

# Med Scribe Professional Test Document

Florida Institute of Technology

Development Team:

Joshua Weil

Alijawaad Dewji

Mohammed Alzahrani

Sultan Almotairi

Faculty Sponsor:

Philip J. Bernhard     [pbernhar@cs.fit.edu](mailto:pbernhar@cs.fit.edu)

Team Contact:

[team@medscribepro.com](mailto:team@medscribepro.com)



**Test Document**

## Table of Contents

1	Doctor Testing.....	3
1.1	Doctor Registration.....	3
1.2	Doctor Log In.....	3
1.3	Doctor Front End.....	3
1.3.1	Practice Info (Doctor Front End).....	3
1.3.2	Access Medical Records (Doctor Front End).....	4
1.3.2.a	Attach Medical Records (Access Medical Records).....	4
1.3.3	Write Prescriptions (Doctor Front End).....	4
1.3.4	Communication (Doctor Front End).....	4
2	Pharmacist Testing.....	5
2.1	Pharmacist Registration.....	5
2.2	Pharmacist Log In.....	5
2.3	Pharmacist Front End.....	5
2.3.1	Pharmacy Info (Pharmacy Front End).....	5
2.3.2	Fill Prescriptions (Pharmacy Front End).....	6
2.3.2.a	Search for Patient (Fill Prescriptions).....	6
2.3.2.b	Change Prescription Status (Fill Prescriptions).....	6
2.3.2.c	Flag Bad Combinations (Fill Prescriptions).....	6
2.3.3	Communications (Pharmacy Front End).....	6
2.3.3.a	Attach Prescription (Communications).....	7
3	Patient Testing.....	7
3.1	Patient Registration.....	7
3.2	Patient Log In.....	7
3.3	Patient Front End.....	7
3.3.1	Personal Info (Patient Front End).....	7
3.3.2	Medical Records (Patient Front End).....	8
3.3.3	Prescriptions (Patient Front End).....	8
3.3.4	My Doctors (Patient Front End).....	8
3.3.5	My Pharmacy (Patient Front End).....	9

- 4 Administrator Testing ..... 9
  - 4.1 Admin Registration ..... 9
  - 4.2 Admin Log In ..... 9
  - 4.3 Admin Features ..... 9
  - 4.4 Admin Development ..... 9
  - 4.5 Admin Testing ..... 10

# 1 Doctor Testing

## 1.1 Doctor Registration

**Test Objective:** Verify that a user can register as a Doctor with Med Scribe Pro website given the proper criteria.

**Test Scenario:** Create fictitious doctor accounts via the Med Scribe Pro website registration process. Attempt to create accounts with correct and incorrect data to make sure input errors and incorrect data are handled properly.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table of doctors and the data used to create each followed by pass (created) or fail (not created) data.

## 1.2 Doctor Log In

**Test Objective:** Verify that any doctor can login to the Med Scribe Pro website.

**Test Scenario:** Using fictitious doctor accounts attempt to login to the Med Scribe Pro website.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table of doctor with pass/fail data on the test scenario.

## 1.3 Doctor Front End

**Test Objective:** Verify that doctors are brought to the Doctor Front End after logging in.

**Test Scenario:** Using fictitious doctor accounts login and record if the user was brought to the Doctor Front End page of the website.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table of doctors with pass/fail data on the test scenario.

### 1.3.1 Practice Info (Doctor Front End)

**Test Objective:** Verify a doctor can access the Practice Info Section. Verify that a doctor can update their personal and practice information. Verify a doctor can view other doctors that are members of the same Practice (Office).

**Test Scenario:** Using fictitious doctor accounts browse to the Practice Info Section, record the current information, attempt to make changes (record these as well). Record if changes were made after refresh.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table of doctors and original practice data, what data should change, and pass/fail data of each field on the test scenario.

### 1.3.2 Access Medical Records (Doctor Front End)

**Test Objective:** Verify a doctor can access the Medical Records Section. Verify that the doctor can view the medical records of patients that have given that doctor proper permissions.

**Test Scenario:** Using fictitious doctor accounts attempt to view patients medical records, and record whether viewing the record was successful.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table containing patient Name, permission given, and whether medical records were accessed (pass/fail).

#### 1.3.2.a Attach Medical Records (Access Medical Records)

**Test Objective:** Verify a doctor can add and attach medical documents to the patient's record.

**Test Scenario:** Using fictitious doctor accounts with correct and incorrect permissions attempt to attach a medical document to a fictitious patient record.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table containing patient Name, permission given, and whether medical records were attached (pass/fail).

### 1.3.3 Write Prescriptions (Doctor Front End)

**Test Objective:** Verify a doctor can access the Medical Records Section. Verify that doctors are able to write a digital prescription for the patient.

**Test Scenario:** Using fictitious doctor accounts with correct and incorrect permissions attempt to write a digital prescription for a patient. Log in as the patient and check if the prescription is viewable in the My Prescriptions Section.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table containing patient name, prescription that should be written, doctor who wrote the prescription, and pass/fail data stating if the prescription was viewable by the patient.

### 1.3.4 Communication (Doctor Front End)

**Test Objective:** Verify that doctors are able to communicate with other doctors, patients, and pharmacists via an integrated message system or Email.

**Test Scenario:** Using a fictitious doctor account choose a doctor, patient, and pharmacist, write a comment and then attempt to send it.

**Test Class:** Doctor Environment

**Qualification Method:** Inspection

**Record Method:** Table of sent doctor sent from, sent to, and received (pass/fail) data.

## 2 Pharmacist Testing

### 2.1 Pharmacist Registration

**Test Objective:** Verify that a user can register as a Pharmacist with Med Scribe Pro website given the proper criteria.

**Test Scenario:** Create fictitious pharmacist accounts via the Med Scribe Pro website registration process. Attempt to create accounts with correct and incorrect data to make sure input errors and incorrect data are handled properly.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of pharmacists and the data used to create each followed by pass (created) or fail (not created) data.

### 2.2 Pharmacist Log In

**Test Objective:** Verify that any pharmacist can login to the Med Scribe Pro website.

**Test Scenario:** Using fictitious pharmacist accounts attempt to login to the Med Scribe Pro website.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of pharmacists with pass/fail data on the test scenario.

### 2.3 Pharmacist Front End

**Test Objective:** Verify that pharmacists are brought to the Pharmacy Front End after logging in.

**Test Scenario:** Using fictitious pharmacist accounts login and record if the user was brought to the Pharmacy Front End page of the website.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of pharmacist with pass/fail data on the test scenario.

#### 2.3.1 Pharmacy Info (Pharmacy Front End)

**Test Objective:** Verify that pharmacists can access the Pharmacy Info Section and update their personal and pharmacy information. Verify that pharmacists are able to view other pharmacists that are members of the same Pharmacy.

**Test Scenario:** Using fictitious pharmacist accounts browse to the Pharmacy Info Section, record the current information, attempt to make changes (record these as well). Record if changes were made after refresh.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of doctors and original practice data, what data should change, and pass/fail data of each field on the test scenario.

### 2.3.2 Fill Prescriptions (Pharmacy Front End)

**Test Objective:** Verify that pharmacists can access the Fill Prescriptions Section.

**Test Scenario:** Using fictitious pharmacist accounts attempt to access the Fill Prescriptions Section.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table with pharmacist name and pass fail data.

#### 2.3.2.a Search for Patient (Fill Prescriptions)

**Test Objective:** Verify that pharmacists can search for patient's information by last name, date of birth, or patient MID.

**Test Scenario:** Using fictitious pharmacist browse to Fill Prescriptions and attempt to look up patients by name, date of birth, and Medical ID.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of patient searched for, search method used, results displayed, and pass/fail data.

#### 2.3.2.b Change Prescription Status (Fill Prescriptions)

**Test Objective:** Verify that the pharmacist can change the prescription status to "filled" and that the date is automatically attached.

**Test Scenario:** Using fictitious pharmacist browse to Fill Prescriptions look up a patient and fill a prescription. Refresh the page to view prescription status.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of patient, prescription to be filled, status after refresh, pass/fail data.

#### 2.3.2.c Flag Bad Combinations (Fill Prescriptions)

**Test Objective:** Verify that the system displays information about all medications the patient is using and flags medications that do not work together.

**Test Scenario:** Create two conflicting prescriptions for a fictitious patient. Using a fictitious pharmacist, browse to Fill Prescriptions look up the fictitious patient and check to see if the conflicting medications have been flagged.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of Medications, Number of conflicting medications, patient, and pass/fail data.

### 2.3.3 Communications (Pharmacy Front End)

**Test Objective:** Verify that the pharmacist can communicate with other pharmacists, patients, and doctors via integrated message system or Email.

**Test Scenario:** Using a fictitious pharmacist account choose a doctor, patient, and pharmacist, write a comment and then attempt to send it.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table of sent doctor sent from, sent to, and received (pass/fail) data.

### 2.3.3.a Attach Prescription (Communications)

**Test Objective:** Verify that Pharmacists can attach a prescription to their communication or comment.

**Test Scenario:** Using a fictitious pharmacist account choose a patient, write a comment, and then attempt to attach it to a prescription.

**Test Class:** Pharmacist Environment

**Qualification Method:** Inspection

**Record Method:** Table containing patient name, note, and linked prescriptions.

## 3 Patient Testing

### 3.1 Patient Registration

**Test Objective:** Verify that any user can register as a patient with Med Scribe Pro website given the proper criteria.

**Test Scenario:** Create fictitious patient accounts via the Med Scribe Pro website registration process. Attempt to create accounts with correct and incorrect data to make sure input errors and incorrect data are handled properly.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of patients and the data used to create each followed by pass (created) or fail (not created) data.

### 3.2 Patient Log In

**Test Objective:** Verify that any patient can login to the Med Scribe Pro website.

**Test Scenario:** Using fictitious patient accounts attempt to login to the Med Scribe Pro website.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of patients with pass/fail data on the test scenario.

### 3.3 Patient Front End

**Test Objective:** Verify that patients are brought to the Patient Front End after logging in.

**Test Scenario:** Using fictitious patient accounts login and record if the user was brought to the Patient Front End page of the website.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of patients with pass/fail data on the test scenario.

#### 3.3.1 Personal Info (Patient Front End)

**Test Objective:** Verify that patients are able to access the Personal Info Section. Verify patients can make changes to their personal information through the user interface. Refresh and record if changes

have taken effect.

**Test Scenario:** Using fictitious patients accounts browse to the Personal Info Section, record the current information, attempt to make changes (record these as well). Record if changes were made after refresh.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of patients and original data, as what data should change, with pass/fail data of each field on the test scenario.

### 3.3.2 Medical Records (Patient Front End)

**Test Objective:** Verify that a patient can access the Medical Records section and access their medical record. Verify the patient can browse through their medical record by date, doctor, and location. Verify that the patient can print his/her record.

**Test Scenario:** Using fictitious patients accounts and fictitious medical records, attempt to browse the medical records by different fields and by viewing the entire record. After each record is accessed attempt to print to PDF and verify the results.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of fields searched and results, copies of PDF printouts for each test.

### 3.3.3 Prescriptions (Patient Front End)

**Test Objective:** Verify that patient can access the Prescriptions section and view their prescriptions as well as send them to a pharmacist.

**Test Scenario:** Prescribe fictitious patients medications using fictitious doctors. View the patients pending prescriptions. Using the fictitious patient account send the prescription to a fictitious pharmacy. Log in as the fictitious pharmacy and make sure the prescription has come through.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of each step of the process containing what data should be at each step of the test process along with pass/ fail information for each step.

### 3.3.4 My Doctors (Patient Front End)

**Test Objective:** Verify that a patient can access the My Doctors Section. Verify that a patient can add and remove doctors from their list as well as assign permissions to their doctors.

**Test Scenario:** Using fictitious patients add a fictitious doctor, assign permissions to the doctor, then remove the doctor. Repeat the process this time not removing the doctor.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of each step of the process containing what data should be at each step of the test process along with pass/ fail information for each step.

### 3.3.5 My Pharmacy (Patient Front End)

**Test Objective:** Verify that a patient can access the My Pharmacy Section. Verify that a patient can add and remove pharmacies and make a pharmacy their default pharmacy.

**Test Scenario:** Using fictitious patients add fictitious pharmacies, then choose to make one pharmacy the default pharmacy.

**Test Class:** Patient Environment

**Qualification Method:** Inspection

**Record Method:** Table of pharmacies created and which pharmacy should be the default.

## 4 Administrator Testing

### 4.1 Admin Registration

**Test Objective:** Verify that a user registered as a patient can be given Admin access by the super administrator.

**Test Scenario:** Create the admin accounts that will be used to do development on this project.

**Test Class:** Admin Environment

**Qualification Method:** Inspection

**Record Method:** List of admin accounts created and the associated person who will use the account.

### 4.2 Admin Log In

**Test Objective:** Verify that an Admin can log into the Joomla! back end Admin login.

**Test Scenario:** Each admin will login to the back end Admin Login at <http://www.medscribepro.com/administrator> .

**Test Class:** Admin Environment

**Qualification Method:** Inspection

**Record Method:** List of Administrators with pass/ fail data on login attempt.

### 4.3 Admin Features

**Test Objective:** Verify that administrators can use the features provided in the Joomla! back end.

**Test Scenario:** During the development process our team will be acting as Administrators and using the features in the Joomla! back end to build the project.

**Test Class:** Admin Environment

**Qualification Method:** Inspection

**Record Method:** As features are used to develop the project add the sections of the sites created to a table containing features and what sections were produced using those features.

### 4.4 Admin Development

**Test Objective:** Verify that the Super Administrator has access to the FTP Server and the MySQL databases using phpMyAdmin.

**Test Scenario:** Log into the FTP Server as the Super Administrator. Access the MySQL databases by logging into the phpMyAdmin client as the Super Administrator. Make changes to the database.

**Test Class:** Admin Environment

**Qualification Method:** Inspection

**Record Method:** Access chart with pass/ fail values of each of the systems the Super Administrator will need access to.

#### 4.5 Admin Testing

**Test Objective:** Verify that an Admin can create dummy accounts for patients, pharmacists, and doctors.

**Test Scenario:** Create multiple dummy or fictitious accounts to represent patients, pharmacists, and doctors for testing purposes.

**Test Class:** Admin Environment

**Qualification Method:** Inspection

**Record Method:** List of fictitious accounts created along with their login information and account type.