6-11-2017

EHRaS: Electronic Healthcare Record Management for Ad Hoc Settlements

Jack Kingsman
Santa Clara University, jkingsman@scu.edu

Evan Paul
Santa Clara University, erpaul@scu.edu

Max Werner
Santa Clara University, mwerner@scu.edu

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

Date: June 11, 2017

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Jack Kingsman
Evan Paul
Max Werner

ENTITLED

EHRaS: Electronic Healthcare Record Management for Ad Hoc Settlements

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Ahmed Amer, Senior Design Advisor

Department Chair

EHRaS: Electronic Healthcare Record Management for Ad Hoc Settlements

by

Jack Kingsman
Evan Paul
Max Werner

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science and Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 11, 2017

EHRaS: Electronic Healthcare Record Management for Ad Hoc Settlements

Jack Kingsman
Evan Paul
Max Werner

Department of Computer Engineering
Santa Clara University
June 11, 2017

ABSTRACT

Providing reliable and accurate healthcare to refugees in an ad hoc setting can be challenging for a variety of reasons, and most settlements currently utilize little to no record keeping for patients. EHRaS, our mobile-first electronic healthcare record management system, can provide an inexpensive electronic healthcare record (EHR) solution that is flexible in last-mile applications where infrastructure and technical support is at a minimum. By utilizing NFC technology, novel caching practices, and an extensible interface, our system not only securely identifies patients and authorizes staff to access sensitive medical data, but generally provides a competitive, low-overhead alternative to other open source EHR systems.

Acknowledgements

Special thanks to Professor Ahmed Amer for his guidance, support, and confidence in our vision.

Thank you also to Fr. Maria Joseph Israel, whose input on EHRaS was invaluable as we considered our design goals.

Finally, thank you to the Department of Computer Engineering and SCU at large for their support and education over the last four years.

Table of Contents

List of Figures

Chapter 1

Introduction

The more than 54 million refugees and displaced persons worldwide[1] are among some of the most vulnerable and overlooked populations on the planet. While often direly in need of medical services, diverse geographic backgrounds and especially varying levels of personal documentation can cause issues for effective delivery of aid. While medical care is perhaps one of the most vital services rendered in ad hoc settlements, it can also be the most demanding in terms of security and availability: maintaining medical records typically requires a great deal of coordination between patient and practitioner, overhead, and organizational consideration to be secure and effective.

Paper records are simple for field doctors to maintain and certainly have the longest history of use in developed medical care systems. However, they suffer from a lack of portability and accessibility (they can only be in one place at one time, generally being accessed by one person), both of which are vital in a dynamic environment such as a refugee camp. Perhaps even more importantly, ensuring physical security can be difficult and risky in an environment that may not necessarily readily provide the tools to do so.

Electronic healthcare record keeping systems are by no means new, and academic research has been ongoing for over two decades stretching back to the early days of the Gopher protocol[2][3]. The advantages in speed of patient processing[4] and overall increase in quality of patient care[5] have been well studied.

EHR systems have generally been regarded as high cost, carrying great technical overhead, and requiring a full PC to interface with as a doctor. There are, however, open source solutions that provide alternatives. OpenEMR is the most popular open source solution and provides integrated billing, scheduling, and prescription handling[6]. Similar solutions such as OpenMRS and Bahmni provide generally the same functionality. Bahmni is unique in that it provides a mobile interface[7], but, just like OpenEMR/MRS, requires at least a medium power server to successfully host the

application.

Overall, open source EHR solutions have been able to provide advantages over traditional large-scale EHR systems in many ways. However, they still require hefty installation processes and, paradoxically, provide so many features that it becomes too large scale for effective use in ad-hoc settlements – for doctors making the first leaps away from a zero record keeping environment, an EHR appropriate for a full hospital complete with a billing department and laboratory staff could be overwhelming.

We propose an electronic healthcare record management system, EHRaS, that enables providers to serve people in disadvantaged, ad hoc regions quickly, accurately, and securely. EHRaS seeks to bridge the gap by providing an interface and deployment process appropriate to small to medium scale settlements that need only the basic building blocks of an EHR; it leans heavily towards freeform data and minimalism, seeking to be more of an organized patient notebook rather than a comprehensive enterprise healthcare solution.

EHRaS is a mobile-first electronic healthcare record management system. It is a web application that:

- provides data security (both in flight and at rest)
- provides a usable and intuitive interface for medical users
- retrieves only one patient record at a time

It is comprised of a web portal, a backend, and unique patient identifiers used to retrieve healthcare records. In order to design EHRaS to meet these functional requirements we divided our project into four objectives that were carried out as sprints:

1. Mockup and design where we determined the authentication flow for record retrieval
2. Frontend build-out, where we designed our intuitive user interface
3. Backend integration, where we designed our secure database
4. Debugging and testing, to asses progress and ensure our system works as intended

Our mobile-first approach will allow minimal equipment overhead and is usable on low bandwidth and intermediately connected networks. A centralized storage system will minimize requirements for on-site record security and allow a single point of access through which all record requests flow. By providing an easy to use system that enables care for disadvantaged populations, the universal right to health and wellbeing can be extended to all.

1.1 Selected Screenshots

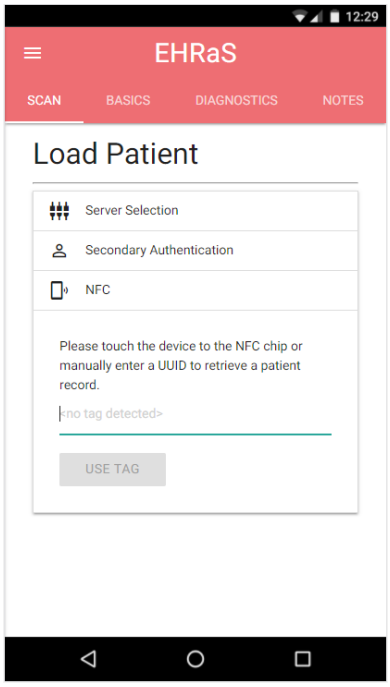

Figure 1.1: Patient load/tag scan screen

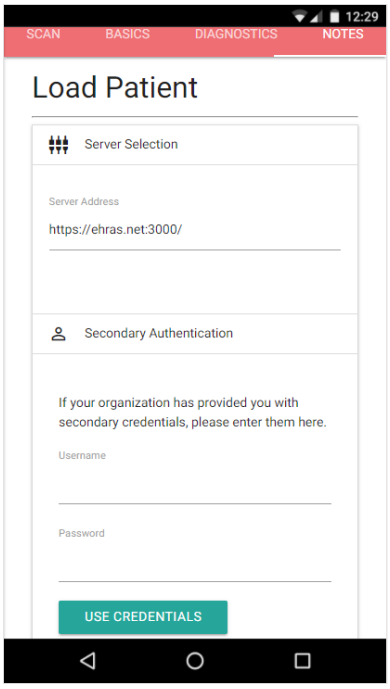

Figure 1.2: Server selection and secondary authentication

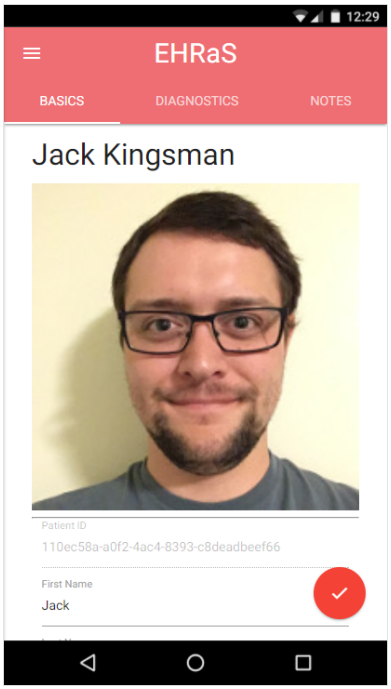

Figure 1.3: Loaded patient interface

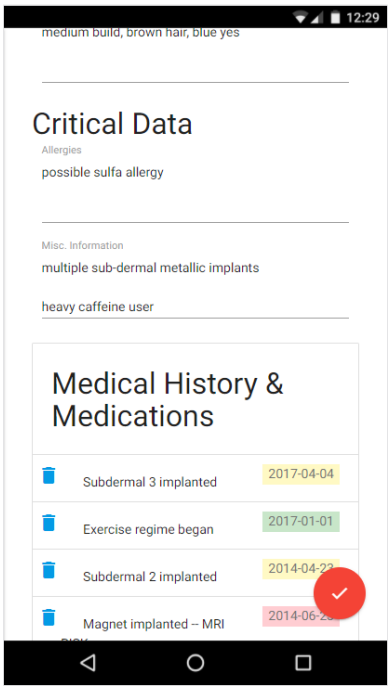

Figure 1.4: Additional patient data and charting

Chapter 2

Data Flow

EHRaS identifies all patients by a universally unique identifier (UUID) that ties their healthcare data to their identity. While EHRaS is intended to be used with NFC identification tokens, it can technically be used by manually keying in any sort of globally unique identifier – the authentication flow as seen after the UUID (beginning at **/session/:uuid**) is decrypted is identical for NFC and manually keyed IDs.

An authorized user scans an encrypted UUID from a patient's NFC chip (which may be blank if it has not been registered with the system). If the chip is empty, the application sends a GET request to **/uuid/new**; otherwise, the UUID is sent to the server for decryption via a POST request to **/uuid/decrypt**. In either case, the client will receive a decrypted UUID (which will be written to the chip in case of an empty tag) which is subsequently sent to a different server endpoint (GET **/session/:uuid**) to receive a temporary session key (EHRaS allows users to configure the session key duration so that access to a patient's data is limited).

Once the session key is received, one last GET request is made to **data/:uuid/:sk** to receive the patients data, which is then used to populate the front-end. Any updates to the patient's data are sent automatically via a POST request to **/data/:uuid/:sk** (as long as the session is still active). Once a session expires or another tag is loaded, the session key is invalidated and any reuse attempt will fail.

This process is illustrated visually in Figure 2.1.

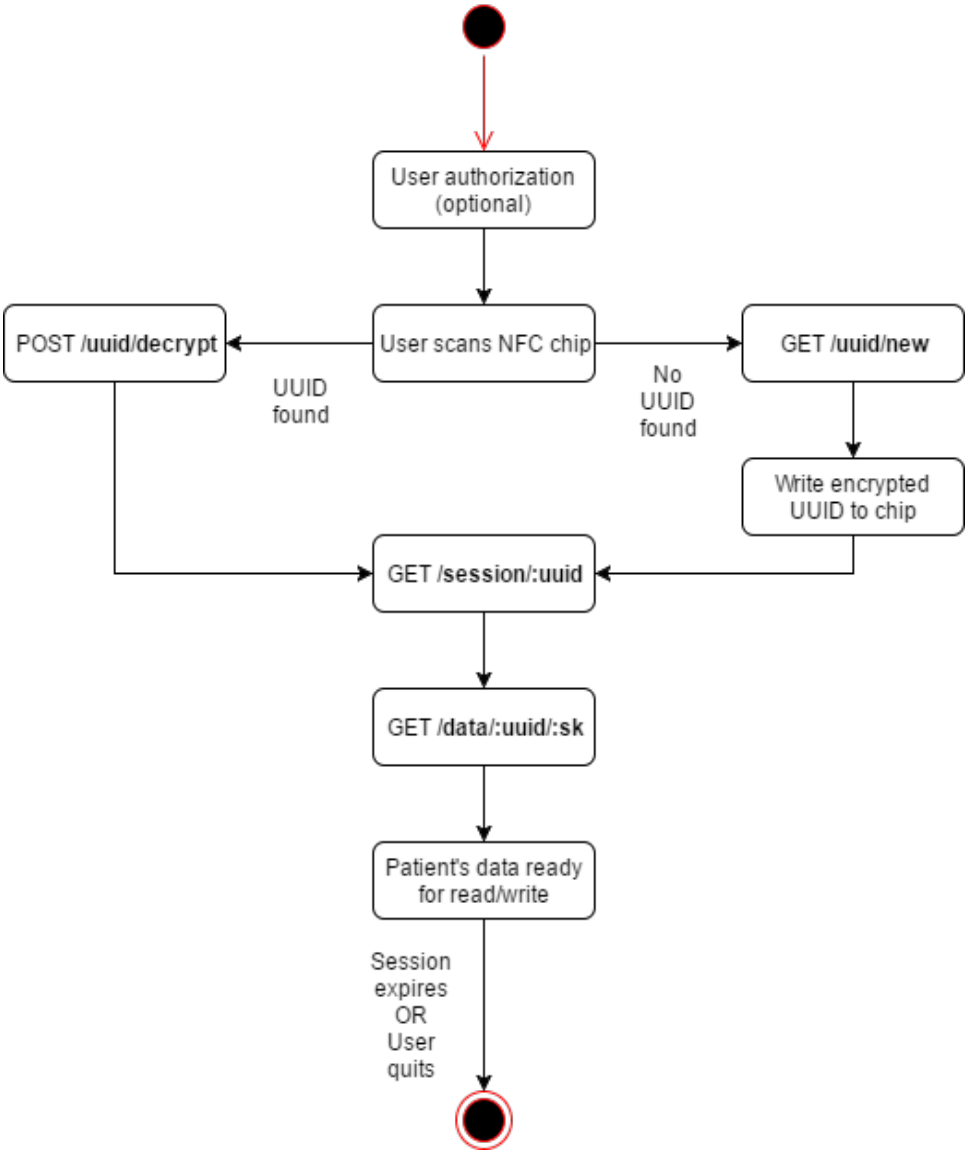

Figure 2.1: Activity Diagram

Chapter 3

Technologies Used

Our design features a standard client/server architecture. To ensure that our application is usable even on sub-standard hardware, we have decoupled the presentation layer of our system from the data layer: the web-based user interface consumes an API to and from which it makes and receives requests. Data CRUD occurs in the API portion only which allows for a bottlenecked backend to queue requests while the front-end remains responsive as opposed to a monolithic architecture.

3.1 Front-end

On the front-end, the application is rendered using HTML and JavaScript, high level languages used to describe the form/function and interactive/data-handling components of a webpage respectively. We use Materialize, a user interface design framework with standardized components in compliance with Google's Material Design guidelines, to style the HTML and provide boilerplate CSS, a language used to style and alter the display of HTML.

To avoid simply serving static HTML pages we use Handlebars.js, a powerful yet lightweight templating framework. This allows EHRaS to dynamically populate patient profiles easily.

The front-end is served by a standard static file server within Express, a Node.js module for servers.

3.2 Backend

The API interface is served via Express and handles web requests, authentication, and CRUD on backend data.

3.2.1 Authentication and Encryption

EHRaS does not enforce any particular user authorization scheme. Instead, it provides stubbed functions on that backend that easily allow for organizations to inject their own pre-existing authorization system (such as LDAP, RADIUS, Kerberos, etc.), or implement a simple one solely for EHRaS. Front-end provisions for authorization are already in place – the end administrator need only write the Node.js function to verify the provided credentials.

Once the user has been authenticated to the server, they can begin to scan a patient's NFC chip. UUIDs stored on the chip are encrypted using AES-256 and any data traveling from clients to server endpoints (or vice versa) are encrypted via HTTPS. Access to a patient's data is granted in the form of a session key. Session keys are valid for a limited duration which can be configured. Once a session key expires, that patient's profile can no longer be read or written to without scanning their NFC chip again. All patient data is encrypted at rest in the database via AES-256.

3.2.2 HIPAA Compliance

Although HIPAA is a restriction specific to the USA, it provides an excellent starting point for international certification and is, in our unqualified opinion, an appropriate stand in for locales that don't provide healthcare privacy restrictions of their own.

EHRaS is as HIPAA compliant as possible while still retaining the freedom for the end user to customize their installation. Access controls must be implemented by the end user (as mentioned above), from which auditing and access logging can follow. Once access control and auditing has been established (which, depending on the level of integration required, can be as simple as a couple of lines of username/password checking and log retention in Express), the end administrator is ready to begin final reviews for technical HIPAA compliance[8].

3.2.3 NFC API

In order to achieve NFC compatibility, we have decided to utilize Mozilla's HTML5 NFC API in order to abstract NFC functionality for future developers. Unfortunately, while this API gives us the ability to structure our code to meet standard NFC data formats (and allowed us to complete our project on time), the HTML5 specification is still in a draft state, and therefore not available on consumer devices yet. Since the Mozilla NFC API is only available for privileged or certified applications on Mozilla hardware, we have used it to establish a template for future development to build upon. Our template utilizes NDEF encapsulation which will standardize development when integrating any new storage/data transmission APIs.

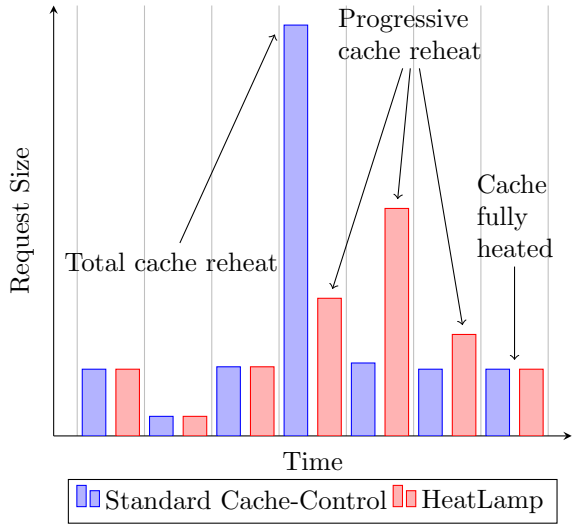

Figure 3.1: Hypothetical max-size-per-request reduction from HeatLamp. Note the lower maximum size afforded by HeatLamp's progressive cache invalidation as three assets' caches expire (all at once for the standard cache-control, and progressively for HeatLamp) at the 4th entry of the graph.

3.3 Caching and Bandwidth: HeatLamp

As our team considered our goals for EHRaS, it was important to us that we not just satisfy the goals for functionality and interface, but also to consider the deployment environment that EHRaS would be used in. Local data networks may not feature high bandwidth mobile connectivity; in the worst case, for example, EDGE data speeds may be as poor as 8kbits/s[9]. As networks proliferate in ad hoc settlements and developing nations, bandwidth is of prime concern. With microcellular hotspots (such as OpenCellular[10]) and aerially transmitted internet (such as Facebook's Project Aquila[11]) on the rise, conservation of in-flight payload size back and forth between the back end was a major design consideration.

To ensure that EHRaS remains usable even in degraded or under-performing network environments, we designed the software to use up-to-date compression and caching standards for web access, including GZip compression and generous cache timing. However, beyond baseline optimizations, we were concerned that the support code for our UI frameworks would cause an undue amount of stress on the network – in a low bandwidth environment, 100 kilobyte scripts could cause significant strain as caches invalidate over the course of normal network operations.

To remedy this, we developed a novel cache management scheme called HeatLamp[12]. HeatLamp randomly staggers Cache-Control header ages to ensure that the client-side HTTP response cache never invalidates all at once (which would cause the first site request after resource expiration to attempt to heat the cache all at once). Because HeatLamp causes resources to expire in random

intervals within a given time window, total and average traffic for a complete cache refresh is unchanged but maximum size of any request is greatly reduced (statistically, more than one resource's cache expiring at once is highly unlikely).

To our knowledge, this style of cache management has not been implemented in a modular format before (understandably so, as cache invalidation on the order of hundreds of kilobytes usually comprises a minute fraction of broadband bandwidth that the average consumer would have). Therefore, we have released HeatLamp as open source software on the NPM (Node Package Manager) site as a plugin for the node.js web server package Express in order that other developers of web applications for bandwidth-limited environments can benefit from our work.

Chapter 4

Testing Procedure

A large portion of our testing simply involved our team using EHRaS and fixing any issues found. We focused on testing our system in a way we would expect a typical user top act, as well as trying things that most users would not attempt to locate edge and corner cases of the UI. Most of the bugs we encountered were located and fixed through this methodology.

4.1 Automated Testing

Integration and system tests were all performed in an automated manner via a lightweight testing framework. The framework was written with asynchronisity in mind due to the dynamic nature of our system.

Expected values for items such as field values, page states, and other useful diagnostic information were tested against actual values to verify that new code didn't break previously functioning modules. We found this style of integration testing (as opposed to rigorous unit testing) to be indispensable to our workflow.

Due to the in-browser nature of the testing framework, we eliminate the need for complicated automated cross-browser testing or potentially overly-technical continuous integration that may not be appropriate for the disconnected nature of our expected deployments – if the tests pass in the browser, EHRaS is ready to use that device.

4.2 Usability Testing

Cognitive Walkthrough (see Ch. 6.9.1)

Chapter 5

Considerations for Further Development

We are aware that we are the beginning and not the end with of project. This chapter will outline how other developers and users can interact with EHRaS to prepare it for implementation.

5.1 DICOM and In-Place Extension

EHRaS' Diagnostics tab allows for the embedding of diagnostic imaging data from MRIs, x-rays, CAT scans, and so forth. EHRaS provides space for remarks and data about the particular diagnostic as well as the ability to link to off-site diagnostics, allowing for dynamic updates of images as they are made available.

EHRaS also features the ability to use an embedded iFrame as the content for an entry on the Diagnostics tab. When combined with a hosted diagnostic imaging system that can serve medical imaging data (such as MRIs or CAT scans), this functionality can provide in-app browsing of high-density test results such as DICOM packages [13] or other rich content (see Figure 5.1).

Ultimately, the usefulness of this feature is only limited by the number of frame- and mobile-friendly content display systems that can be set up and made accessible to EHRaS.

5.2 Documentation and Code Commentary

Excellent documentation and code commentary seek to not just clarify complex portions of code but to provide end administrators with an intuitive understanding of EHRaS' data flow.

The industry standard JSDoc JavaScript documentation platform provides a quick reference when editing code and is used to generate full-platform documentation in an easy-to-read HTML format for more in-depth analysis of all functions used on the front-end.

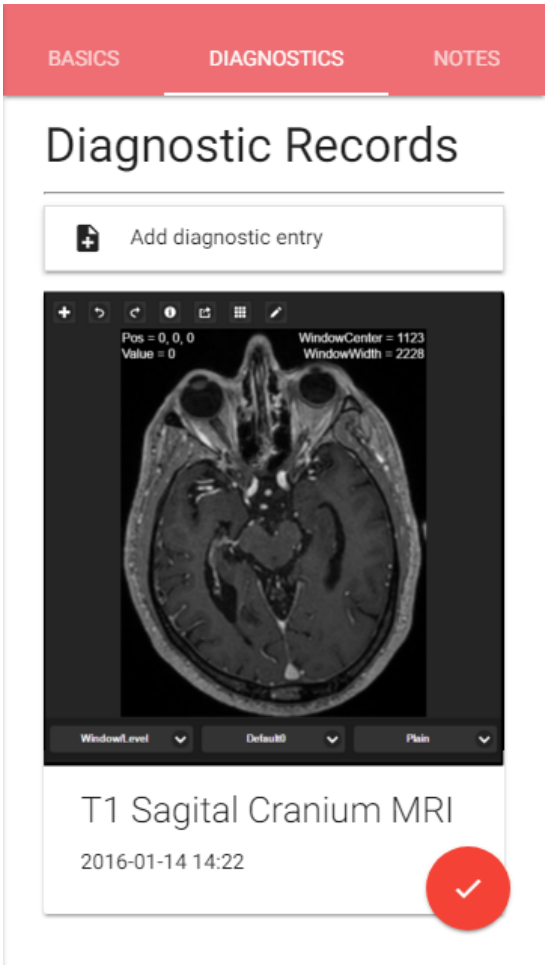

Figure 5.1: Embedded mobile DICOM imaging viewer displaying a cranial MRI via IVMartel's DWV system[13].

5.3 Authentication and Security

EHRaS is designed with flexible and open-ended security that is secure out-of-the-box while still providing easy customization and enhancement depending on the end user's needs. In-flight and at-rest encryption ciphers are easily swappable to any implementation available for Node.js if the suite already in place (Google's CryptoJS library) is insufficient.

User authentication is fully allocated for in the code but intentionally left minimally implemented – any authentication system that has a JavaScript extension (whether that be database-oriented, RADIUS, LDAP, Kerberos, etc.) can be quickly and simply hot swapped in as part of the authentication toolchain.

5.4 Licensing

Given the short timespan that our team had to complete EHRaS, it was important for us to contextualize our role as an initial developer rather than provider and supporter of a single, final version of the system. EHRaS and all its components developed for the Senior Design Project (such as HeatLamp) are permissively licensed under the MIT License. By providing nearly no restriction on future development, incorporation into other projects, and so forth (with only a minor proviso that exempts the design team from legal liability), EHRaS can be a complete solution or simply a jumping-off point for future development or remixing to fit the exact needs of the end user.

5.4.1 Open Source Access

EHRaS is available on GitHub at https://github.com/EHRaS/. This organization page provides access to the following components:

- **Frontend**: provides the server component that provides browsers access to EHRaS thing client
- **Backend**: the engine of EHRaS; provides data storage, tag management, and a connection for the front-end to communicate with
- **HeatLamp**: the source code for the HeatLamp component; also available on NPM (https://www.npmjs.com/) under *express-heatlamp*
- **Creative-Assets**: branding, screenshots, and miscellaneous creative assets
- **Licensing**: attribution and licensing details for subcomponents of EHRaS

Chapter 6

Societal Issues

In this chapter we will cover how EHRaS interacts with society and humanity in general and discuss the prevalent issues surrounding our work.

6.1 Ethical

While the intention of EHRaS is clearly to help people in ad hoc settlements, our system involves a large amount of personal and sensitive medical information. If this information is not handled with the security of the patient's medical data in mind, there could be information leaks which could be devastating for a patient.

In order better understand refugee camp conditions and the general needs of the people within them, we met with Fr. Maria Joseph Israel[14], a member of the Jesuit community at Santa Clara University with a background in providing educational opportunities in refugee camps. On this issue of patient identification, Fr. Joseph explained that many refugees do not disclose their name or identity. Many times refugees enter a camp with little or no identifying documentation due to their expedited escape from their native country; even within the confines of a refugee camp, a refugee could still be at risk of being thrown out of the camp or even killed if their identity is revealed to the wrong people.

EHRaS was designed with this in mind. Rather than needing to give up their name and identity when visiting medical services, our system identifies patients using a unique identifying token stored on a patient's NFC chip or another secure medium. As long as the patient has their chip or token, their medical record can be accessed. NCF authentication provides a simpler solution to identification than biometrics. The technology required for NFC authentication can be completely non-invasive where the patient does not need to touch or scan any part of their body. Furthermore, the usage of NFC/tokens allows for patients to easily shed their connection to the system by discarding their chip.

Unlike methods such as biometric identification that are forever and irrevocably tied to a human, we abstract identity from identification – the power of control over one's sensitive information is put back into the hands of the patient

Another concern that Fr. Joseph cited was the amount of access a doctor or authorized user would have in regards to the database of every patient's medical records. This is important because if a single user could access as many records as they desired, then none of the patients within our system would be reliably safe.

We designed EHRaS' authentication system to limit doctors to only have access to one record at a time. A patient's universally unique identification number (UUID) is needed to access a record. That UUID is stored in an encrypted format on an NFC chip. In other words, a doctor needs to have a patient's NFC chip in order to access the healthcare record, and the doctor will have no way of accessing the record without the NFC chip with the UUID.

6.2 Social

In its current state, our system is not entirely ready for deployment. We have outlined and designed a functional data handling system that could be scaled up to then manage an ad hoc settlement's medical records as the primary system. If it were to be modularly expanded and implemented on a large scale, we think EHRaS would greatly assist large populations of people by managing their healthcare records in a more efficient and detailed manner. The drawbacks of paper based systems would be eliminated, and our more organized system could instead be implemented.

When considering the potentially negative impacts our system could have, we looked at our system from the perspective of the refugee camp issue of identity discretion. While our system does hold personal information about a patient, the alternative is not having records at all. The costs and likelihood of extreme encounters (such as the potential of an armed assailant looking to get patient data) need to be considered when deciding to implement our system in a refugee camp setting.

6.3 Political

EHRaS is an aid that seeks to improve upon systems used by ad hoc settlements for managing healthcare records. While our goals are not to interfere with the political systems of countries with refugee camps, we recognize the importance in considering how our system could have a political impact – ultimately, while providing medical care to persons in need would ideally be an apolitical act, geopolitical tensions and restrictions on certain populations need to be holistically evaluated

before any sort of aid is rendered.

6.4 Economic

One intention with EHRaS was to keep cost as low as possible. For many developing nations, an enterprise-level digital healthcare records management system would be far too expensive. Our system counters this by offering a mobile-first solution. ERHaS is extremely lightweight, specifically when it comes to server load. The web server backend for ERHaS can be run on consumer hardware. After server setup, an Android smartphone with NFC capabilities is all that is required to scan and access a patient record as identified by a required NFC chip or token for each patient. Hardware costs would be minimal compared to the expenditure for the ad hoc settlement as a whole. A complete installation with minimal hardware could be accomplished for under $200; a larger set up for 10 doctors with hundreds of patients would be less than $1000.

6.5 Health and Safety

The health and safety concerns for this project are primarily concerned with the security of patient data. Please see our Ethics section at the beginning of this chapter for more information on this.

6.6 Manufacturability

EHRaS as a web application itself does not need to be manufactured in order to be reproduced; the components required for the entire system are listed in the Economic section. As a whole, EHRaS can be "manufactured" quite easily so long as enough server space, Android Phones, and NFC chips can be distributed. How much of each depends on the method in which the system is deployed.

6.7 Sustainability

EHRaS is a low-impact digital solution; sustainability and longevity of the system is not of particular concern once it is deployed. Power interruption and network degradation are the primary concerns with EHRaS; unfortunately, assurances against these issues within an ad-hoc settlement are far out of the scope of this project.

6.8 Environmental Impact

The environmental impact of our system is minimal. As a web-based application with a lightweight server load, EHRaS is as low impact as websites with databases come. Extra care has been taken to

ensure that EHRaS sips resources during standard operation, including novel caching systems and efficient crypto implementations.

6.9 Usability

We have completed a formal design evaluation of EHRaS to test usability and performed a cognitive walkthrough to examine how a user might traverse our system.

6.9.1 Cognitive Walkthrough

Task: Examine and update existing patient health record

Optimal action sequence:

1. Load patient via their UUID
   (a) As the crux of our security system, users will need to be familiar with the process of scanning NFC chips to retrieve a patient's UUID. The NFC accordion tab is the first thing opened on the Load Patient page. This helps the user understand that this action is needed to load the patient record.
   (b) The user will notice that this correct action is available because the NFC accordion tab instructs the user to touch their device to the NFC chip. This shows good visibility of system status to the user.
   (c) The user will associate this action with their desired effect because the instructions on the NFC accordion tab say to enter the UUID in order for the record to be retrieved.
   (d) The patient will automatically be taken to the patient record upon successful entry of the UUID. The feedback in this sense is actually taking the user to their desired page, the patient's record.
2. Enter new information into desired data fields
   (a) The editing of a patient's record is as simple as clicking the desired field to edit. The user will know to edit fields to update information because they are now viewing the patient's record.
   (b) The user will know they can edit this page because for both the Medical History and Diagnostics sections, the user will see an "Add [section] entry". Should the user wish to delete an entry in either of these sections, they can do so by clicking the blue trash can. The process for adding and deleting entries in these fields demonstrates good visibility of available actions to the user.

(c) For editing the Medical History and Diagnostics sections, both creation and deletion are signified with an icon of a paper with a plus on it and a trashcan respectively. These common signifiers help the user associate the correct action with their desired effect of adding or deleting entries.

(d) The user is informed of the progress they make in adding and deleting entries through Toasts. Toasts are ephemeral feedback messages that appear in dark grey once the user adds or deletes an entry in the Medical History and Diagnostics sections.

3. Save changes

(a) The user will know their work will be saved because our system autosaves every 10 seconds. A manual save button also exists for users who wish to save manually.

(b) The save button is located on the bottom right side of the page with nothing else around it. Because the button is a good contrasting color to the white background and the fact that it persists in the same location no matter how far down the page, it has good visibility for the user.

(c) A check mark makes sense as a save button because it is a common signifier for completing an action. Seeing as how this is the final step in making changes, it makes sense to use a checkmark to bookend the editing session.

(d) Once clicked the user receives feedback that the data was saved through a Toast.

6.10 Life-long learning

EHRaS presented unique challenges to the entire group – all members were able to acquire career-empowering skills ranging from NFC protocols to dynamic cache optimization to novel JS templating frameworks to learning more about refugee camp conditions. As we worked to assimilate this new knowledge, we found ourselves digging deeply into our toolkit for self education and practicing teaching each other new systems.

6.11 Compassion

EHRaS was created out of a desire to help others; a desire to better peoples' lives (especially the lives of people in need) guided our entire design process. From ensuring that the interface was as clear as possible to reduce distraction in high-stress medical environments to designing algorithms that won't stress burgeoning developing networks, compassion and awareness remained at the forefront.

Providing a tool to streamline the provision of one of our basest needs as humans makes us proud and honored.

Chapter 7

Conclusion

We're proud to present EHRaS as a completed, working web application requiring minimal customization to be field ready. Our testing framework quantitatively proves its quality, it incorporates novel features that make it a leader in degraded network operation among EHRs, and our simple data pathway is secure. EHRaS allows for authorized medical personnel to access and update treatment information for the most vulnerable and at-risk populations on the globe through an easy to use interface. It's extensible, standards compliant, and simple to deploy. We're thrilled to say that EHRaS is the first system of its kind that we are aware of on the market today, and especially excited to have a system working in practice, not just theory.

7.1 Strengths and Weaknesses

EHRaS' ultimate benefit in ad hoc settlements is that it bridges the gap from between typical discontinuous care (such as in refugee settlements where no records are kept at all[14]) and the beginnings of a tracked healthcare systems where doctors have at least the rudiments of a patient 'chart'. The ability to review previous treatments and assemble a patient's story is vital part of clinical care[15], and in that regard, EHRaS is an excellent start.

EHRaS is a lighweight alternative to other EHR solutions that carry prohibitive overhead in the field (such as server infrastructure or cost[16]). While this low overhead solution does lack comprehensive fields for all possibilities of care, it makes up for it in flexibility by allowing care providers to dynamically enter and format information on the fly, or modify source code for more concrete updates to data input and output.

EHRaS is suitable as a first-line deployment to an ad-hoc settlement; it provides patients with data-backed healthcare and doctors with (what will be for many, in refugee camps) a simple but powerful introduction to EHR.

Despite EHRaS' dramatically lower overhead than competing EHR solutions, it does still require the rudiments that any web application needs – reliable power, a physically secure server, and a network connection from the server to the doctors' devices. Obviously, in an environment where this is not available, EHRaS is not a suitable option until the settlement's infrastructure is sufficiently developed to support deployment.

EHRaS is also best suited for continuing care; data-driven treatment is only an option once a certain baseline of patient throughput and care quality is established. In newly settled camps, for example, doctors operating primarily in triage mode might find digital charting not only unhelpful, but harmful to their ability to treat dire health concerns rapidly and move onto the next patient.

Generally speaking, EHRaS' ideal usability is in an established camp with at least somewhat reliable infrastructure and a mature medical practice on-site. Outside of these bounds, EHRaS utility begins to diminish and doctors may prefer to fall back to zero record keeping. Ultimately, EHRaS exists to make healthcare safer and more accurate; if medical professionals in a given settlement believe it to be poorly suited, then it should not be used.

7.2 Lessons Learned

Throughout this project, we have learned many key lessons about collaboration. The importance of open and frequent communication was underlined for us as we were able to navigate issues and concerns as they came up. One issue we encountered was the difficulty we faced when trying to integrate our system with Ember for our frontend. After discussing the matter we settled on using more familiar technologies like Handlebars and jQuery in order to more efficiently and effectively move forward with our work.

Another lesson we learned was about the importance and the benefits of good organization. As the deadline for the Senior Design Conference approached, we were able to greatly streamline our production pace through the usage of a kanban board on Trello, an activity board web service. After establishing the board, we were able to efficiently manage the remaining tasks of our project.

Lastly, we learned about the importance of incorporating the useful skills of our teammates. In meetings and discussions, we openly discussed what improvements we could make upon our existing work. Max, for example, was able to utilize the usability skills he had gained from a upper division course which greatly benefited our Societal Lessons component of EHRaS.

Overall, EHRaS both opened our eyes to the specific needs of at-risk populations as well as stretched and exercised our technical and organizational, providing a challenging and rewarding

conclusion to over a year of work.

Bibliography

[1] T. U. R. Agency, "UNHCR Statistical Yearbook," tech. rep., 2015.

[2] L. Hancock, "CancerNet: More than PDQ on the net," *Database*, vol. 17, no. 1, p. 86, 1994.

[3] J. J. Moynihan and K. Norman, "Chin provides vital healthcare linkages," *Healthcare Financial Management*, vol. 48, no. 1, pp. 59–64, 1994.

[4] J. Bushelle-Edghill, J. Lee Brown, and S. Dong, "An examination of EHR implementation impacts on patient-flow," *Health Policy and Technology*, vol. 6, pp. 114–120, mar 2017.

[5] T. Heart, O. Ben-Assuli, and I. Shabtai, "A review of PHR, EMR and EHR integration: A more personalized healthcare and public health policy," *Health Policy and Technology*, vol. 6, pp. 20–25, mar 2017.

[6] Bradymiller, "OpenEMR Features," 2017.

[7] V. Singh, "Bamni Rodmap," 2017.

[8] US Department of Health & Human Services, "Summary of the HIPAA Security Rule."

[9] 3GPP, "GSM/EDGE Physical layer on the radio path; General description," 2016.

[10] K. Ali, "Introducing OpenCellular: An open source wireless access platform," 2016.

[11] Y. Maguire, "Building communications networks in the stratosphere," 2015.

[12] J. Kingsman, "express-heatlamp," 2017.

[13] Ivmartel, "DICOM Web Viewer," 2014.

[14] M. J. Israel and M. Werner, "Interview with Fr. Maria Joseph Israel," 2017.

[15] L. Varpio, J. Rashotte, K. Day, J. King, C. Kuziemsky, and A. Parush, "The EHR and building the patient's story: A qualitative investigation of how EHR use obstructs a vital clinical activity," *International Journal of Medical Informatics*, vol. 84, pp. 1019–1028, dec 2015.

[16] C. W. Day, "The Total Cost of EHR Ownership," *American School & University*, vol. 72, no. 6, pp. 48–49, 2000.