Description

Title	CS 4389 EMR
Team Members	Gwen Eskridge, Colin Haney, Jacob Gillespie
Delegation of Tasks	Colin developed the data model design and implementation Jacob developed the server backend Gwen developed the web frontend
Timeline	1. Assigned Project 2. Delegated tasks to team members 3. Developed data model 4. Developed frontend and backend simultaneously 5. Integrated frontend and backend
Motivation	The lack of communication between various physicians, patients, and other medical personnel is a significant problem. Our goal is to create an easy-to-use electronic medical record system that can allow various actors to share information securely.

Introduction

We designed and implemented an electronic medical record system that allows secure medical data to be shared in a hospital setting with various different roles, such as billing personnel or physicians, on behalf of a patient.

This project demonstrates various different data and application security principles working in a real-world setting. We wanted to build something that remained secure, but was easy to use. We each brought our own technical knowledge into this project, and Gwen brought her own personal experience with EMR systems in the field.

Background and Related Work

There are many, many electronic medical record platforms available for the healthcare industry today. [1] The domain covered by the management of medical records, patients, and personnel is vast, and as such there are many varied competing products / alternatives. While we, in our project, could not hope to truly rival a commercial EMR system, we wanted to build a solid starting point for one that focused on usability without sacrificing security.

Implementation and Results

Our implementation tech stack included a PostgreSQL database, a backend written in Ruby with the Ruby on Rails framework, and a web frontend powered by AngularJS. For hosting the application for development / testing purposes, we hosted the application on Heroku, a “platform as a service” that manages the application servers and database hardware for the application. In a real-world setting, we would have upgraded to some of their isolated hosting options to comply with privacy regulations when dealing with health information (HIPAA [2]), however that option was significantly outside our budget for development.

This particular tech stack, though, is more or less inconsequential to our goal - making something easy to use. This we accomplished through a simple but solid data and access model as well as a web frontend that provided universal access.

From a data and access model, users were classified as a patient, physician, or billing. Access rules were defined as the following:

- Patients can access their own records and can update basic demographic information
- Physicians can access records that they have created for a specific patient
- Patients can authorize specific physicians to access records created by other physicians
- Billing personnel can access demographic and billing information for all users
- Anything not specified in these rules is assumed to be denied

For security, the application requires users to connect with TLS (https) so that all traffic between the browser and the backend servers is encrypted. Additionally, the interface between the frontend and backend is secured via cookie sessions and XSRF tokens. The Ruby on Rails framework also brings battle-hardened protection against many common web application exploits. And finally, a basic audit log was included to allow external auditing of the system to ensure data integrity and monitor security.

By providing a web-based frontend, the system is more universally accessible to patients and hospital staff alike as only a web browser and internet connection is required. The system’s accessibility was further enhanced by focusing on the user experience when designing the frontend.

Development of the frontend loosely followed the process described by Hartson and Pyla’s wheel, which consists of essentially four stages: Analyze, Design, Implement, and Evaluate. [3] After the final stage, the process loops, beginning again at the Analyze stage.

The analysis stage began with an attempt to identify issues with current EMR systems. These issues were discovered by first discussing the problems that individual group members had

encountered when dealing with healthcare providers. Further issues were identified through a group member's personal experiences as a medical scribe, and from discussions with medical personnel who were currently working in the field. Once this information was gathered, examination of the data hinted at several patterns. These patterns are distilled into the following major issues:

- There is a lack of adequate communication between patients and healthcare providers. Patients seldom know the contents of their health record, and providers often miss crucial information about a patient, such as allergies or chronic conditions, despite that information already being documented.
- Current EMRs are not user-friendly. Use of commercial EMRs requires weeks of training, which inhibits effective communication between providers and patients.
- Medical information is not consolidated, but is instead spread between many systems in many places, making it more likely that information will be lost or overlooked.

After the completion of the initial analysis stage, design began. The design stage consisted of breaking down the project into several use cases, and creating sketched storyboards to describe each case.

The implementation stage was approached in an incremental fashion, with each use case being the basis for one increment. Once the increment was complete, it was tested for correctness and usability in the evaluation stage.

Conclusion and Future Work

A significant amount of development would be required to take this base and form a commercial product, so with an eye to future improvements, there are many areas that could be developed or polished, including visual design, functionality, additional security / compliance layers, integrations with third-party systems, etc. Overall however our project was successful in its goal of being a simple but solid base for implementing an EMR system. The baseline is in place.

References

[1] Software Advice, 'Compare Electronic Medical Records Software', 2015. [Online]. Available: http://www.softwareadvice.com/medical/electronic-medical-record-software-comparison/.

[2] HHS.gov, 'Health Information Privacy', 2007. [Online]. Available: http://www.hhs.gov/ocr/privacy/.

[3] R. Hartson and P. Pyla, *The UX Book: Process and Guidelines for Ensuring a Quality User Experience*, 1st ed. 2012.