template rendering

## Preconditions: Server is running, User account exists, Patient data exists

## Input data:

## Username: accessuser

## Password: testpass123

## Patient Data: Access User, access@example.com

# Test steps:

# Create user account with username 'accessuser' and password 'testpass123'

# Create associated patient record with first name 'Access', last name 'User'

# Authenticate user (force login)

# Navigate to '/records/' endpoint

# Verify successful page load and correct template rendering

# Postconditions: Medical records page displays with correct template

# Expected output: HTTP 200 status code, 'users/medical\_records.html' template rendered

# Actual output: tests/users/test\_medical\_records.py::test\_medical\_records\_authenticated\_access PASSED

# Pass or Fail: Pass

# Bug id/link: <https://github.com/BUMETCS673/CS673OLSum25Team3/issues/25>

# Additional notes: Comprehensive test of authenticated access with template verification

# Testing Metrics

In this section, you shall report any metrics used for the evaluation, e.g. # of test cases, test coverage, defects rate, etc.

| **Metric Name** | **Description: User Registration & Log In - by Indra Sigicharla** |
| --- | --- |
| **Test Cases** | 6 unique test cases executed as shown in the output from pytest -v tests/. |
| **Test Coverage** | Currently executing statement-level tests like user registration, login, dashboard access, and page loads. As development progresses, this can be expanded to path coverage by including tests that validate different conditional flows and edge cases in views, forms, or APIs. |
| **Test Case Pass Rate** | 100% pass rate (6/6 passed), which is well above the 80% threshold. |
| **Defect Rate** | No test failures or defects detected in the current suite. Assuming fewer than 20 known bugs per 1000 lines of code (KLOC), this passes the given quality threshold. |

| **Metric Name** | **Description: Medical Records Automated Test Cases - by** [**John Gutierrez**](mailto:johng13@bu.edu) |
| --- | --- |
| **Test Cases** | 5 unique test cases executed as shown in the output from pytest -v tests/.   1. test\_pdf\_preview\_with\_records - Tests medical records page access for authenticated users with patient data 2. test\_pdf\_preview\_without\_records - Tests medical records page behavior for users without patient data 3. test\_pdf\_download\_endpoint - Tests PDF download functionality (placeholder implementation) 4. test\_medical\_records\_requires\_login - Tests authentication requirement enforcement 5. test\_medical\_records\_authenticated\_access - Tests successful authenticated access with template verification |
| **Test Coverage** | Currently executing statement-level tests covering medical records page authentication, authorized access, patient data validation, and basic page rendering functionality. The test suite validates:   * Authentication enforcement for protected medical records endpoints * Proper handling of users with and without associated patient data * Template rendering verification for successful page loads * HTTP status code validation for various access scenarios |
| **Test case pass rate** | Pass rate must be greater 80%  **100% pass rate (5/5 passed)** - Exceeds the 80% threshold |
| **Defect rate** | 20 bugs or less per 1000 lines of code (KLOC)  **No test failures or defects detected** in the current suite. All medical records functionality tests are passing successfully |

# References

# Glossary