CRISYS

Electronic Health Records

Status Report

CSE308: Software Engineering
Section 1
Professor Scott Stoller
May 5, 2009

Jonathan Farrell

Claude Castille

Timothy Quinto

Michael Wu (Hung Ngo)

For all documentation and progress, please visit us at:
WWW.CORPUSVILE.COM/CRISYS

This document contains the coding conventions, status report, test reports, and log files for the CRISYS Electronic Health Records system. This Document was completed March – April 2009 at Stony Brook University

Contents

Code Conventions

All of our coding conventions will strictly follow that of our main framework, CodeIgniter. For full conventions and other information, visit: http://codeigniter.com/

Implementation Status Report

At this point we have completed 100% functionality, including the following key functions:

- Server setup / hosting
- Framework setup on server
- Framework configuration
- Database configuration
- GUI Design / Implementation
- Log in including password hashing
- Homepage / permissions
- Manage EHR
- Manage clinicians
- Manage referrals
- Manage concealments
- Manage patients
- Manage agents
- Admin page with full add user and logs

Server setup / hosting: We have our server all configured and hosting space available. Hosting/serving is being provided by www.justhost.com. We have our content divided into two main sections: documentation and build. All of our documentation is available at:

http://www.corpusvile.com/crisys

The main project will be running live with the main access point being at either the sub domain or full path:

http://crisys.corpusvile.com
http://www.corpusvile.com/crisys/CI

Everything is up and running and all group members have ftp accounts.

Framework setup on server: We have all of the files for CodeIgniter extracted, setup, and running live on the server. The file structure is intact and all of the bare bone files are present. We have configured files within the framework to our specifications.

Framework configuration: The framework's configuration files have been setup to run on the server and database connections are working! All is up and running with no bugs!

Database configuration: The SQL CREATE script has been executed on the server and our database schema is up and running. Usernames/passwords and all permissions have been set and can connect without error.

GUI Design / Implementation: The GUI design has been implemented using html, css, and javascript. All pages will use the default layout file to display common features (header, footer, sidebar) and the content will simply be displayed using specific view pages within the framework.

Log in including password hashing: Log in functionality is working and will check a user's username / hashed password combination against the information stored in the database. If the submitted information is valid, flags and basic information is stored in the session variable including the users permissions. We use a hashing function when adding users and the same hashing function to check password validity. For example, "password" will be hashed to "5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8" in the database. We feel this is much more secure. All login functionality is contained in the login.php controller.

Homepage / permissions: The user's homepage is a dynamically changing page depending on who they are and what their permissions are. Each user's homepage is created on the fly and will display the proper menu items depending on permissions set for roles in the database. All homepage functionality is contained in the home.php view and permissions.php views.

Manage EHR: Users are able to view their EHRs and all items/item details. Clinicians are able to view their patient's EHRs and all items/item details. Clinicians and patients are able to add items to the EHRs but what items and what EHR depends on the user's role and permissions. EHR search supports by author, item type, or medical area. All EHR functionality is contained in the manageEHR.php controller.

Manage clinicians: Patients are able to add and remove a primary clinician. A patient may only add a primary clinician if they have none. If they already have a primary clinician they will NOT be able to add, they will ONLY have the option to remove that primary clinician. In addition, a patient can view a list of all clinicians that they have granted consent to, can add a new doctor and can always remove consent. All clinician functionality is contained in manageprimary.php

Manage referrals: A clinician may create a referral for any of his patients. He can search for a doctor based on medical area then create the referral. When a referral is created, the patient is added as being treated by the referred clinician. A clinician may cancel any referrals that they authored and a patient may cancel any referrals made for them. All referral functionality is contained in managereferrals.php

Manage concealments: A patient or full agent may conceal any item in the EHR and may also cancel a concealment that has been made for them. Clinicians cannot see any items that they were concealed from. All concealment functionality is contained in managecomcealments.php

Manage patients: A clinician can manage their patients by either searching or browsing their patient's EHRs. When a clinician views one of their patient's EHRs, there will be different functionality as compared to a patient viewing their own EHR. For example, a clinician cannot conceal any items in the EHR (which patients have the option to do), but clinicians can add items to the EHR (which patients cannot do). All patient management functionality is contained in managepatients.php

Manage agents: A patient or full agent have full access to managing agents. They can add or remove any agents. A clinician may add agents for any of their primary patients and may only remove agents that they appointed. All aganet functionality is contained in manageagents.php

Admin page with full add user and logs: The administration page has been separated from the main page of EHR service. This was done just to make it cleaner and separate. All login and management works exactly the same way, it is just a different login portal. The only required administration function was to add a user, which we have completed. We are adding on administrator functionality as we need it. Currently, this includes:

- Adding a new user user with the option to make them a patient, a clinician, an administrator, none or any combination.
- View a list of all active users in the system.
- View the log files for the Crisys system.

All administration functionality is contained in admin.php
The administration page can be accessed by clicking on "Admin Tools" at our information page: http://www.corpusvile.com/crisys

This is just a brief outline of the functionality that is completed. To view and experience everything that has been completed, please login to our system and demo it for yourself with the login information provided below!

100% of coding is completed!

The code submitted is an exact copy of what is running on our server (as of 5/4/2009).
The current state of our system can always be viewed/demoed at: http://crisys.corpusvile.com
Username: 308
Password: demo

User Manual

Please refer to the "UserManual.pdf" file in the docs folder.

Privacy

Please refer to the "Privacy.pdf" file in the docs folder.

Test Report

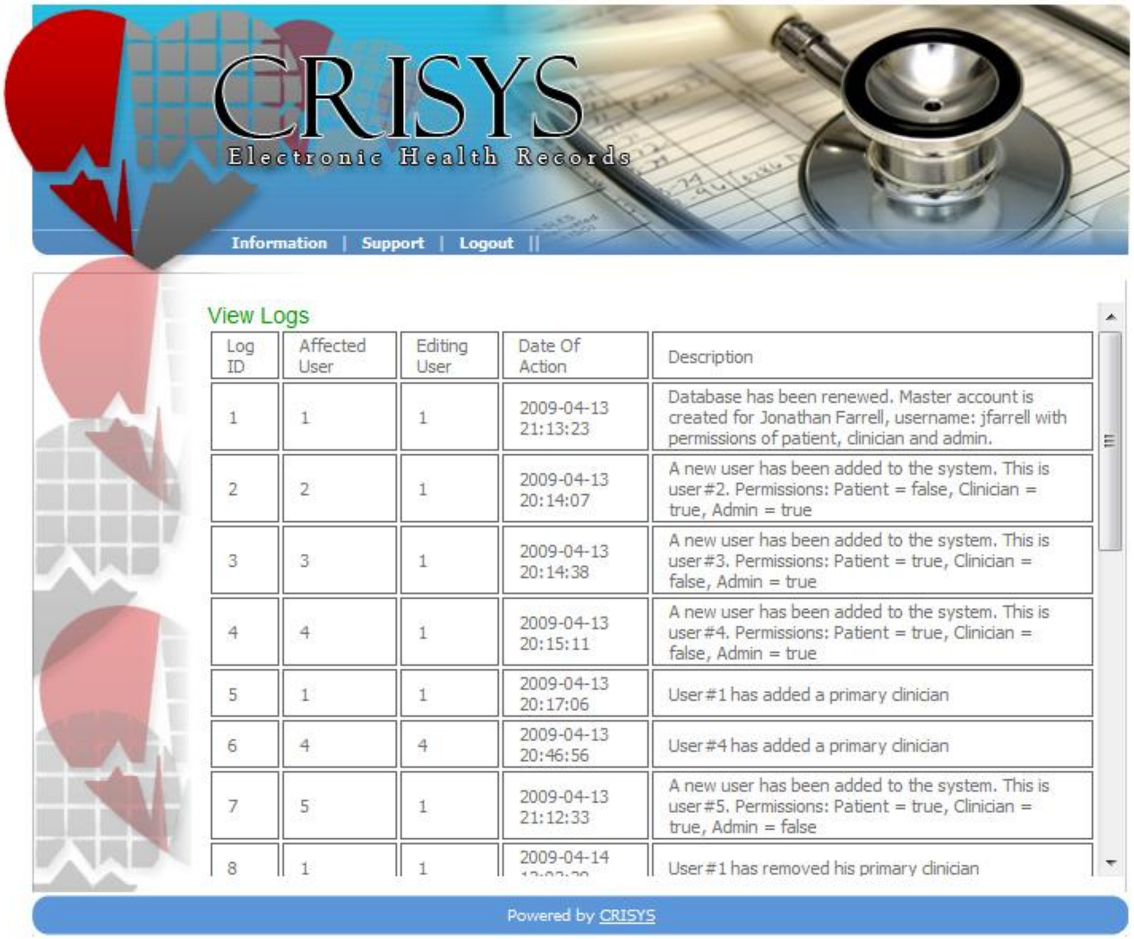
Please refer to the "TestReports.pdf" and "TestReports_AppendixAB.pdf" files in the docs folder.

Log Files

Our system is currently set up to log and is logging. We have created a table in our SQL database to handle logging. The table (Log) has the following attributes:

- ID
- Affected User
- Editing User
- Date/Time
- Description

When log files need to be viewed, an administrator may log into the system and simply click “view logs”. Below is a screen shot of the page the administrator will have:



View Logs

Log ID	Affected User	Editing User	Date Of Action	Description
1	1	1	2009-04-13 21:13:23	Database has been renewed. Master account is created for Jonathan Farrell, username: jfarrell with permissions of patient, clinician and admin.
2	2	1	2009-04-13 20:14:07	A new user has been added to the system. This is user#2. Permissions: Patient = false, Clinician = true, Admin = true
3	3	1	2009-04-13 20:14:38	A new user has been added to the system. This is user#3. Permissions: Patient = true, Clinician = false, Admin = true
4	4	1	2009-04-13 20:15:11	A new user has been added to the system. This is user#4. Permissions: Patient = true, Clinician = false, Admin = true
5	1	1	2009-04-13 20:17:06	User#1 has added a primary clinician
6	4	4	2009-04-13 20:46:56	User#4 has added a primary clinician
7	5	1	2009-04-13 21:12:33	A new user has been added to the system. This is user#5. Permissions: Patient = true, Clinician = true, Admin = false
8	1	1	2009-04-14	User#1 has removed his primary clinician

We are currently logging:

- When a new user is created
- When a user adds a primary clinician
- When a user removes a primary clinician
- When an item is added to an EHR
- When a medical field is added to a clinician’s profile
- When a medical field is removed from a clinician’s profile
- When a referral is created
- When a referral is cancelled
- When consent to treatment is added
- When consent to treatment is removed
- When an agent is added
- When an agent is removed

To view a dump of the logs which were present after running the rough draft of our demo, please refer to the “Logs.html” file in the log folder.