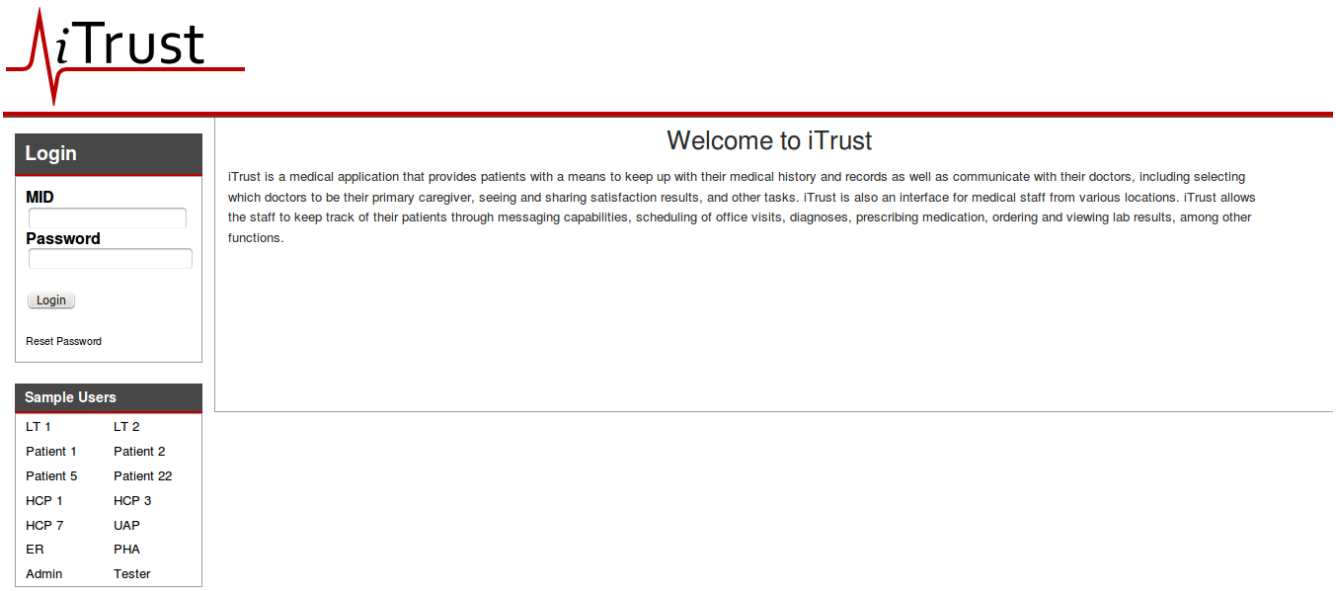
Short Tutorial on iTrust

During the course of this study, you will be working within the iTrust code base. iTrust is a JSP project written on Servlets. More simply, it's a Java program for the web. The affected files related to the tasks you will be performing will all be Java files.

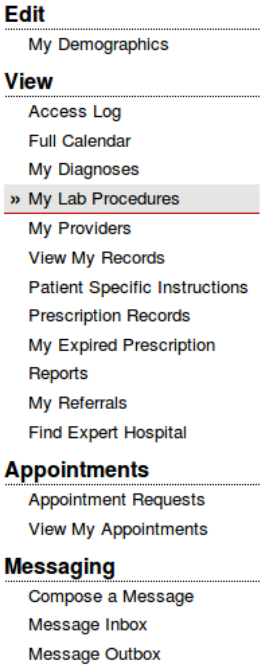
You can find more at http://agile.csc.ncsu.edu/iTrust/wiki/doku.php?id=start. Below we have included a shorter version describing iTrust. The long version of the requirements document is available at http://agile.csc.ncsu.edu/iTrust/wiki/doku.php?id=requirements

iTrust is a tool for medical professionals and patients to access, create, modify, and delete records. When you first reach the log-in page, the side bar will contain a log-in form and a list of sample users. For the purposes of this study, it is easiest to click on a sample user and you will automatically be logged in. Each link in this list is appropriately named based on the user time. Afterwards, you will be brought to a customized home page, including a feed of recent activity and a navigation side bar. See the figure below for an example of the login page.

Go to http://agile.csc.ncsu.edu:8081/iTrust/auth/forwardUser.jsp to view this screen in your browser and interact with the demo system. This would be a good idea to do before the study so you familiarize yourself with the user interface and system functions. Just click on the sample users to log in.



Each classification of users has a different list of options in the side bar, which is broken into categories, listed in bold, with individual pages listed as links below the category heading. Use these links to perform functions throughout the system. An example of the sidebar is shown below.



Although not an exhaustive list, here are some of the functions each classification of users may perform:

- **Patient:** Access medical logs, view upcoming appointments, and message his/her healthcare practitioners.
- **Health Care Practitioner (HCP):** Access and modify patient general information, health information, view list of office visits, schedule appointments, document office visits, and read messages from patients.
- **Lab Tech (LT):** View lab procedures, mark status of procedures as received, and provide results.
- **Emergency Room (ER):** View emergency patient report.
- **Unlicensed Assistive Personnel (UAP):** Edit patient health information, view report requests, and view laboratory procedures.
- **Public Health Agent (PHA):** Monitor adverse events, view diagnosis trends

Sample Ranking Task for Bug Fix

Ranking Scale

Please rank each entity on a five point scale by relevance to the bug being fixed

1 = lowest relevance	2	3	4	5 = highest relevance

Task Sample 1

The validation on the emergency contact name field of the Patient Information page does not work correctly. We want to make sure that the HCPs enter two words separated by a space in the emergency contact name field.

> **Expected:** System checks to see that HCP entered two words separated by a space in the emergency contact name field of the Patient Information page.
>
> **Actual:** System allows HCP to enter one word.
>
> **Related Use Cases:** UC10 Subflow S2

The bug was fixed in the following place

Class name: `PatientValidator`

Method/Member variable name: `validate`

Explanation:

This is how you should go about thinking about the bug fix.

I begin in class `EditPatientAction`. From `EditPatientAction.updateInformation`, I noticed it uses a `PatientValidator`. After going to `PatientValidator` I notice that all validation is done in `PatientValidator.validate`.

Entities to Rank

Source code entity	Your Ranking	How confident are you of the ranking?
`EditPatientAction.updateInformation`	1 2 3 **[4]** 5	Low Medium **[High]**
`PatientValidator.validate`	1 2 3 4 **[5]**	Low Medium **[High]**
`AddPatientFileAction.patients`	**[1]** 2 3 4 5	Low Medium **[High]**