

OPENMRS METADATA SHARING SERVER: CONNECTING ACADEMIA AND HFOSS

JOSEPH WILDER, 2012

Submitted to the
Department of Computer Science,
Princeton University,
towards fulfillment of the requirements
of Undergraduate Independent Work

FINAL REPORT

May 2, 2012



Advisor: Dr. Robert Dondero
Reader: Professor Brian Kernighan

This paper represents my own work in accordance with University regulations.

Acknowledgements

This has been an up and down year. I would never have gotten this far without the patience, insight, frustration, persistence, disappointment, and resourcefulness of my advisor Dr. Robert Dondero and my mentor from the OpenMRS project Rafal Korytkowski. Thank you to Darius Jazayeri for the initial inspiration of working on HFOSS and introduction to OpenMRS. Thanks also to Piotr Bryk for his work developing the OpenMRS Metadata Sharing Server. Special thanks go to my friends Nathan Keyes '12 for hardware and moral support, to Erin Mills '13 for assisting with formatting of this report and believing in me even when I didn't, and to my parents, whose support from afar was greatly appreciated.

Contents

1	Background	1
1.1	OpenMRS and HFOSS	1
1.2	Metadata Sharing Module	4
2	Methodology	7
2.1	Getting to know an OSS Community and Distributed Development	7
2.2	Software Development Tools	8
2.3	Agile and Test Driven Development Software Processes	10
3	Design	12
3.1	Use Cases	12
3.2	Scenarios	13
3.2.1	Anonymous Users	13
3.2.2	Authenticated Regular Users	14
3.2.3	Authenticated Administrator Users	16
3.3	Software Requirements Specification (SRS)	17
3.4	Initial Approach	19
3.5	New Approach	22
4	Functionality	24
4.1	Needs of Users	24
4.2	Implementation Walkthrough	24
4.3	Testing	29
5	Evaluation	30
6	Conclusion	32
6.1	Future Work	32
6.2	Conclusion	33
A	Global OpenMRS Distribution	36
B	Initial Project Schedule	40
C	Example Metadata Package	41
C.1	header.xml	41
C.2	metadata.xml	43
D	IEEE SRS Standard	47
E	Database Schema	48

F Code	50
F.1 Setup	50
F.2 templates.php	51
F.3 utils.php	56
F.4 Functions	58
F.5 home.php	59
F.6 newaccount.php	61
F.7 loginPage.php	62
F.8 profilepage.php	63
F.9 upload.php	64
F.10 managePackages.php	65
F.11 manageUsers.php	67
F.12 search.php	70
F.13 download.php	71
F.14 addUser.php	73
F.15 login.php	75
F.16 uploader.php	76
F.17 peditor.php	79
F.18 ueditor.php	81
F.19 deletePackage.php	84
F.20 removeUser.php	86
F.21 Exit pages	87
F.22 proc.php	88
F.23 Javascript	92

List of Figures

3.4.1	GUI mockup for uploading a package to MDSS.	20
3.4.2	GUI mockup for browsing and downloading packages from MDSS.	21
4.2.1	Screenshot of the OpenMRS MDSS homepage.	26
4.2.2	Screenshot of the OpenMRS MDSS user registration page.	27
4.2.3	Screenshot of the OpenMRS MDSS profile page.	27
4.2.4	Screenshot of the OpenMRS MDSS upload page.	28
4.2.5	Screenshot of the OpenMRS MDSS homepage.	28
A.0.1	A map of current OpenMRS clinical and research sites	36
A.0.2	A table listing the current OpenMRS clinical and research sites	39
B.0.1	The initial project management schedule.	40

Abstract

Implementation of software to gain hands-on experience with OSS communities and software engineering processes in support of the OpenMRS project. Deployments of the open source medical records software application OpenMRS can create and export/import metadata but do not have a good centralized way to share it. The Metadata Sharing Server (MDSS) was conceived as an answer to that obscurity and inefficient duplication of effort. MDSS can be thought of as an OpenMRS specific type of package manager way for community members from over 26 countries to pool their metadata resources in support of advancing medical care in situations where enterprise level software solutions are infeasible or undesirable. Major lessons involved getting accustomed to distributed collaboration, navigating the multitude of tools in use in OSS projects, and introduction to Agile and test driven design methodologies. A thesis project for academic requirements and ‘...In the service of all nations’.

Chapter 1

Background

OpenMRS, as the acronym suggests, is an Open Source Medical Records Software application. The project technically falls into a subset of open source software projects known as humanitarian free open source software (HFOSS, <http://hfoss.org>). HFOSS and software engineering is not currently a part of the Princeton University undergraduate computer science curriculum. Consequently, working with OpenMRS involved not just design and implementation of software but also a large and novel component of exposure to and learning about open source software communities, distributed software development and collaboration, an alphabet soup of OSS tools, and Agile software engineering processes. The core idea and emphasis which led to seeking out a project in HFOSS to complete independent work is to seek out dynamic work that won't end up gathering dust on a shelf. Instead, the ideal is a real world project to gain experience, which positively impacts the lives of real people, and bridges between academia in the university and software engineering in the wild. A project that would truly live up to the broad scope of Princeton's motto 'Princeton...in the service of all nations.'

1.1 OpenMRS and HFOSS

The OpenMRS project is a multi-institution non-profit collaborative software and humanitarian project open to all who are interested in contributing, including many organizations such as international and government aid groups, NGO's, and for-profit and non-profit corporations, as well as the many individual contributors [1]. The project is led by the Regenstrief Institute, a world-renowned leader in medical informatics research, and Partners In Health, a Boston-based philanthropic organization with a focus on improving the lives of

underprivileged people worldwide through health care service and advocacy. The mission of OpenMRS is to improve health care delivery in resource-constrained environments by coordinating a global community that creates a robust, scalable, user-driven, open source medical record system platform. OpenMRS is a software platform and application which enables design of a customized medical records system. No programming knowledge is necessary to use OpenMRS (although medical and systems domain knowledge is required). OpenMRS is a common platform upon which medical informatics efforts in developing countries can be built. The OpenMRS software is free to download and released under the GNU Public License (GPL). HFOSS projects like OpenMRS include elements of both free software (FS) and open source software (OSS). Generally, discussions about the application of these terms have become almost a fundamental philosophical discourse and reach far beyond just developing software.

Implicitly, it reflects the choice between two different fundamental self-perceptions, aligned with different life-styles and political conceptions of the world.

According to members of the FS community, the intended meaning of "Free Software" is "software that gives the user certain freedoms", but the term also invites to the unintended interpretation as "software you can get for zero price" (cf. www.gnu.org/philosophy/free-software-for-freedom.html). These freedoms contain ethical issues, aspects of responsibilities and of convenience. [2]

On the other hand,

Members of the OSS community define "Open Source Software" as software that allows everybody to have a look at its source code and stress the practical benefits of such software, while aspects of freedom are rather neglected in the definition. "Open Source Software" contains a broader variety of software than it is allowed by the term "Free Software", it comprises free software as well as semi-free software and even certain proprietary programs. [2]

OpenMRS, then, is truly HFOSS because its source code is open and has practical benefits, primarily that it is available for zero cost and it gives developing or underprivileged people more freedoms in access to modern healthcare. Since being created in 2004 OpenMRS has spread far and wide through the work of its collaborative community. See Appendix A for more information on OpenMRS research and clinical implementation sites in more than 26 countries worldwide.

OpenMRS is written in Java and HTML, and is designed to be cross platform. The application handles a variety of use cases from the level of national health infrastructure to a local field clinic. To be adaptable to any context that might arise, individual customized setups of OpenMRS for specific clinical contexts are created. These unique setups are known as implementations, and are identified by their implementation ID number. Members of the OpenMRS community who specialize in creating new implementations are referred to as implementers. These and other community groups are further discussed in Section 2.1. The platform itself is a data model surrounded by an abstracting API layer and then paired with a web-based app utilizing a client-server design to provide access to the electronic medical records. The OpenMRS data model is in practice treated as a black box, but is based around the idea of a central 'concept dictionary' which handles operations on information codified as 'concepts' independent of the type or any particulars of the information collected and operated on. This model is descended from the Regenstrief Institute EMRS model first developed by the eponymous Indianapolis-based international biomedical informatics and healthcare research organization in 1994 [1]. Unique implementation configurations are enabled not just by being able to handle operations on diverse concepts and running in the JVM over multiple hardwares, but also by the ability to include a variety of modules and metadata. Modules are community developed add-ons to the application containing additional libraries and functionality for particular tasks or situations. The word metadata refers a whole class of data including members in multiple categories such as: a) concepts, b) locations, c) encounter types, d) user roles, e) programs, f) new drug regimens, g) treatment

plans, h) language packs, or i) clinical encounter forms. Metadata do not include individual electronic patient records. Some metadata are pre-installed as part of configuring an implementation, but more often metadata are created by users of an implementation in the course of their work. Metadata are also created and improved on by users of the OpenMRS development community at large. To contribute, collaborate, and avoid redundancy those users then often want to share their metadata with other members of the community who might be tackling a similar challenge or situation. Currently the best mechanism that exists for sharing metadata is the Metadata Sharing Module.

1.2 Metadata Sharing Module

The Metadata Sharing (MDS) Module is an add-on module to OpenMRS that facilitates exporting and importing of metadata. It was initially implemented in two phases as Google Summer of Code projects in 2010 and 2011 and is currently in beta version 0.10.x. The module, when installed, is accessed through the OpenMRS application's Administration panel. Users can choose to create a new metadata package containing a selection of any of the locally installed metadata and their dependencies. Users can also manage any of their previously created packages or import a new package sourced from some other channel. This system is effective at these tasks but has limited spread and networking as a byproduct of its individual and distributed nature which often leads to frustration or duplication of efforts when users are not aware of or do not have access to the previous work of other metadata developers, implementers, and users. The next paragraph will go over more detailed usage of the MDS module operations.

When choosing to create a new metadata package with the MDS module, the user is prompted to select which metadata from all available metadata in the current implementation should be added to the package. Users then give the package a name and description and the module packs the chosen metadata, as well as any concept or other metadata dependencies of the chosen metadata, into an XML file called `metadata.xml`. The module

then creates a header file for the package containing the package name, package version, package description, package uuid, creation date, OpenMRS version, and group. See appendix E for example metadata package XML files. These two XML files together are then bundled into a zip archive which is the metadata package. All exported packages are then published and available for download through the web-based app as a REST service from a URL of this form: <http://%yourhost%:8080/openmrs/ws/rest/metadatasharing/package/%UIDOFPACKAGE%/%VERSION/download>. Managing previously exported packages simply entails unpublishing or republishing those packages to their URLs. Importing a package requires having either the zip archive of a metadata package or having an active published URL of the package that you would like to download. When importing it is possible to choose different levels of trust which configure whether or not local metadata and concepts will be overwritten with imported metadata and concepts. A warning will be displayed if a user tries to import a package exported with a different version of OpenMRS or if a newer version of the package being imported is already installed. A list of all metadata in the package is then shown and options are given for each piece including: a) create new, b) skip if possible, c) keep mine, and d) overwrite. After that the metadata is imported. Note that uuids and conceptids stay constant, if overwriting occurs only the content is overwritten. Managing already imported packages gives an option to check for a new update of a package from a published URL or to ‘subscribe’ to a package and set a timer for how often the MDS module will automatically check the URL for an updated version of that package. The obvious limitations of this functionality lie in accessing metadata to import and include the necessity of either somehow having a package to start with or having a working internet connection and knowing the URL of any possible exporters of the package who are currently actively publishing it.

The OpenMRS MDSS then is envisioned as an assistant to the MDS module consisting of a web application acting as a central repository of metadata packages exported from different users’ MDS modules. The server will centralize, publicize, and standardize packages

of metadata so that all OpenMRS implementers, developers, and users can easily access or peruse all available metadata packages created by the community globally without having to find out that a package exists, track down someone who has the most recent version of a package, and then receive a copy of the package or the published URL of the package from the exporter in a different channel. This will broaden awareness of the available metadata, reduce metadata creation redundancy, better communicate community metadata needs, and allow easier tracking of multiple versions of a given package. In addition, exporters will no longer need to continuously expose their base application to publish packages and may use less of their internet bandwidth for a wider level of distribution.

Chapter 2

Methodology

2.1 Getting to know an OSS Community and Distributed Development

A large part of this project is getting real world experience and learning about how OSS projects and communities operate. In the OpenMRS organization, community members who work on development are divided into three main groups. The first is a small group of core developers who are intimately familiar with the main project, devote much of their time to it, and are continuously involved in discussing, designing, and implementing new changes and updates for it. This group is also the primary source of decisions about the direction of OpenMRS and setting the road map and milestones, although those decisions are also made in consultation with the community at large and other important players. A larger group also exists of regular developers who are less involved in the project but still active in the code base. Many times regular developers may be professionals volunteering their spare time or students trying to fulfill university requirements. Regular developers often work only for a set season or shorter period of time or may focus only on one ‘pet project’, feature set, module, or aspect of the software. In some organizations core developers or regular developers can be paid for their work even if the software they work on is free, but this is not the case with OpenMRS. A third group of developers are casual developers. These developers often take on introductory tasks or seek heavily the mentorship of more experienced developers. Casual developers tend to be relatively short lived, either advancing to be a regular developer or transiently moving on to other projects. Some of the more long lived casual developers are active members of other parts of the OpenMRS community but simply don’t have a

programming background or interest in software development beyond what they need to use OpenMRS. Another important user group within the OpenMRS community are the implementers. These advanced users take the application and set it up, adapt it to particular local systems and needs, train other users (especially non-technical users), and maintain it in various clinical and hospital settings around the world. The implementer and developer groups each have their own email mailing lists and weekly conference calls, but they are not mutually exclusive and many of the best development suggestions come from users who are in both groups.

OSS development projects that are distributed especially internationally like OpenMRS are referred to as DOSSD projects. Because the OpenMRS project is open to all who are interested in contributing, including many organizations such as international and government aid groups, NGO's, and for-profit and non-profit corporations, as well as the many individual contributors, the interested parties are highly scattered geographically and internationally over the whole world. OpenMRS is then a prime example of an DOSSD project and a distributed OSSD community. When a collaborative community is so dispersed many different methods of communication must be utilized to enable software development and collaboration.

2.2 Software Development Tools

Tools which are themselves OSS are preferred for enabling collaboration on OSS projects like OpenMRS and many software companies will give special licenses to OSS projects and their developers. Companies which have given OpenMRS special OSS licenses to their tools include Atlassian, AquaFold, Balsamiq Studios LLC, Bitrock, Blueberry Software, JetBrains, Inc., Napkee Labs, Yourkit, and Zero Turnaround. Many of the tools used (both licensed and OSS) are new to me coming from the experience of largely individualistic academic classwork but their purposes are common knowledge to anyone who has worked in software development, especially distributed software shops.

The primary tool that aids distributed development is source code control. OpenMRS uses Subversion to keep track of their repository and code edits, patches, branches, and permissions. They suggest that users interact with Subversion through various pre-tested IDE plug-ins, graphical interfaces, and command line tools such as Subclipse and Tortoise. Use of other clients for interacting with the Subversion repository is up to the discretion of the individual developer but may not be supported. At the same time as this project was taking place investigations were being conducted by the community to begin the process of transitioning to use Git for source control in place of Subversion. The first code branches to be moved to Git were the main application and certain key ‘core’ modules adopted for maintenance by the core developers. Git is preferred to Subversion, as long as the transition is manageable, because it simplifies permissions and access control. In Subversion, permissions are time consuming because each and every developer must be added to the user class which has permissions to interact with the particular branch they want to work on. In Git, only a few very active core developers need permission to push code changes to trunk of core and everyone else can easily fork a branch with no permissions necessary and then submit a pull request to the core developers when they are done working. This setup provides a lower barrier to entry for new developers to try out their ideas quickly. Git is also preferred because as a distributed repository it has better support for ‘commit’ operations in conditions with unstable or intermittent internet connections, as is typical in developing countries. Subversion is preferred because it is much simpler to learn at first and has the momentum of already being established in use. It is likely that OpenMRS will soon follow the growing trend of OSS projects moving to Git, although the effort to make the switch, update all the documentation, and educate the users will not be insignificant.

After source control in the tool importance hierarchy comes the primary issue tracking tool JIRA. JIRA is useful for keeping track of all phases of software development process through entering, assigning, and taking actions on tickets that track through the necessary phases as well as providing a platform for discussions and code reviews. JIRA ties in with

Subversion and Git so that committed code changes are hotlinked to the tickets they pertain to. Additional information is also generally referenced from articles on the OpenMRS wiki. Dashboards can then be created which track progress on a given class of tickets, on tickets related to a given section of code, or on tickets labeled belonging to a given sprint.

The final most important type of tool in DOSSD is a whole class of communication technologies for collaboration. When developers are in different timezones around the world the number one communication tool is email but OpenMRS also successfully utilizes their documentation and discussion wiki, their own IRC channel, Skype, and other video and teleconferencing applications such as Youtube, Adobe Connect, Breeze, and Etherpad. These tools allow for a variety of meetings and more or less in-person, mostly synchronous communications. Daily scrum meetings from the core developers involved in any active sprints are a highly visible example of these technologies in practice. OpenMRS University sessions aim to teach skills or answer questions of new users and developers interested in getting involved. Weekly design calls and implementers calls serve as forums of discussion for their respective groups for decisions that are better handled in person instead of over endless email threads.

2.3 Agile and Test Driven Development Software Processes

OpenMRS favors a modern style of software engineering process which is similar to Agile Programming and Test Driven Development. This class of processes are often called extreme programming and rely on rapid iteration and flexibility in the process. The key principles of this approach include: a) stakeholder involvement, b) incremental delivery of intermediate products, c) expectation of change, d) fluid requirements, and e) active emphasis on promoting simplicity and reducing coupling ([3] chapter 17.1). Each developer is free to follow their own individual preferred process, but the organization as a whole is very flexibility-focused. The sprint process OpenMRS uses, by which all effort is concentrated on a set of well-defined goals for a short period of time, is very much a hallmark of these kinds of processes. Daily

scrum meetings by the core developers which emphasize quick reporting of what has been done in the last day, what work is being done now, and what project blockers if any are being waited on, is another example of extreme programming in OpenMRS. The OpenMRS application is built such that any changed code when rebuilt is run against extensive Unit Tests through Jrebel. These tests were written before the classes that they test and help make sure that unexpected breakages and/or work stoppages due to committed code with bugs are as rare as possible. OpenMRS, also as part of their Agile-like process, (at least for the core developers) ascribes to a philosophy of being as fluid in design as possible, as seen in their use of many mock-ups and rapid prototypes. These iterative and less-defined processes are quite an experience and very different from the code development experience available in my undergraduate curriculum.

Chapter 3

Design

Beyond learning about the OSS community, its many tools, and software engineering and Agile methodologies, the primary goal of the project was to develop software for the MDSS. The first step in building the software was to define the goal. The definition process for this project started with text-based use-cases (as opposed to use-case diagrams in UML), which were further explored in individual scenarios, which in turn were specified by eliciting criteria. This was followed by construction of requirements, and finally the proper tools were selected. This process was careful to take into account and accurately represent the goals, domain knowledge, and stakeholders of the MDSS software ([4] chapter 2).

3.1 Use Cases

Primary use cases were the first step in defining the MDSS. There are three main actor classes in these use cases: 1) an anonymous user, 2) an authenticated regular user, and 3) an authenticated administrator user. These user classes are listed here hierarchically in that each class shares all the use cases of the classes that are less access restricted. Anonymous users have three main uses of the MDSS: browsing available packages, downloading a package, and registering as a new user. They can also transcend actor classes by logging into an existing account. Regular users then share all of those use cases. There are four additional uses available to authenticated regular users: 1) uploading a package, 2) interacting with their own uploaded packages, 3) editing their own user account, and 4) logging out . Authenticated administrator users then share all of the use cases of anonymous and authenticated regular users and, in addition, have two use cases of their own. Only authenticated administrator users can interact with all users' uploaded packages and accounts. Each of these use cases

encapsulates a set of different actions, so after describing the MDSS use cases I proceeded to tease out individual interaction threads ([3] chapter 7).

Next, scenarios were constructed to describe the individual actions possible in each use case. These scenarios, given below grouped by the use case to which they belong, capture another, deeper level of details about the system but are still relatively informal in order to be easy to relate to and explore. These scenarios lay out the key capabilities of the software and the desired patterns of interaction with the three different actor classes, as well as what can go wrong in each of the situations and how that might be handled ([4] Knowledge Area 2).

3.2 Scenarios

3.2.1 Anonymous Users

Initial assumption: User with a properly functioning OpenMRS implementation, MDS module, an internet connection and browser, and the address of a MDSS.

Browsing available packages A user navigates to a page with a list of available uploaded packages. The user can view multiple types of data about each package, sort the list, and search through the list.

Things that can go wrong: If search returns no results the user is notified and continues browsing.

Downloading a package When looking at a list of packages a user can click on a download link and be prompted for a path to save the selected package, which then downloads.

Things that can go wrong:: If a user tries to open a download link to an invalid or no longer available package they are notified that their download link was invalid and there is no such package, then the user is returned to browsing. If the selected location is invalid or the connection is interrupted the download will fail but the MDSS is not responsible for handling these failures

Registering as a new user account A user navigates to a registration form and enters a username, email address, and password. The user is registered, notified of successful registration, and then prompted to try logging in.

Things that can go wrong:: If a user doesn't enter any of the three pieces of required info they are prompted to enter the required info and returned to the form. If a user enters a username or email address already registered to another user they are notified that those credentials are already in use and returned to the form.

Logging into an existing account A user navigates to a login form and enters credentials of an existing account, is logged in, welcomed, and shown the account details of their account.

Things that can go wrong:: If the account entered does not exist or the password entered is incorrect the user is notified generically that they typed something wrong and returned to the login form.

3.2.2 Authenticated Regular Users

Uploading a package An authenticated user can navigate to an upload form where they are prompted to choose a file to upload. The package is then uploaded and the user provides details about that package for display.

Things that can go wrong:: If the file does not have the correct characteristics the file will fail to upload. The user will be notified and returned to the upload form. If no details are available the upload will fail, the user will be notified and returned to the upload form.

Interacting with own uploaded packages An authenticated user can see a list of all their own uploaded packages. They will be able to navigate to a form to edit the entered details any of those packages. The details of a package that are editable will

be displayed and when finished editing a user will confirm saving any changes. They will also be able to remove those packages from the server.

Things that can go wrong:: If a user has not successfully uploaded any packages the list will notified to be empty. If a user begins editing the details of a package but does not finish or confirm the edits they should not be saved. A user may edit the unique details of a package to be the same as another package already shared or a user may edit a field (such as subscription url) to a value that is not valid for that field type. In those cases the user should be notified that they have made an illegal/restricted edit and be returned to their uploaded packages list.

Editing with their own account An authenticated user can navigate to an editing form for the details of their user account. The current details which are editable should be displayed. The user can then edit them and when finished confirm and submit the changes and then be redirected to their user page.

Things that can go wrong:: A user may begin editing their details and stop in the middle, in that case the edits should not be saved. A user may edit their unique details to be the same as another registered user or a user may edit a field (such as email) to a value that is not valid for that field type. In those cases the user should be notified that they have made an illegal/restricted edit and be returned to their user page.

Logging out An authenticated user can, from any page on the site, choose a log out option which will terminate their session and return them to the home page as an anonymous user.

Things that can go wrong:: If no user is found to be logged in when the logout option is activated it will silently redirect to the home page with no other changes.

3.2.3 Authenticated Administrator Users

Interact with all users' uploaded packages An authenticated administrator user can navigate to a listing of all uploaded packages. They should be able to sort these packages by different criteria and search through these packages. For any package in that list, an authenticated administrator user will be able to navigate to a form for editing the details of that package or removing that package from the server in the same way that any authenticated user could for their own uploaded packages in Section ?? above.

Things that can go wrong:: The same things that could go wrong for individual authenticated users interacting with their own uploaded packages above, except that the redirect will be to the administrator's listing of all uploaded packages instead of to the authenticated user's own profile page. An administrator can attempt to edit their own account details to remove their admin status. This operation is unusual but should succeed because at the time of the transaction they have permission to make that edit.

Interact with all users' accounts An authenticated administrator user can navigate to a listing of all registered users. For any user listed they can navigate to a form for editing the details of that user's account. They can also remove any user from the server.

Things that can go wrong:: The same things that can go wrong with a user editing their own account details above except that the redirect should be to the administrator's list of all users instead of to the user's own account details page. An administrator tries to remove themselves as a user from the server. This is an unusual case but should succeed because at the time of the operation the administrator has permission to do so.

3.3 Software Requirements Specification (SRS)

The next step in defining the software is generating a software requirements specification (SRS) document from the use cases, scenarios, and more detailed criteria. The SRS is an official statement of what the developer (in this case, me) should implement. I wanted to define a more formal SRS than most Agile processes would use (they even go so far as to recommend having a set of cards which can be constantly reorganized by the customer according to their changing priorities) because I saw meeting my requirements contract as one of my primary criteria for success. A more standard formal SRS specification is IEEE/ANSI 830-1998 (See Appendix D). This specification on the other end of the spectrum has way too much detail for this type of project, but it served as an example of a formal document for requirements. The type of SRS is dependent on the type of system being developed and the development process being used. In my case I am part a very small team in a very iterative process on a small scale system. Therefore I can have a more flexible, less detailed SRS and count on resolving any ambiguities when I reach them in implementation ([3] chapter 6).

Server Generics

1. The server will run on any operating system.
2. The server will use PHP/HTML as its primary language.
3. The server may also use javascript as necessary.
4. The server may also use CSS as necessary.
5. The server will interact with a MySQL database.

Uploading

1. The server will accept uploaded metadata package files which are of MIME types application/zip, application/x-zip-compressed, multipart/x-zip, and application/s-compressed.
2. The server will accept uploaded metadata package files which are up to 1MB in size.
3. The server will not accept uploaded php files (these are dangerous security hazards).
4. The server will rename any uploaded files to a random name, for security reasons.
5. The server will be able to save uploaded files outside of the webserver root, for security reasons.
6. The server will check for name, OpenMRS version, and package version collisions before accepting an uploaded package.

7. The server will verify that the header.xml and metadata.xml files are properly-formed XML before accepting a new upload.
8. The server will index package headers for information including name, uuid, description, publication date, publication url, OpenMRS version, and package version.

Editing Packages

1. Users will be able to edit the name and description details of their own uploaded packages.
2. Administrator users will be able to edit the name and description details of other users' uploaded packages.
3. Administrator users will be able to remove any uploaded package from the server.
4. The server will not allow package detail edits which cause a name, OpenMRS version, and package version collision with an already shared package.
5. The server will show a confirmation or failure message for attempted package detail edits.
6. The server will persist package detail edits, not only in the database, but also in the header.xml file itself.

Passwords

1. All passwords will be stored in the database, hashed for security.
2. All hashed passwords will be salted to guard against rainbow table attacks.
3. All password must be at least 8 characters in length.

Browsing shared packages

1. The MDSS will list packages available including information about name, description, author, date uploaded, package version, OpenMRS version, and download count.
2. The listed packages will be orderable by popularity (download count), recency of upload, author, OpenMRS version, and name.
3. The listed packages will be searchable by name, description, or author.
4. All displayed packages will have corresponding download links.

User accounts

1. New users will be able to register with the site.
2. Registration will collect username, email address, and password.
3. No two users will be able to register the same email address.
4. Users will be able to use their email and password credentials to authenticate themselves and log in.
5. The server will show an error message for failed authentications.
6. All pages will contain a link to log in.

7. There will be at least two user classes with different privileges regular user and administrator user.
8. All pages visible to authenticated users will contain a link to log out.
9. Users will be able to edit their own account details, including name, email, and password.
10. The server will give an error message and not save an edit if it causes a name or email collision with another registered user.
11. Administrator users can edit the account details of other users including name, email, password, and user class.
12. Administrator users can reset the password of any user to a default password.
13. Administrator users can remove any user from the server.
14. If a user is completely removed by an administrator, all of that user's uploaded packages are also removed.

3.4 Initial Approach

One of the initial tasks for any software development project management plan is to plan out a schedule. This was very difficult to do on this project. I planned out an initial schedule which listed large milestones such as Learning Tasks, Version 1.0: Implement Server, and Feature Set 1. The slightly improved but still entirely naïve schedule is included in Appendix B for your amusement and chagrin. This projection was in terms of naïve feature wishes, but had no realistic concept of what it would take to reach those and in which order they might necessarily be developed. Beyond the initial idea and a really good story, I had difficulty anticipating what kinds of steps might be necessary in the process of developing my first server. The target project schedule was one that involved very granular tasks to the level of days and keeping track of projected days to completion of each step and actual days to completion to track project overrun. The initial idea was to develop in Java and this lead to an initial schedule with a large portion of time spent on learning tasks about Java and the other tools necessary for Java servers. I also spent a lot of time getting OpenMRS trunk checked out and set up for development in Eclipse and running locally on Jetty. My third major task was beginning to create GUI mock-ups for the MDSS in Balsamiq which

is integrated with JIRA. The OpenMRS principles guiding gui design include: simplicity, unclutteredness, self-explanation, and least number of clicks for the most common tasks. See Figures 3.4.1 and 3.4.2 for examples of some of the resulting mock-ups. Progress on the Java front was very slow on my own and repeatedly overshot estimates up until mid-December. At that point we tried implementing a new schedule where there was no fixed overall schedule, appropriate goals would be created based on progress and success was determined on a weekly basis of tickets or tasks, skype calls, and call summary notes. This system got some work going, but it was slow and difficult and often the weekly evaluations backfired into ignoring everything and prevented more attention from being paid to the tasks at hand especially during university breaks and exam periods early in the year.

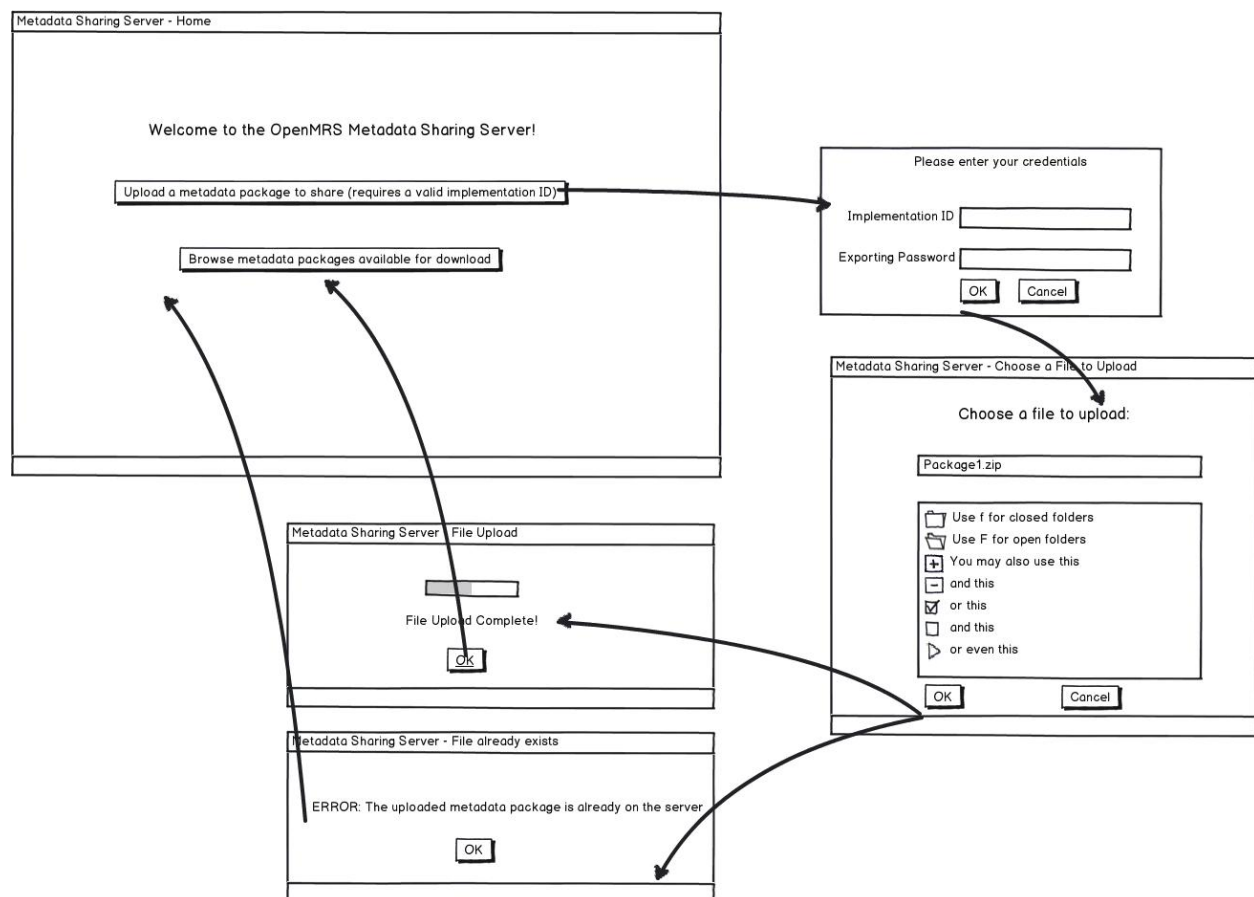


Figure 3.4.1: GUI mockup for uploading a package to MDSS.

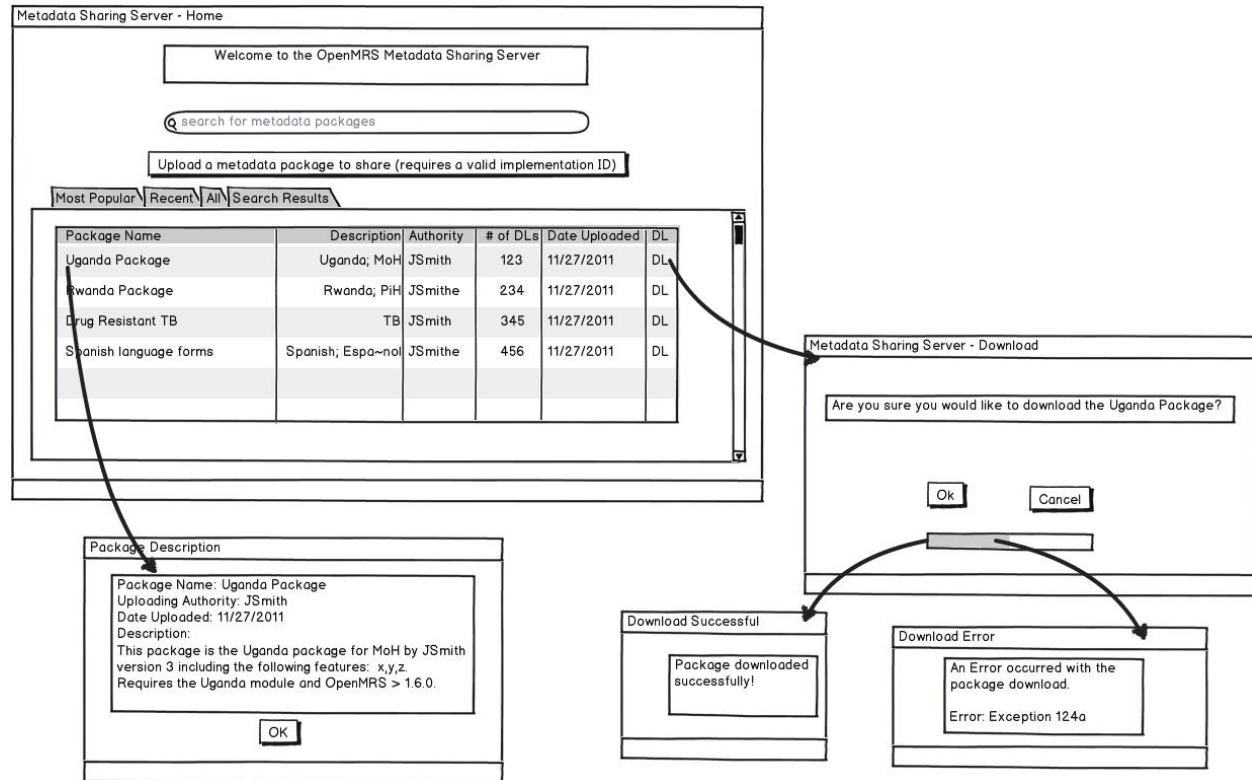


Figure 3.4.2: GUI mockup for browsing and downloading packages from MDSS.

In the end the Java approach turned out to be too complicated with its multitude of tools (Eclipse, Maven, AppFuse, Spring, Jetty, Hibernate, etc.) that needed to be configured without getting familiar with their particulars and work along the way but were incidental to the project at hand. Eclipse should be familiar to any developer as one of the premier IDEs for working in Java. Maven is a dependency repository manager and building tool that is very helpful when the projects get as large as OpenMRS and depend on many different resources to all work together. Jetty is the lightweight java based HTTP client/server that is bundled in Maven. AppFuse is quickstart scaffolding generator which initiates a sample project and all of the structure in place in order to use Spring and Hibernate. Spring is a MVC framework for building web applications. Hibernate is an ORM library for Java so that persistence can be done regardless of the underlying database setup. The hardest part of using so many tools has been that the expectation is not to get to know each tool fully but instead to get each configured right so that they contribute to development of

the end goal with minimal intervention on the part of the developer. Each tool is not to be learned per se but rather picked up enough to quickly do what needs done always with an eye toward where the project needs to be going. I am not familiar with this approach coming out of the academic context where things are taught and learned thoroughly. This approach is necessitated by the scale of the project complexity, rapidly changing nature of Agile development, and the time frame of the proposed project such that everything should not be written from low level but instead recycled, and utilizing freely available tools is absolutely necessary.

3.5 New Approach

The low level of collaboration on my part in the project at this time made navigating these tools independently impossible and meant that a different approach was needed. Therefore, a third of the way through spring semester a new approach was hatched to use PHP for development which is more comfortable, very well documented, and requires much fewer supporting technologies although it doesn't integrate as well with the OpenMRS platform itself. The new focus was on getting functionality first as quickly as possible and then getting minimally to the SRS however possible. The new approach needed new tools with a target on achieving functionality. For the requirement that the MDSS be fully portable the initial proposal to write the server in Java made a lot of sense because it could run above the hardware in the JVM. When I switched tracks to focus on PHP that meant finding a way to deploy my server cross platform. The solution that I found was to develop using XAMPP. XAMPP is a free cross platform Apache, MySQL, PHP, and Perl webserver stack. The server side code is written in PHP/HTML and interacts with a MySQL database (see Appendix E for database schema). The decision was made to do a majority of the operation on the server side for security but there will also be some client side operations. For achieving functionality on the client side the MDSS will use Javascript and a smattering of CSS. Some Javascript tasks such as form validation are simple enough to code directly, but for others such as

search functionality, Ajax requests for table pagination, and tabbed content presentation, a Javascript library will be necessary. The Javascript library for this project will be JQuery with some assistance from JQuery UI. There was also an attempt to separate out code for logic from code for presentation (see Appendix F for all PHP code). The MDSS is written first in English just like OpenMRS but later there may be opportunities to translate the interface. The database is collated in `utf_8_general` so that packages written in non-Roman alphabets can be supported for upload and hosting. Security is a large aspect of design of the MDSS. Because there are authenticated and non-authenticated users care was taken that all pages behave gracefully even if accessed directly by a non-authenticated user. Several security best practices are integrated in the design of the MDSS. All inputs that enter MySQL queries are escaped. Passwords are hashed, not stored in plain text, and those passwords are first salted against rainbow table attacks on a stolen database. File uploads are configured to be stored outside the webserver root, have a maximum file size, a mime type whitelist, and a file extension blacklist. XML files in packages are checked for proper form. In order to progress in more gradual steps even when I could not predict with fine enough granularity what tasks would need doing throughout the rest of the project I made renewed use of the JIRA ticket system. In this case Rafal, my mentor from OpenMRS, would make tickets for tasks he knew I would need to complete next and then I committed code to the repository linking each revision to a particular ticket. After several tickets were complete a code review would be started for all the code checked in during those tickets. The code review system in JIRA makes use of Crucible and FishEye to diff, comment, and flag changes. In all 16 tickets in JIRA were completed for this project although commits outnumber them 5 to 1. Other metrics used included: 1) three code reviews completed, 2) around 2000 lines of code written (growing and then shrinking as I factored out more common functionality; code is included in Appendix F), and 3) code quality measures which will be discussed in Section 4.3.

Chapter 4

Functionality

4.1 Needs of Users

The minimal needs of the users include the following:

1. Users need to be able to upload packages via the website with the ability to enter text fields such as name, description, and date published, which will describe the package. Users need to authenticate themselves in order to do this.
2. Users need to be able to (without authenticating themselves) browse a display of available uploaded packages including information about the packages' a) name, b) description, c) authority, d) date published, e) download count, and f) download link.
3. Users need to be able to filter browsed packages by id, name, description, and authority with a single search field.
4. Users need to be able to see the top 10 downloaded packages without logging in.
5. Users need to be able to register as a new user, edit their user details (name, e-mail, password), and display/edit/remove their uploaded packages.
6. Some users need the ability to log in as an administrator and do the following: edit package details, remove any or all packages, remove users, and reset passwords.

4.2 Implementation Walkthrough

Broadly, the system works based on PHP scripts (see Appendix F) which interact with a MySQL database (see Appendix E) and generate HTML and Javascript pages which take input from users and hand it off to other php scripts to operate on.

There are five main and two restricted display pages in the MDSS. They all start with the MDSS header and end with the MDSS footer. The home page has navigation links, uploaded

packages search box and results area, and tabbed displays of tables of uploaded packages. The register page has inputs for submitting a name, email, and password to register and will indicate that they are required if any of them are submitted empty. The login page has space to input an already-registered email and password combination followed by navigation links. The upload page has navigation links and a file selection dialog with an upload button. The profile page displays navigation links, account details and an edit link, and tabs displaying that user's uploaded packages. There are also the 'edit user' details and 'edit package' details dialogs which are subsets of the administrators-only 'Manage Users' and 'Manage Packages' pages, which have navigation links and list all records of their respective types with editing links. The site was implemented to match the requirements, because those requirements were constructed from the minimum user needs.

Without authenticating, users can browse the available packages on the home page in two ways: by searching for a name, description, or author through the search box, or by selecting one of the preloaded tabs for organized results. The initial tabs are: **Newest** (top 10 by upload date, not publishing date), **Most Popular** (top 10 by number of downloads), **Alphabetical** (all packages organized by name), **By Author** (all packages organized by author), and **By Release** (all packages by OpenMRS version). Each of these areas also shows columns for other info including a) name, b) description, c) authority, d) date published, e) package version, f) OpenMRS version, g) download counter, and h) download link. (See Figure 4.2.1). New users can register by giving a name, email, and password. Users can register a new account so long as no other user has previously registered the same name or user the same email address. (See Figure 4.2.2). Users can login to authenticate. When logging in, users are redirected to their profile page. From a user's profile page, they can a) see and edit their account details including name, email, and password, b) see any packages they have uploaded, c) download packages, d) edit name and descriptions of packages (which are persisted in the header.xml file), or e) remove those packages. (See Figure 4.2.3). When logged in, users can upload new packages. The header.xml file is parsed for the package's

data, including name, description, publication date, package version, and OpenMRS version. (See Figure 4.2.4). Users who are in the admin user class will also have additional navigation links to ‘Manage Packages’ and ‘Manage Users’. In ‘Manage Packages’ admin users can see all uploaded packages and download, edit name and description, or remove them. These changes are persisted in the header.xml file. In ‘Manage Users’ admin users can a) see all registered users, b) reset their password to a default of ‘MDSSUser-name’, c) edit their name, email, password, or class directly, or d) remove them. (See Figure 4.2.5). If an admin removes a user, all their uploaded packages are also removed.

Logged in users should log out when finished.



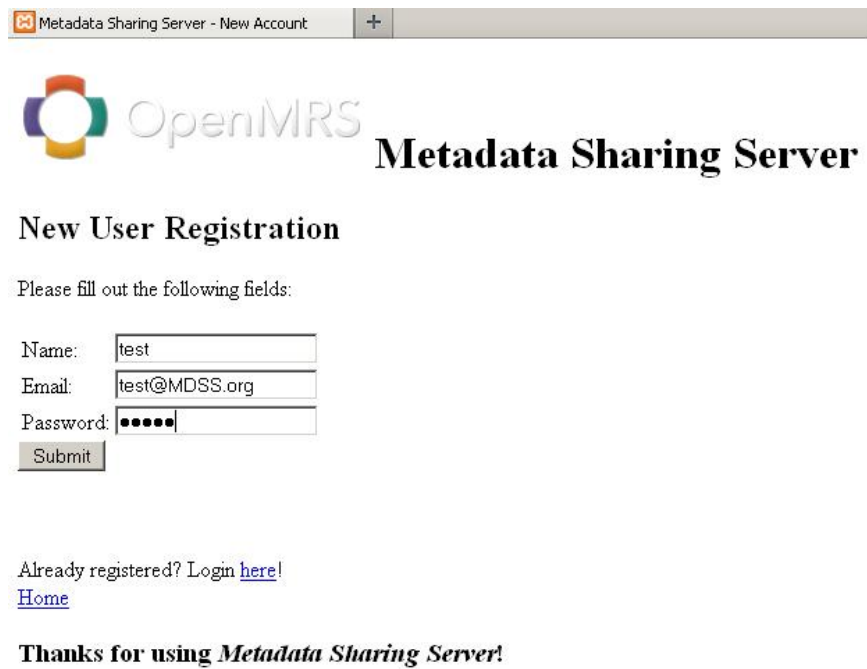
Home

By Name By Description By Author


Uploaded Metadata Packages:

Newest	Most Popular	Alphabetical	By Author	By Release			
name	description	authority	date published	version	OpenMRS version	downloads	actions
Really Long Name of the Coolest Metadata Package Ever!	Test package 3 of 4.	a	2011-12-02	1	1.9.0-SNAPSHOT	3	DL
Drug Resistant TB Plans	Test package 4 of 4.	test	2011-12-02	1	1.9.0-SNAPSHOT	2	DL
HTML	Test package 1 of 4.	admin	2011-12-02	1	1.9.0-SNAPSHOT	1	DL
Espanol	Test package 2 of 4.	Bob	2011-12-02	1	1.9.0-SNAPSHOT	1	DL

Figure 4.2.1: Screenshot of the OpenMRS MDSS homepage.



Metadata Sharing Server - New Account

 **OpenMRS** Metadata Sharing Server

New User Registration

Please fill out the following fields:

Name:

Email:

Password:

Already registered? Login [here!](#)
[Home](#)

Thanks for using *Metadata Sharing Server!*

Page by JWild. 2012.

Figure 4.2.2: Screenshot of the OpenMRS MDSS user registration page.



 **OpenMRS** Metadata Sharing Server

Welcome, test!

You are in the admin user class!

[Manage Packages](#)
[Manage Users](#)
[Upload](#)
[Home](#)
[Logout](#)

Name: test
 Email Address: test@test.com
 Member Since: 2012-04-30
 Packages Uploaded: 0
[Edit](#)

Your Uploaded Packages:

Newest Most Popular Alphabetical By Release

No records found for that query.

name	description	authority	date published	version	OpenMRS version	downloads	actions
No records found for that query.							

Figure 4.2.3: Screenshot of the OpenMRS MDSS profile page.



Figure 4.2.4: Screenshot of the OpenMRS MDSS upload page.

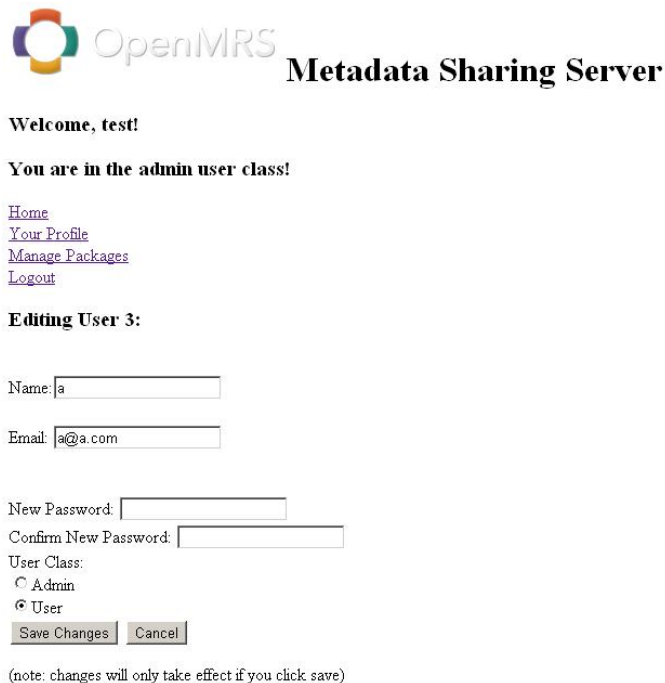


Figure 4.2.5: Screenshot of the OpenMRS MDSS homepage.

4.3 Testing

Although the MDSS project did not directly involve handling any tickets in OpenMRS trunk itself, the time spent getting a running development environment of OpenMRS was useful for installing the MDS module and generating test packages for use in the MDSS.

The MDSS was successfully tested for graceful failure behavior on upload with different input files including: a) valid packages, b) zip archives with corrupted XML files, c) zip archives with no xml files, d) corrupted zip files, e) php files, f) images, and g) XML files. The MDSS was tested on 5 browsers: Chrome, Chromium, Firefox, Aurora, and Safari. There is some cross-browser support in the code, especially in `download.php` (see Section F.13), and in the elements which use JQuery and JQuery UI. Overall cross-browser support needs more consideration in the next generation of MDSS.

For code formatting, several different external checkers were used. HTML output of the MDSS is run through the W3Schools -XHTML standard checker to maintain well-formed output. Plain Javascript code (for example, see Section F.23 in Appendix F) is run through JSLint style checker. JSON output is run through JSONLint validator. Analogous to the rigorous unit tests of the OpenMRS Java coding guidelines, the `simpletest.org` PHP unit testing framework was investigated for unit testing. Simpletest has very good support for mocking classes and doing test driven design. However, this MDSS design was already set by the time Simpletest became an option and it is not Object Oriented enough for the tests to be pervasive. Instead all functions were independently tested to see that they fulfill their interface contracts. This implementation of the MDSS was not stress-tested and did not undergo any non-technical user trials.

Chapter 5

Evaluation

The original success evaluation criteria for this project proposed in October were: 1) OpenMRS successfully using the MDSS software, 2) MDSS integrated into MDS Module, 3) project keeping on schedule, and 4) intangibles of experience/learning with real world software engineering and development.

Two of those original criteria have been abandoned as failed along the way. The initial schedule was not kept, and although it might be shown to be unrealistic, I made a case that it was okay when I proposed it and it would not have been approved if it was entirely unfeasible. Nonetheless, it had to be abandoned because the milestones were consistently not met. With the shift from Java to PHP, the potential integration with the MDS module was also delayed indefinitely. The MDSS and MDS can still work together well by sharing packages as zip archives, but they still require minimal user action to move the files from one to the other.

The other two original criteria have been successfully met. OpenMRS has not had time yet to use the MDSS in production, but my mentor Rafal successfully deployed the MDSS onto XAMPP on the OpenMRS server on May 2nd. His feedback on the code is that it is “clean and easy to understand” and he also stated his opinion that the logic of the code successfully met the functionality goals for the server although the UI was not what he had envisioned. With notable caveats he even thought it could be ‘good’. Those caveats include the following changes: 1) refactoring to separate logic from view, 2) moving the PHP into a framework such as Zend or Cake to make further UI development easier for someone with CSS-only skills, and 3) switching the database calls over to an ORM. The MDSS is the subject of a Google Summer of Code Project this summer and this code may

or may not find a home in the next generation of server. The code in Appendix F is in flux even now as I improve the MDSS with new ideas that did not make it into the build that was deployed. This is the nature of OSS work, that all contributions go into the code base and will likely be adapted or co-opted by future developers, but there is no need for egos when all improvements are helping a humanitarian cause. When it comes to intangibles this project has bought lots of experience. I have new appreciation and a closer perspective on the difficulties of project scheduling and maintaining working relationships under distributed development and long distance collaboration.

I would also like to suggest a new fifth success criteria for this project. Any software project following engineering processes and generating a formal requirements specification, will be validated against those same requirements. There is no formal validation process in place for this project, but it has succeeded in meeting all of the points of its SRS. A success criteria of functionality, meeting the elicited and specified requirements, is met in this project. Overall, my project meets at least 3 out of 5 of its success criteria, and has both benefited the world and hopefully set a pattern for more student work to do so.

Chapter 6

Conclusion

6.1 Future Work

In the short term there are several features that occurred to me during the development of this implementation of the MDSS which given another month of work I might like to include. Some of these additions include: a) allowing uploading of password protected or private packages, b) other increased security measures such as password strength requirements and better facility for users to recover their own lost or forgotten password, c) comments on packages, d) tracking of time between user logins for inactivity, e) a warning/banning system to allow removal of a users uploading privileges without removing them from the server entirely, and f) a complete style/theme for the page instead of random CSS hacks.

In the longer term future, given another semester or two to continue working on the MDSS there are many more options to consider and directions that I would take. I want to expand on the proofs of concept featured in this implementation which features examples of many techniques that could be further expanded to improve the user experience and capabilities of the MDSS. Javascript, which was used for some form validation, searching, and tabbed display, is the technology with the most potential for upgrading the MDSS. Especially interesting is the potential further use of a javascript framework for this task. A PHP framework such as Cake could also be investigated. Chances are that something general that I would implement has been implemented by someone else before and been standardized, made more full featured, efficient, and/or elegant, and generally improved for inclusion in a framework library. JQuery and JQuery UI are both Javascript frameworks which have lots to offer the MDSS and JQuery UI in particular might be useful in composing a unifying

theme and style for the whole UI. Because this server was targeted to use a MySQL database it is coded directly with MySQL RDBMS in mind. A good expansion in the future would be to replace this with the abstraction of an ORM system such as PDO so it can be storage solution independent.

Finally, a major design pattern that I learned about through the community at OpenMRS but was too far along to re-implement to use at this time is RESTful web services patterns. These web services define four design principles that greatly neaten up and formalize the interfaces of the server: 1) Use HTTP methods (Requests) explicitly, 2) Expose directory structure-like URIs, 3) be Stateless, and 4) Transfer XML, JavaScript Object Notation (JSON), or both. Implementation of RESTful style web services would provide a much cleaner, orderly design to the MDSS as well as helping aid further integration with the main OpenMRS platform and the MDS module in particular. The ideal end goal is for the MDS module to interact with the MDSS directly so that the lists of uploaded packages are visible in the MDS and no extra import of the downloaded packages from file is necessary because they download directly into the MDS import processes.

6.2 Conclusion

The OpenMRS MDSS project was a very good introduction to a world of software development which the curriculum at Princeton University does not focus on. Connections were made between faculty and developers, and hopefully the particular complications of this project have not hampered future academic-HFOSS collaborative endeavors. It is my hope that, having proven the feasibility of independent work in a real world project that helps others, it will inspire future students to also find their interest in making the world a better place.

Bibliography

- [1] OpenMRS. (2011). OpenMRS » Open source health IT for the planet. [Website]. <http://www.openmrs.org>
- [2] Ghosh, Rishab Aiyer, Ruediger Glott, Bernhard Krieger, and Gregorio Robles. (2002). Free/Libre and Open Source Software: Survey and Study FLOSS. Part IV: Survey of Developers. [Report]. International Institute of Infonomics. University of Maastricht, The Netherlands. <http://flossproject.org/report/Final4.htm>
- [3] Sommerville, Ian. Software Engineering 8. Harlow, England New York: Addison-Wesley, 2007.
- [4] Abran, Alain , Moore, James W. , Bourque, Pierre , and Dupuis, Robert Editors. Guide to the Software Engineering Body of Knowledge : SWEBOK. Los Alamitos, Calif: IEEE Computer Society Press, 2004.
- [5] Scacchi, Walt. (2007). Free/open source software development. In Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering (ESEC-FSE '07). ACM, New York, NY, USA, 459-468. DOI=10.1145/1287624.1287689 <http://doi.acm.org/10.1145/1287624.1287689>
- [6] Noll, John and Wei-Ming Liu. (2010). Requirements elicitation in open source software development: a case study. In Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development (FLOSS '10). ACM, New York, NY, USA, 35-40. DOI=10.1145/1833272.1833279 <http://doi.acm.org/10.1145/1833272.1833279>
- [7] Stol, Klaas-Jan and Muhammad Ali Babar. (2010). Challenges in using open source software in product development: a review of the literature. In Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development(FLOSS '10). ACM, New York, NY, USA, 17-22. DOI=10.1145/1833272.1833276 <http://doi.acm.org/10.1145/1833272.1833276>
- [8] Hislop, Gregory W., Heidi J.C. Ellis, and Ralph A. Morelli. (2009). Evaluating student experiences in developing software for humanity. In Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science education (ITiCSE '09). ACM, New York, NY, USA, 263-267. DOI=10.1145/1562877.1562959 <http://doi.acm.org/10.1145/1562877.1562959>

- [9] Ellis, Heidi J.C., Gregory W. Hislop, and Ralph A. Morelli. (2011). A comparison of software engineering knowledge gained from student participation in humanitarian foss projects. In Proceedings of the 16th annual joint conference on Innovation and technology in computer science education (ITiCSE '11). ACM, New York, NY, USA, 360-360. DOI=10.1145/1999747.1999874 <http://doi.acm.org/10.1145/1999747.1999874>
- [10] Tierney WM, Achieng M, Baker E, Bell A, Biondich P, Braitstein P, et al. (2010). Experience implementing electronic health records in three East African countries. Stud Health Technol Information. IOS Press.
- [11] Mamlin BW, Biondich PG, Wolfe BA, Fraser H, Jazayeri D, Allen C, et al. (2006). Cooking up an open source EMR for developing countries: OpenMRS - a recipe for successful collaboration. AMIA Annu Symp Proc. 2006:529-33.
- [12] Allen C, Jazayeri D, Miranda J, Biondich PG, Mamlin BW, Wolfe BA, et al. (2007). Experience in implementing the OpenMRS medical record system to support HIV treatment in Rwanda. Stud Health Technol Inform.
- [13] Nisanbayev, Yerbol, Huiseong Na, Dongwook Lim, and Franz Ko. (2009). Designing an electronic medical records system using design patterns. In Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human (ICIS '09). ACM, New York, NY, USA, 1410-1415. DOI=10.1145/1655925.1656185 <http://doi.acm.org/10.1145/1655925.1656185>
- [14] Hoffmann, Leah. (2009). Implementing electronic medical records. Commun. ACM 52, 11 (November 2009), 18-20. DOI=10.1145/1592761.1592770 <http://doi.acm.org/10.1145/1592761.1592770>

Appendix A

Global OpenMRS Distribution



Figure A.0.1: A map of current OpenMRS clinical and research sites

OpenMRS Sites : Clinical & Research Sites

Type	Locality	Country	Site	Latitude	Longitude
Clinical	Buenos Aires	Argentina	Gobierno de la Ciudad de Buenos Aires	-34.5761	-58.4088
Research	Yerevan	Armenia	American University of Armenia	40.193922	44.503984
Clinical	Palacios	Bolivia	Centro Médico Humberto Parra	-17.35	-63.6
Clinical	Gaborone	Botswana	Botswana National TB Control Programme	-24.6464	25.9119
Research	Santa Maria	Brasil	Universidade Federal de Santa Maria	-29.712087	-53.71687
Clinical	Siem Reap	Cambodia	Angkor Hospital for Children	13.345	103.854
Clinical	Siem Reap	Cambodia	Siem Reap Provincial Hospital of Cambodia	13.36865	103.86441
Clinical	Douala	Cameroon	Hôpital Général De Douala	4.0636	9.7589
Research	Santiago	Chile		-33.425362	-70.566467
Research	Temuco	Chile		-38.73333	-72.6
Research	Beijing	China	Beijing University of Aeronautics and Astronautics	39.943	116.388
Clinical	Brazzaville	Congo	CANTAM Project : Center for Vegetable Resources Studies	-4.2592	15.2847
Research	Camagüey	Cuba	University of Camagüey	21.386	-77.914
Clinical	Koraro	Ethiopia	Millennium Villages Project	13.978	39.43
Clinical	Lambaréné	Gabon	Albert Schweitzer Hospital	-0.701546	10.240386
Research	Göttingen	Germany	University of Göttingen	51.534404	9.938464
Research	Freiburg im Breisgau	Germany	University of Freiburg	48.119	8.399
Research	Accra	Ghana	Ghana Health Service	5.555717	-0.196306
Clinical	Bonsasso	Ghana	Millennium Villages Project	6.687119	-1.621969
Research	Guatemala City	Guatemala	Universidad del Valle de Guatemala	14.643	-90.513
Clinical	Deschappelles	Haiti	Albert Schweitzer Hospital	19.083332	-72.5
Clinical	Port-au-Prince	Haiti	Martissant Health Center	18.539268	-72.336411
Clinical	Cange	Haiti	Partners In Health	18.934498	-71.994965
Clinical	Petite-Riviere-de-Nippes	Haiti	Visitation Hospital	18.475124	-73.23304
Clinical	Nuevo Paraiso	Honduras	Global Brigades	13.86414	-86.552689
Clinical	Dehradun	India	Doon Hospital	30.33357	78.04776
Research	Kuningan	Indonesia	University of Bakrie School of Management	-6.220711	106.833618
Research	Jerusalem	Israel	Hebrew University of Jerusalem	31.779	35.2253
Research	Bolzano	Italy	Free University of Bozen	46.4927	11.3336
Clinical	Tokushima	Japan	University of Tokushima Medical and Dental Hospital	34.0667	134.5667
Clinical	Eldoret	Kenya	AMPATH	0.51524	35.266369
Clinical	Kisumu	Kenya	Lumumba Health Centre	-0.09516	34.74733
Clinical	Garissa	Kenya	Millennium Villages Project	-0.444659	39.671711
Clinical	Sauri	Kenya	Millennium Villages Project	0.10131	34.200765
Clinical	Nohana	Lesotho	Partners In Health	-30.054213	27.854214
Clinical	Bela-Bela	Lesotho	Monyetla Centre	-29.02285	28.01702
Clinical	Lilongwe	Malawi	Baobab Health Partnership	-13.9826	33.773762
Clinical	Neno	Malawi	Partners In Health	-15.559766	34.504368
Clinical	Namitete	Malawi	St. Gabriel's Hospital	-14.0167	33.35
Clinical	Sabah	Malaysia	Tenom Hospital	5.5	117
Clinical	Ségou	Mali	Millennium Villages Project	13.440597	-6.265322
Clinical	Tiby	Mali	Millennium Villages Project	14.116667	-7.983333
Clinical	Toya	Mali	Millennium Villages Project	12.342	-7.869
Clinical	Sourakabougou	Mali	Mali Health Organizing Project	13.6552	-9.0256
Clinical		Mozambique	OASIS - South African Medical Research Council	-25.968945	32.56955
Clinical		Nepal	World Health Organization	27.702871	85.318245
Clinical	Sanepa	Nepal	Star Hospital	27.68421	85.30108
Clinical	Managua	Nicaragua	Sustainable Sciences Institute	12.136	-86.275
Research	Managua	Nicaragua	Sustainable Sciences Institute	12.136	-86.275

Clinical	Ikaram	Nigeria	Millennium Villages Project	7.1	4.833
Clinical	Pampaida	Nigeria	Millennium Villages Project	10.523	7.44
Clinical	Kaduna	Nigeria	Shehu Idris College of Health Sciences and Technology	10.51901	7.443122
Clinical	Karachi	Pakistan	Indus Hospital Research Center	24.893379	67.028061
Clinical	Karachi	Pakistan		24.893379	67.028061
Research	Karachi	Pakistan		24.893379	67.028061
Clinical	Lima	Peru	Socios En Salud Sucursal Peru	-12.093084	-77.046494
Clinical	Santa Anita	Peru	Hospital Hermilio Valdizan	-12.0333	-76.9667
Clinical	Lima	Peru	Hospital Daniel A. Carrión	-12.06016	-77.12797
Clinical	Manila	Philippines	National Chronic Disease Registry - Philippine Diabetes Reg	14.601033	120.976158
Research	Manila	Philippines	University of the Philippines National Telehealth Center	14.601033	120.976158
Research	Legazpi	Philippines	Bicol University	13.5	123.5
Clinical	Mayange	Rwanda	Millennium Villages Project	-2.2114	30.1172
Clinical	Rwinkwavu	Rwanda	Partners In Health	-1.902265	30.509197
Clinical	Kigali	Rwanda	Partners In Health, TB program, MOH	-1.950106	30.058769
Research	Kigali	Rwanda	TRACplus Clinic	-1.950106	30.058769
Clinical	Gitwe	Rwanda	Moi + Medical Missions for Children	-2.2461	29.689301
Research	Jeddah	Saudi Arabia	Capitals Institute of Health Sciences	21.5169	39.2192
Clinical	Potou	Senegal	Millennium Villages Project	15.720919	-16.506308
Clinical	Freetown	Sierra Leone	Connaught Hospital	8.48799	-13.23829
Clinical	Umkhanyakude	South Africa	Africa Centre PCIS	-27.616667	32.033333
Clinical	Cape Town	South Africa	Desmond Tutu HIV Center	-33.923775	18.423346
Research	Cape Town	South Africa	Jembi Headoffice	-33.923775	18.423346
Clinical	Durban	South Africa	SA Medical Research Council/UKZN	-29.857002	31.024794
Research	Bloemfontein, Free State	South Africa	University of the Free State Department of Virology	-29.116	26.15
Research	Pretoria	South Africa	University of South Africa	-25.7069	28.2294
Clinical	Tygerberg	South Africa	Tygerberg Hospital	-33.910433	18.613179
Clinical	Queenstown	South Africa	Frontier Hospital	-31.8917	26.8819
Research	Colombo	Sri Lanka	University of Colombo Software Development Unit	6.901948	79.861561
Clinical	Colombo	Sri Lanka	Post Graduate Institute of Medicine	6.917523	79.869306
Research	Peradeniya	Sri Lanka	University of Peradeniya Department of Computer Engineering	7.25382	80.5917
Research	Geneva	Switzerland	World Health Organization (WHO)	46.2022	6.1457
Research	Geneva	Switzerland	University of Geneva	46.194887	6.140055
Clinical	Mbola	Tanzania	Millennium Villages Project	-4.9167	32.8333
Clinical	Kibaha	Tanzania	University of Dar es Salaam	-6.7667	38.916698
Clinical	Dar es Salaam	Tanzania	University of Dar es Salaam	-6.822921	39.269661
Clinical	Morogoro	Tanzania	University of Dar es Salaam	-6.822097	37.66116
Research	Bangkok	Thailand	International School of Bangkok	13.75	100.5167
Clinical	Fajara, Banjul	The Gambia	Medical Research Council (UK) The Gambia	13.4716	-16.6968
Clinical	Kampala	Uganda	Makarere University	0.314269	32.572872
Clinical	Ruhiira	Uganda	Millennium Villages Project	-0.843543	30.803947
Clinical	Mbarara	Uganda	Immune Suppression Syndrome (ISS) Clinic	-0.605675	30.64855
Clinical	Masaka	Uganda	Masaka Regional Referral Hospital	-0.3128	31.7131
Clinical	Kasese	Uganda	Kagando Hospital	0.23	29.9883
Clinical	Masaka	Uganda	Kitovu Mobile AIDS Organisation	-0.5	31.75
Research	Leeds	United Kingdom	University of Leeds	53.801335	-1.553278
Research	Southampton	United Kingdom	University of Southampton School of Health Sciences	50.92872	-1.40266
Clinical	Indianapolis, Indiana	United States	Child Health Improvement through Computer Automation (C	39.77726	-86.180008
Clinical	Indianapolis, Indiana	United States	Indianapolis Motor Speedway	39.790975	-86.235133
Research	Indianapolis, Indiana	United States	OpenMRS LLC	39.78133	-86.16678
Research	Boston, Massachusetts	United States	Partners In Health	42.358433	-71.059776
Research	Indianapolis, Indiana	United States	Regenstrief Institute	39.77864	-86.17785
Clinical	Los Angeles, California	United States	Skid Row Homeless Healthcare Initiative	34.052235	-118.24368

Research	Oak Hill, Virginia	United States	SYSNET International, Inc.	38.937763	-77.412819
Research	Pittsburgh, Pennsylvania	United States	University of Pittsburgh	40.443184	-79.961586
Research	Carrollton, Georgia	United States	University of West Georgia	33.580109	-85.076614
Research	Nashville, Tennessee	United States	Vanderbilt University	36.142952	-86.805489
Research	San Mateo, California	United States	Actuate Corporation	37.563	-122.3255
Research	Palo Alto, California	United States	Stanford University School of Medicine	37.4312	-122.1703
Research	Denver, Colorado	United States	Genesee Academy LLC	39.739	-104.985
Research	Salina, Kansas	United States	Kansas Wesleyan University	38.813892	-97.609478
Research	Boston, Massachusetts	United States	Harvard Medical School	42.3659	-71.1226
Clinical	Boston, Massachusetts	United States	Children's Hospital Boston	42.337126	-71.106037
Clinical	Kansas City, Missouri	United States	Children International	39.1	-94.581
Research	New York, New York	United States	Doctors Without Borders US	40.728	-73.947
Research	Tulsa, Oklahoma	United States	Tulsa Community College	36.04384	-95.86203
Research	Portland, Oregon	United States	Oregon Health & Science University	45.524	-122.668
Research	Conshohocken, Pennsylvania	United States	ePharmaSolutions	40.077	-75.303
Research	Pittsburgh, Pennsylvania	United States	University of Pittsburgh	40.444752	-79.952568
Clinical	Philadelphia, Pennsylvania	United States	The Children's Hospital of Philadelphia	39.94816	-75.19464
Research	Charlottesville, Virginia	United States	University of Virginia	38.060495	-78.502842
Research	Seattle, Washington	United States	University of Washington	47.656099	-122.31055
Clinical	Zanzibar City	Zanzibar	Ministry of Health and Social Welfare	-6.1639	39.1979
Clinical	Harare	Zimbabwe	OASIS - South African Medical Research Council	-17.82922	31.053961
Clinical	Chitungwiza	Zimbabwe	Opportunistic Infection Clinic	-18.01978	31.067907

Figure A.0.2: A table listing the current OpenMRS clinical and research sites

Appendix B

Initial Project Schedule

Joseph Wilder Metadata Sharing Server : Initial Project Plan

Start Date: 10/27/11 End Date: 04/30/12

Item	Description	Start Date	Anticipated	
			Duration	Completion Date
Server		10/27/11	53	12/20/11
1.1	Project definition tasks			
1.2	Learning tasks (Eclipse, Java servers, threads,Server			
1.3	Scenarios and Requirements (SRS)			
1.4	GUI mock-ups for uploading file, listing and			
1.5	Development Environment			
1.6	Server Implementation – web interface			
1.7	Documentation			
Feature Pack 1		12/27/11	38	02/05/12
2.1	Integration with Metadata Sharing Module			
2.2	Scenarios and Requirements (SRS)			
2.3	Authentication of Publishers			
2.4	GUI mock-ups for uploading, downloading			
2.5	Implementation in server and MDSM			
2.6	Documentation			
Feature Pack 2		02/10/12	29	03/11/12
3.1	Scenarios and Requirements (SRS)			
3.2	GUI mock-ups for search, stats, recommendation?			
3.3	Implementation in server and MDSM			
3.4	Documentation			
Updates from Feedback		03/12/12	19	03/31/12
4.1	Open tickets in JIRA			
4.2	Other feedback			
4.3	Documentation			
Documentation				
5.1	Write documentation and university paper	03/31/12	12	04/12/12

Figure B.0.1: The initial project management schedule.

Appendix C

Example Metadata Package

C.1 header.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<package id="1" uuid="f446e1b9-10ca-4c2a-822f-ad0c972158fb">
  <dateCreated id="2">2011-12-02 15:59:40</dateCreated>
  <name>New NAME</name>
  <description>Test package 1 of ?.</description>
  <openmrsVersion>1.9.0-SNAPSHOT</openmrsVersion>
  <version>1</version>
  <group>3bc4ab33-18e8-4e5a-817b-0b08a13bb9e1</group>
  <subscriptionUrl>http://localhost:8080/openmrs/ws/rest/metadatasharing/
    package/3bc4ab33-18e8-4e5a-817b-0b08a13bb9e1/latest</subscriptionUrl>
  <items id="3">
    <explicitItem id="4">
      <classname>org.openmrs.Role</classname>
      <uuid>8d94f280-c2cc-11de-8d13-0010c6dffd0f</uuid>
      <name>Provider</name>
    </explicitItem>
    <explicitItem id="5">
      <classname>org.openmrs.Role</classname>
      <uuid>f7fd42ef-880e-40c5-972d-e4ae7c990de2</uuid>
      <name>Authenticated</name>
    </explicitItem>
    <explicitItem id="6">
      <classname>org.openmrs.Role</classname>
      <uuid>774b2af3-6437-4e5a-a310-547554c7c65c</uuid>
      <name>Anonymous</name>
    </explicitItem>
    <explicitItem id="7">
      <classname>org.openmrs.Role</classname>
      <uuid>8d94f852-c2cc-11de-8d13-0010c6dffd0f</uuid>
      <name>System Developer</name>
    </explicitItem>
    <explicitItem id="8">
      <classname>org.openmrs.Role</classname>
      <uuid>6a1378ce-1d1d-11e1-94a3-0018de157d2c</uuid>
      <name>Clinician</name>
    </explicitItem>
    <explicitItem id="9">
      <classname>org.openmrs.Role</classname>
      <uuid>6a137bda-1d1d-11e1-94a3-0018de157d2c</uuid>
      <name>Data Assistant</name>
    </explicitItem>
    <explicitItem id="10">
      <classname>org.openmrs.Role</classname>
      <uuid>6a1383b0-1d1d-11e1-94a3-0018de157d2c</uuid>
      <name>Data Manager</name>
    </explicitItem>
    <includedDependency id="11">
      <classname>org.openmrs.Privilege</classname>
      <uuid>68dc7b8e-1d1d-11e1-94a3-0018de157d2c</uuid>
```

```

    <name>View Problems</name>
  </includedDependency>
  <includedDependency id="12">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc5301-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Concepts</name>
  </includedDependency>
  <includedDependency id="13">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc5917-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Encounters</name>
  </includedDependency>
  <includedDependency id="14">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc6f5d-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Patient Identifiers</name>
  </includedDependency>
  <includedDependency id="15">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc3937-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Allergies</name>
  </includedDependency>
  <includedDependency id="16">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc673f-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Observations</name>
  </includedDependency>
  <includedDependency id="17">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc737a-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Patients</name>
  </includedDependency>
  <includedDependency id="18">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc757d-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View People</name>
  </includedDependency>
  <includedDependency id="19">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc5f20-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Global Properties</name>
  </includedDependency>
  <includedDependency id="20">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc87d5-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Roles</name>
  </includedDependency>
  <includedDependency id="21">
    <classname>org.openmrs.Privilege</classname>
    <uuid>68dc4ab3-1d1d-11e1-94a3-0018de157d2c</uuid>
    <name>View Concept Classes</name>
  </includedDependency>
</items>
</package>

```


C.2 metadata.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<list id="1">
  <org.openmrs.Role id="2" uuid="8d94f280-c2cc-11de-8d13-0010c6dffd0f">
    <description>All users with the &apos;Provider&apos; role will appear as
      options in the default Infopath </description>
    <retired>>false</retired>
    <role>Provider</role>
    <privileges id="3">
      <org.openmrs.Privilege id="4" uuid="68dc7b8e-1d1d-11e1-94a3-0018de157d2c
        ">
        <description>Able to view problems in OpenMRS</description>
        <retired>>false</retired>
        <privilege>View Problems</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="5" uuid="68dc5301-1d1d-11e1-94a3-0018de157d2c
        ">
        <description>Able to view concept entries</description>
        <retired>>false</retired>
        <privilege>View Concepts</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="6" uuid="68dc5917-1d1d-11e1-94a3-0018de157d2c
        ">
        <description>Able to view patient encounters</description>
        <retired>>false</retired>
        <privilege>View Encounters</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="7" uuid="68dc3937-1d1d-11e1-94a3-0018de157d2c
        ">
        <description>Able to view allergies in OpenMRS</description>
        <retired>>false</retired>
        <privilege>View Allergies</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="8" uuid="68dc6f5d-1d1d-11e1-94a3-0018de157d2c
        ">
        <description>Able to view patient identifiers</description>
        <retired>>false</retired>
        <privilege>View Patient Identifiers</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="9" uuid="68dc673f-1d1d-11e1-94a3-0018de157d2c
        ">
        <description>Able to view patient observations</description>
        <retired>>false</retired>
        <privilege>View Observations</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="10" uuid="68dc737a-1d1d-11e1-94a3-0018
        de157d2c">
        <description>Able to view patients</description>
        <retired>>false</retired>
        <privilege>View Patients</privilege>
      </org.openmrs.Privilege>
      <org.openmrs.Privilege id="11" uuid="68dc757d-1d1d-11e1-94a3-0018
        de157d2c">
        <description>Able to view person objects</description>
        <retired>>false</retired>
        <privilege>View People</privilege>
      </org.openmrs.Privilege>
    </privileges>
    <inheritedRoles id="12"/>
  </org.openmrs.Role>

```

```

<org.openmrs.Role id="13" uuid="f7fd42ef-880e-40c5-972d-e4ae7c990de2">
  <description>Privileges gained once authentication has been established.</
    description>
  <retired>>false</retired>
  <role>Authenticated</role>
  <privileges id="14">
    <org.openmrs.Privilege id="15" uuid="68dc5f20-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view global properties on the administration
        screen</description>
      <retired>>false</retired>
      <privilege>View Global Properties</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="16" uuid="68dc87d5-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view user roles</description>
      <retired>>false</retired>
      <privilege>View Roles</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="17" uuid="68dc4ab3-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view concept classes</description>
      <retired>>false</retired>
      <privilege>View Concept Classes</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="18" uuid="68dc81a7-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view relationships</description>
      <retired>>false</retired>
      <privilege>View Relationships</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="19" uuid="68dc7f9d-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view relationship types</description>
      <retired>>false</retired>
      <privilege>View Relationship Types</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="20" uuid="68dc6130-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view patient identifier types</description>
      <retired>>false</retired>
      <privilege>View Identifier Types</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="21" uuid="68dc4cd3-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view concept datatypes</description>
      <retired>>false</retired>
      <privilege>View Concept Datatypes</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="22" uuid="68dc5b1c-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view field types</description>
      <retired>>false</retired>
      <privilege>View Field Types</privilege>
    </org.openmrs.Privilege>
    <org.openmrs.Privilege id="23" uuid="68dc6332-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Able to view locations</description>
      <retired>>false</retired>
      <privilege>View Locations</privilege>
    </org.openmrs.Privilege>
  </privileges id="14">
</org.openmrs.Role>

```

```

<org.openmrs.Privilege id="24" uuid="68dc777f-1d1d-11e1-94a3-0018
  de157d2c">
  <description>Able to view person attribute types</description>
  <retired>>false</retired>
  <privilege>View Person Attribute Types</privilege>
</org.openmrs.Privilege>
<org.openmrs.Privilege id="25" uuid="68dc798f-1d1d-11e1-94a3-0018
  de157d2c">
  <description>Able to view user privileges</description>
  <retired>>false</retired>
  <privilege>View Privileges</privilege>
</org.openmrs.Privilege>
<org.openmrs.Privilege id="26" uuid="68dc5710-1d1d-11e1-94a3-0018
  de157d2c">
  <description>Able to view encounter types</description>
  <retired>>false</retired>
  <privilege>View Encounter Types</privilege>
</org.openmrs.Privilege>
<org.openmrs.Privilege id="27" uuid="68dc6944-1d1d-11e1-94a3-0018
  de157d2c">
  <description>Able to view order types</description>
  <retired>>false</retired>
  <privilege>View Order Types</privilege>
</org.openmrs.Privilege>
</privileges>
<inheritedRoles id="28"/>
</org.openmrs.Role>
<org.openmrs.Role id="29" uuid="774b2af3-6437-4e5a-a310-547554c7c65c">
  <description>Privileges for non-authenticated users.</description>
  <retired>>false</retired>
  <role>Anonymous</role>
  <privileges id="30">
    <org.openmrs.Privilege id="31" uuid="68dc6534-1d1d-11e1-94a3-0018
      de157d2c">
      <description>Ability to see the navigation menu</description>
      <retired>>false</retired>
      <privilege>View Navigation Menu</privilege>
    </org.openmrs.Privilege>
  </privileges>
  <inheritedRoles id="32"/>
</org.openmrs.Role>
<org.openmrs.Role id="33" uuid="8d94f852-c2cc-11de-8d13-0010c6dffd0f">
  <description>Developers of the OpenMRS .. have additional access to change
    fundamental structure of the database model.</description>
  <retired>>false</retired>
  <role>System Developer</role>
  <privileges id="34"/>
  <inheritedRoles id="35"/>
</org.openmrs.Role>
<org.openmrs.Role id="36" uuid="6a1378ce-1d1d-11e1-94a3-0018de157d2c">
  <description>Privileges for doctors</description>
  <retired>>false</retired>
  <role>Clinician</role>
  <privileges id="37"/>
  <inheritedRoles id="38">
    <org.openmrs.Role reference="2"/>
  </inheritedRoles>
</org.openmrs.Role>
<org.openmrs.Role id="39" uuid="6a137bda-1d1d-11e1-94a3-0018de157d2c">
  <description>Privileges for data-entry operators.</description>
  <retired>>false</retired>

```

```

<role>Data Assistant</role>
<privileges id="40">
  <org.openmrs.Privilege reference="8"/>
  <org.openmrs.Privilege id="41" uuid="68dbb089-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to add relationships</description>
    <retired>>false</retired>
    <privilege>Add Relationships</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege id="42" uuid="68dc8bfd-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to view users in OpenMRS</description>
    <retired>>false</retired>
    <privilege>View Users</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege id="43" uuid="68dbe9d1-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to edit relationships</description>
    <retired>>false</retired>
    <privilege>Edit Relationships</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege id="44" uuid="68dbcd5b-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to delete relationships</description>
    <retired>>false</retired>
    <privilege>Delete Relationships</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege reference="5"/>
  <org.openmrs.Privilege reference="6"/>
  <org.openmrs.Privilege id="45" uuid="68dc2635-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to view the &apos;Encounters&apos; tab on the
      patient dashboard</description>
    <retired>>false</retired>
    <privilege>Patient Dashboard – View Encounters Section</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege id="46" uuid="68dc2858-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Allows user to view the Forms tab on the patient
      dashboard</description>
    <retired>>false</retired>
    <privilege>Patient Dashboard – View Forms Section</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege id="47" uuid="68dc2417-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to view the &apos;Demographics&apos; tab on the
      patient dashboard</description>
    <retired>>false</retired>
    <privilege>Patient Dashboard – View Demographics Section</privilege>
  </org.openmrs.Privilege>
  <org.openmrs.Privilege id="48" uuid="68dbe56e-1d1d-11e1-94a3-0018
    de157d2c">
    <description>Able to edit patients</description>
    <retired>>false</retired>
    <privilege>Edit Patients</privilege>
  </org.openmrs.Privilege>
</privileges>
<inheritedRoles id="71">
  <org.openmrs.Role reference="39"/>
</inheritedRoles>
</org.openmrs.Role>
</list>

```

Appendix D

IEEE SRS Standard

IEEE/ANSI 830-1998
Standard structure for Software Requirements Specifications

1. Introduction
 - 1.1. Purpose of the requirements documentation
 - 1.2. Scope of the product
 - 1.3. Definitions, acronyms, and abbreviations
 - 1.4. References
 - 1.5. Overview of the remainder of the document
2. General description
 - 2.1. Product perspective
 - 2.2. Product functions
 - 2.3. User Characteristics
 - 2.4. General Constraints
 - 2.5. Assumptions and dependencies
3. Specific requirements
 - 3.1. Functional requirements
 - 3.2. Non-functional requirements
 - 3.3. Interface requirements
 - 3.4. Further structure at the discretion of organisational practice. May include: external interfaces, system functionality and performance, logical database requirements, design constraints, emergent system properties, and quality characteristics.
4. Appendices
5. Index

Appendix E

Database Schema

```
— phpMyAdmin SQL Dump
— version 3.4.5
— http://www.phpmyadmin.net
—
— Host: localhost
— Generation Time: May 01, 2012 at 07:20 AM
— Server version: 5.5.16
— PHP Version: 5.3.8
SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

— Database: 'mdss'
—
—
— Table structure for table 'metadata'
—
CREATE TABLE IF NOT EXISTS 'metadata' (
  'id' int(11) NOT NULL AUTO_INCREMENT,
  'name' varchar(255) CHARACTER SET utf8 NOT NULL,
  'path' varchar(255) CHARACTER SET utf8 NOT NULL,
  'auth' varchar(255) CHARACTER SET utf8 NOT NULL,
  'authid' int(11) NOT NULL,
  'pubdate' date NOT NULL,
  'update' date NOT NULL,
  'desc' varchar(510) CHARACTER SET utf8 NOT NULL,
  'omrsver' varchar(255) CHARACTER SET utf8 NOT NULL,
  'ver' int(11) NOT NULL,
  'suburl' varchar(255) CHARACTER SET utf8 NOT NULL,
  'dls' int(11) NOT NULL,
  PRIMARY KEY ('id')
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=10 ;

— Dumping data for table 'metadata'
—
INSERT INTO 'metadata' ('id', 'name', 'path', 'auth', 'authid', 'pubdate', '
update', 'desc', 'omrsver', 'ver', 'suburl', 'dls') VALUES
(6, 'HTML', 'C:/xampp/htdocs/MDSS/uploads/788061
d1ae339ee8dd8b5f08d1565b6b7f53b4abda8baef7.zip', 'admin', 4, '2011-12-02',
'2012-04-30', 'Test package 1 of 4.', '1.9.0-SNAPSHOT', 1, 'http://
localhost:8080/openmrs/ws/rest/metadatasharing/package/3bc4ab33-18e8-4e5a
-817b-0b08a13bb9e1/latest', 1),
(7, 'Espanol', 'C:/xampp/htdocs/MDSS/uploads/4
ee94231c55659a252e70bcc706533382b9a1077a0c52b0f.zip', 'Bob', 5, '
2011-12-02', '2012-04-30', 'Test package 2 of 4.', '1.9.0-SNAPSHOT', 1, '
http://localhost:8080/openmrs/ws/rest/metadatasharing/package/3bc4ab33-18
e8-4e5a-817b-0b08a13bb9e1/latest', 1),
(8, 'Really Long Name of the Coolest Metadata Package Ever!', 'C:/xampp/htdocs
/MDSS/uploads/2550333310861a0a0afdca93964cddb0eeb48e692042b5a.zip', 'a',
```

```

3, '2011-12-02', '2012-04-30', 'Test package 3 of 4.', '1.9.0-SNAPSHOT',
1, 'http://localhost:8080/openmrs/ws/rest/metadatasharing/package/3bc4ab33-18e8-4e5a-817b-0b08a13bb9e1/latest', 3),
(9, 'Drug Resistant TB Plans', 'C:/xampp/htdocs/MDSS/uploads/5107f73ef55f166e784313e3c5a5cf00d0c85b46e2e1e592.zip', 'test', 6, '2011-12-02',
, '2012-04-30', 'Test package 4 of 4.', '1.9.0-SNAPSHOT', 1, 'http://localhost:8080/openmrs/ws/rest/metadatasharing/package/3bc4ab33-18e8-4e5a-817b-0b08a13bb9e1/latest', 2);

```

— Table structure for table 'users'

```

CREATE TABLE IF NOT EXISTS 'users' (
  'id' int(11) NOT NULL AUTO_INCREMENT,
  'name' varchar(255) CHARACTER SET utf8 NOT NULL,
  'email' varchar(255) CHARACTER SET utf8 NOT NULL,
  'pw' varchar(255) CHARACTER SET utf8 NOT NULL,
  'class' int(11) NOT NULL,
  'joindate' date NOT NULL,
  'packages' int(11) NOT NULL,
  UNIQUE KEY 'email' ('email'),
  KEY 'id' ('id')
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=7 ;

```

— Dumping data for table 'users'

```

INSERT INTO 'users' ('id', 'name', 'email', 'pw', 'class', 'joindate', 'packages') VALUES
(3, 'a', 'a@a.com', '1d5ebced5d3a2042a0fd830e4fcaccab50cd3b048dbe1b9', 0, '2012-04-26', 1),
(4, 'admin', 'admin@MDSS.org', '07ef2f6965f5217ea781e042bab327d263a7c1e9e469b93f', 1, '2012-04-26', 1),
(5, 'Bob', 'bob@bob.com', '607b101e41c7757063acd12d73688d6b7351243661e99beb', 0, '2012-04-30', 1),
(6, 'test', 'test@test.com', '7698f626e47c73620cc0b827a1bf53fd3ecc23747e95da76', 1, '2012-04-30', 1);
/*!40101 SET CHARACTER SET CLIENT=@OLD_CHARACTER SET CLIENT */;
/*!40101 SET CHARACTER SET RESULTS=@OLD_CHARACTER SET RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Appendix F

Code

The latest version of this code can be found in the Subversion repository at: <https://svn.openmrs.org/openmrs-contrib/metadata-sharing-server/trunk>

F.1 Setup

includes.php

```
1 <?php
2     include_once 'configs.php';
3     include_once 'pwFuncs.php';
4     include_once 'dbFuncs.php';
5     include_once 'templates.php';
6     include_once 'utils.php';
7 ?>
```

configs.php

```
1 <?php
2 // path to uploads folder
3     $uploadpath = dirname(__FILE__) . "/uploads/";
4
5 // DB connection details
6     $dbhost = 'localhost';
7     $dbuser = 'root';
8     $dbpass = 'admin';
9     $dbname = 'MDSS';
10
11 // DataTables table
12     $dttable = 'metadata';
13 ?>
```


F.2 templates.php

```

1 <?php
2 //General template and printing functions
3
4 // Basic page start
5 // Outputs HTML and DOCTYPE, does not handle including includes.php
6 function startPage($name)
7 {
8     Session_start();
9     echo '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" ';
10    echo '"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">';?>
11
12    <html xmlns="http://www.w3.org/1999/xhtml">
13    <head>
14        <meta charset="utf-8" />
15        <title>Metadata Sharing Server - <?php echo $name;?></title>
16        <link rel="stylesheet" type="text/css" href="http://ajax.googleapis.
17            com/ajax/libs/jqueryui/1.8.9/themes/ui-lightness/jquery-ui.css" />
18        <script type="text/javascript" src="http://ajax.googleapis.com/ajax/
19            libs/jquery/1.7.1/jquery.min.js"></script>
20        <script type="text/javascript" src="http://ajax.googleapis.com/ajax/
21            libs/jqueryui/1.8.18/jquery-ui.min.js"></script>
22        <!-- DataTables CSS -->
23        <link rel="stylesheet" type="text/css" href="http://ajax.aspnetcdn.com
24            /ajax/jquery.dataTables/1.8.2/css/jquery.dataTables.css">
25        <!-- DataTables -->
26        <script type="text/javascript" charset="utf8" src="http://ajax.
27            aspnetcdn.com/ajax/jquery.dataTables/1.8.2/jquery.dataTables.min.
28            js"></script>
29
30        <?php
31        ini_set('display_errors',1);
32        error_reporting(E_ALL);
33        echo '</head>';
34    }
35
36    // Basic MDSS header
37    // Outputs HTML
38    function printHeader()
39    { ?>
40        <div id="header">
41            <h1><a href="home.php"></a>
42            Metadata Sharing Server</h1>
43        </div>
44        <?php
45    }
46
47    // Basic MDSS printFooter
48    // Outputs HTML
49    function printFooter()
50    { ?>
51        <div id="Footer">
52            <h3>Thanks for using <em>Metadata Sharing Server</em>!</h3><br />
53            <p>Page by JWild. 2012.</p>
54        </div>
55        <?php
56    }
57
58    // Prints navigation option links
59    // Takes as parameter the current user for determining user class based
60    options

```

```

54 // Outputs HTML
55 function printNavOptions($user)
56 {
57     if($user['class'] == 1)
58     { ?>
59         <h3>You are in the admin user class! </h3>
60         <div id="options">
61             <a href="managePackages.php">Manage Packages</a><br />
62             <a href="manageUsers.php">Manage Users</a><br />
63         <?php
64         }
65     else
66     {
67         echo '<div id="options">';
68     } ?>
69         <a href="upload.php">Upload</a><br />
70         <a href="home.php">Home</a><br />
71         <a href="profilepage.php?<?php echo $user['id'];?>">Profile</a><br
72         />
73         <a href="logout.php">Logout</a><br />
74         <a href="loginPage.php">Login</a> to a different account<br />
75         <a href="newaccount.php">Register</a> a new account<br />
76     </div>
77     <br />
78 <?php
79 }
80 // Generates tabs and uses JQuery UI to style them
81 // Takes as parameters a user
82 // Beyond the initial query, delegates the actual content of the tabs to calls
83 // tabcontent
84 function tabs($who = false)
85 { ?>
86     <script>
87         $(document).ready(function() {
88             $("#tabs").tabs();
89         });
90         $(document).ready(function() {
91             $('#tab1table').dataTable( {
92                 "bProcessing": true,
93                 "bServerSide": true,
94                 "sAjaxSource": "proc.php",
95                 "aaSorting": [[ 3, "desc" ]],
96                 "bJQueryUI": true,
97                 "sPaginationType": "full_numbers"
98             } );
99         });
100         $(document).ready(function() {
101             $('#tab2table').dataTable( {
102                 "bProcessing": true,
103                 "bServerSide": true,
104                 "sAjaxSource": "proc.php",
105                 "aaSorting": [[ 6, "desc" ]],
106                 "bJQueryUI": true,
107                 "sPaginationType": "full_numbers"
108             } );
109         });
110         $(document).ready(function() {
111             $('#tab3table').dataTable( {
112                 "bProcessing": true,
113                 "bServerSide": true,

```

```

113         "sAjaxSource": "proc.php",
114         "aaSorting": [[ 0, "asc" ]],
115         "bJQueryUI": true,
116         "sPaginationType": "full_numbers"
117     } );
118     } );
119     $(document).ready(function () {
120     $('#tab4table').dataTable( {
121         "bProcessing": true,
122         "bServerSide": true,
123         "sAjaxSource": "proc.php",
124         "aaSorting": [[ 2, "asc" ]],
125         "bJQueryUI": true,
126         "sPaginationType": "full_numbers"
127     } );
128     } );
129     $(document).ready(function () {
130     $('#tab5table').dataTable( {
131         "bProcessing": true,
132         "bServerSide": true,
133         "sAjaxSource": "proc.php",
134         "aaSorting": [[ 5, "desc" ]],
135         "bJQueryUI": true,
136         "sPaginationType": "full_numbers"
137     } );
138     } );
139 </script>
140
141 <div id="tabs">
142     <ul>
143         <li><a href="#firsttab">Newest</a></li>
144         <li><a href="#secondtab">Most Popular</a></li>
145         <li><a href="#thirdtab">Alphabetical</a></li>
146         <?php if (!$who)
147             { ?>
148             <li><a href="#fourthtab">By Author</a></li>
149             <?php
150             } ?>
151         <li><a href="#fifthtab">By Release</a></li>
152     </ul>
153     <div id="firsttab">
154         <table id="tab1table" class="tabtable" border = "5px">
155             <thead>
156                 <tr><td>Name</td>
157                 <td>Description</td>
158                 <td>Authority</td>
159                 <td>Date Uploaded</td>
160                 <td>Version</td>
161                 <td>OpenMRS Version</td>
162                 <td>Downloads</td>
163             </tr>
164             </thead>
165         </table>
166     </div>
167     <div id="secondtab">
168         <table id="tab2table" class="tabtable" border = "5px">
169             <thead>
170                 <tr><td>Name</td>
171                 <td>Description</td>
172                 <td>Authority</td>
173                 <td>Date Uploaded</td>
174                 <td>Version</td>

```

```

175         <td>OpenMRS Version</td>
176         <td>Downloads</td>
177     </tr>
178     </thead>
179 </table>
180 </div>
181 <div id="thirdtab">
182     <table id="tab3table" class="tabtable" border = "5px">
183         <thead>
184             <tr><td>Name</td>
185                 <td>Description</td>
186                 <td>Authority</td>
187                 <td>Date Uploaded</td>
188                 <td>Version</td>
189                 <td>OpenMRS Version</td>
190                 <td>Downloads</td>
191             </tr>
192         </thead>
193     </table>
194 </div>
195 <div id="fourthtab">
196     <table id="tab4table" class="tabtable" border = "5px">
197         <thead>
198             <tr><td>Name</td>
199                 <td>Description</td>
200                 <td>Authority</td>
201                 <td>Date Uploaded</td>
202                 <td>Version</td>
203                 <td>OpenMRS Version</td>
204                 <td>Downloads</td>
205             </tr>
206         </thead>
207     </table>
208 </div>
209 <div id="fifthtab">
210     <table id="tab5table" class="tabtable" border = "5px">
211         <thead>
212             <tr><td>Name</td>
213                 <td>Description</td>
214                 <td>Authority</td>
215                 <td>Date Uploaded</td>
216                 <td>Version</td>
217                 <td>OpenMRS Version</td>
218                 <td>Downloads</td>
219             </tr>
220         </thead>
221     </table>
222 </div>
223 </div>
224
225
226
227 <?php
228 }
229
230 // Takes as parameters a referring page and a database query
231 // Displays the data resulting from making the query in the way appropriate
    for the referring page
232 // Outputs HTML
233 function tabcontent($query, $off)
234 { ?>
235     <script type="text/javascript">
236         function confirmDelete(name){

```

```

237         var confirmed = confirm("Are you sure? This will remove package
238             '" + name + "' forever.");
239         return confirmed;
240     }
241     </script>
242     <?php
243     /*
244         echo '<td><a href="download.php?id=' . $row['id'] . '>DL</a>&
245             nbsp';
246         if($user && ($user['class'] == 1 || ($user['id'] == $row['
247             authid'])))
248         {
249             echo '&ref=maned';
250         }
251         else
252         {
253             echo '&ref=prof';
254         }
255         echo '>ED</a>&nbsp;';
256         echo '<a href="deletePackage.php?id=' . $row['id'];
257         if($user['class'] == 1)
258         {
259             echo '&ref=man';
260         }
261         else
262         {
263             echo '&ref=prof';
264         }
265         echo '" onclick="return confirmDelete(\'' . $row['name
266             ']' . '\')">RM</a></td></tr>';
267     }
268     dbclose($conn);
269 }
270 else
271 {
272     echo 'No records found for that query.';
273 }?>
274 </tbody>
275 </table><br />
276 <?php */
277 }
278 ?>

```

F.3 utils.php

```

1 <?php
2 //General utility functions
3
4 // Open a zip archive
5 // Takes as parameter a path to the archive
6 // Outputs the extracted archive to an unzip folder in document root
7 // Returns a boolean success indicator
8 function openZip($path)
9 {
10     $zip = new ZipArchive;
11     if($zip->open($path) == true) {
12         for($i = 0; $i < $zip->numFiles; $i++) {
13             $filename = $zip->getNameIndex($i);
14             $zip->extractTo(dirname(__FILE__) . "/unzip/");
15         }
16         $zip->close();
17         return true;
18     }
19     else
20     {
21         return false;
22     }
23 }
24
25 // Adds a file to a zip archive
26 // Takes as parameters a path to the archive to add to, a path to the file to
27 // add, and the name of the file to add
28 // Returns a boolean success indicator
29 function writeZip($archpath, $filepath, $filename)
30 {
31     $zip = new ZipArchive;
32     if($zip->open($archpath) == TRUE) {
33         $zip->addFile($filepath, $filename);
34         $zip->close();
35         return true;
36     }
37     else
38     {
39         return false;
40     }
41 }
42 // Redirects the browser to a different page
43 // Takes as parameters the page url to redirect to and a delay in seconds
44 // Outputs HTML
45 function redirect($url, $delay = 0)
46 {
47     echo '<META HTTP-EQUIV="REFRESH" CONTENT="' . $delay . ' ';
48     echo '>';
49 }
50 // Validates that the current visitor to the page is logged in and has a valid
51 // session ID
52 // Returns the validated user or false if none
53 function getAuthenticatedUser()
54 {
55     $ret = false;
56     if(isset($_SESSION['email']) && isset($_SESSION['id']))
57     {

```

```
58     // Open the connection with MySQL
59     $conn = dbconnect();
60
61     $email = mysql_real_escape_string($_SESSION['email']);
62     $query = "SELECT * FROM 'users' WHERE 'email' = '$email'";
63     $result = mysql_query($query);
64     $isSesh = mysql_num_rows($result);
65     $person = mysql_fetch_array($result);
66
67     dbclose($conn);
68
69     if($isSesh == 1)
70     {
71         $ret = $person;
72     }
73 }
74 return $ret;
75 }
76
77 // helper function if you don't need user details
78 // returns the authentication status
79 function isAuthenticated()
80 {
81     return (getAuthenticatedUser() != false);
82 }
83
84
85 // Deletes a file and redirects the browser
86 // Takes as parameters a path to the file to delete and a target url to
87 // redirect to
88 function deleteFile($path, $target = 'home.php')
89 {
90     // File Exists?
91     if(file_exists($path))
92     {
93         unlink($path);
94     }
95     else
96     {
97         redirect($target, 5);
98         die('File Not Found');
99     }
100 }
101 ?>
```

F.4 Functions

dbFuncs.php

```

1 <?php
2
3 // Database functions
4
5 // Connect to the MySQL database,
6 // Returns the open connection
7 function dbconnect()
8 {
9     // Open the connection with MySQL
10    $conn = mysql_connect($GLOBALS['dbhost'], $GLOBALS['dbuser'], $GLOBALS['
11        dbpass']) or die('Error connecting to mysql');
12
13    // Connect to the database in MySQL
14    mysql_select_db($GLOBALS['dbname']);
15    return $conn;
16 }
17 // Close a connection to the MySQL database
18 // Takes as parameter a connection
19 function dbclose($conn)
20 {
21     mysql_close($conn); // Close the connection with MySQL
22 }
23 ?>
```

pwFuncs.php

```

1 <?php
2 // Password security functions
3
4 // Returns a random salt to add to the password to help discourage rainbow
5 // table attacks on a stolen db
6 function getPasswordSalt()
7 {
8     return substr(str_pad(dehex(mt_rand()), 8, '0', STR_PAD_LEFT), -8);
9 }
10 // Takes as parameters an 8-bit salt and a plaintext password
11 // Calculates and returns the hash from the salt and the password
12 function getPasswordHash($salt, $password)
13 {
14     // TODO: use other hash function if possible SHA1 has known collisions
15     return $salt . sha1($salt . $password);
16 }
17
18 // Takes as parameters a plain text password and the stored hash
19 // Compares the two parameters and returns a boolean
20 function comparePassword($password, $hash)
21 {
22     $salt = substr($hash, 0, 8);
23     return $hash == getPasswordHash($salt, $password);
24 }
25 ?>
```


F.5 home.php

```

1 <?php
2 include_once 'includes.php';
3 startPage('Home');
4 ?>
5 <script type="text/javascript">
6 var searchString, type1, type2, data;
7 $(function () {
8     "use strict";
9     $(".search_button").click(function () {
10         // getting the value that user typed
11         searchString = $("#search_box").val();
12         type1 = $("#search_name").is(':checked');
13         type2 = $("#search_auth").is(':checked');
14         // forming the queryString
15         data = 'search=' + searchString + '&type1=' + type1 + '&type2=' +
16             type2;
17         // if searchString is not empty
18         if(searchString) {
19             // ajax call
20             $.ajax({
21                 type: "POST",
22                 url: "search.php",
23                 data: data,
24                 beforeSend: function () { // this happens before actual call
25                     $("#results").html('');
26                     $("#searchresults").show();
27                     $(".word").html(searchString);
28                 },
29                 success: function (html) { // this happens after we get
30                     results
31                     $("#results").show();
32                     $("#results").append(html);
33                 }
34             });
35         }
36         return false;
37     });
38 });
39 </script>
40 </head>
41 <body>
42 <?php printHeader();?>
43 <h1>Home</h1>
44 <div id="container">
45     <div style="margin:20px auto; text-align:center;">
46         <form>
47             <input type="text" name="search" id="search_box" class="
48                 search_box" />
49             <input type="submit" value="Search" class="search_button" /><
50                 br />
51             <input type="radio" name="searchtype" value="1" id="
52                 search_name" class="search_type" checked />By Name
53             <input type="radio" name="searchtype" value="0" id="search_desc
54                 " class="search_type" />By Description
55             <input type="radio" name="searchtype" value="2" id="search_auth
56                 " class="search_type" />By Author
57         </form>

```

```
53     </div>
54     <h3>Uploaded Metadata Packages:</h3>
55     <div>
56         <div id="searchresults" style="display:none">Search results for <
            span class="word"></span>
57         </div>
58         <ul id="results" class="update">
59         </ul>
60     </div>
61 </div>
62 <?php
63     tabs();
64     $user = getAuthenticatedUser();
65     if($user)
66     {
67         echo '<h3>Welcome, '. $user['name'] . '!</h3>';
68         printNavOptions($user);
69     }
70     else
71     { ?>
72         <a href="loginPage.php">Login</a> to Upload your favorite metadata
73         package!<br />
74         New User? Register <a href="newaccount.php">Here</a><br />
75     <?php
76     }
77     printFooter();
78 ?>
79 </body>
80 </html>
```

F.6 newaccount.php

```

1 <?php
2   include_once 'includes.php';
3   startPage('New Account');
4   echo '<body>';
5   printHeader();?>
6   <script type="text/javascript" src="formvalidate.js"></script>
7   <h2>New User Registration</h2>
8   <div id="registration">
9     Please fill out the following fields: <br /><br />
10    <form name="new" action="addUser.php" onsubmit='return checkRequired("
11      new")' method="post">
12      <table>
13        <tr><td>Name:</td><td><input type="text" name="name" /></td><
14          td><div id="namecheck"></div></td></tr>
15        <tr><td>Email:</td><td><input type="text" name = "email" /></
16          td><td><div id="emailcheck"></div></td></tr>
17        <tr><td>Password:</td><td><input type ="password" name = "pass
18          " /></td><td><div id="passcheck"></div></td></tr>
19      </table>
20      <input type="submit" value="Submit" />
21    </form>
22  </div>
23  <br /><br /><br />
24  Already registered? Login <a href="loginPage.php">here</a>!<br />
25  <a href="home.php">Home</a><br />
26  <?php printFooter();?>
27 </body>
28 </html>

```

F.7 loginPage.php

```

1 <?php
2     include_once 'includes.php';
3     startPage('Login Page');
4     echo '<body>';
5     printHeader();
6     ?>
7     <h2>Login:</h2>
8     <div id="login">
9         <?php if(isset($_GET['ref']) && $_GET['ref'] == "bad")
10            {
11                echo '<p>That username/password combination was invalid.
12                    Please try again.</p>';
13            } ?>
14     <form name="new" action="login.php" onsubmit='return loginRequired("
15     new")' method="post">
16     <table>
17         <tr><td>Email:</td><td><input type="text" name = "email" /></
18         td><td><div id="emailcheck"></div></td></tr>
19         <tr><td>Password:</td><td><input type ="password" name = "pass
20         " /></td><td><div id="passcheck"></div></td></tr>
21     </table>
22     <br />
23     <input type="submit" value="Submit" />
24 </form>
25 </div>
26 <br /><br /><br />
27 New user? Register <a href="newaccount.php">here</a>!
28 <br />
29 <a href="home.php">Home</a>
30 <?php printFooter(); ?>
31 </body>
32 </html>

```

F.8 profilepage.php

```

1 <?php
2     include_once 'includes.php';
3     startPage('Your Profile');
4     echo '<body>';
5
6     printHeader();
7
8     $user = getAuthenticatedUser();
9     if($user)
10    {
11        // Open the connection with MySQL
12        $conn = dbconnect();
13
14        $auth = $user['name'];
15        $packages = $user['packages'];
16
17        $query = "SELECT * FROM 'metadata' where 'auth' = '$auth'";
18        $table = mysql_query($query);
19
20        dbclose($conn);
21
22        echo "<h3>Welcome, " . $user['name'] . "! </h3>";
23
24        if(isset($_GET['ref'])&& $_GET['ref'] == 'up')
25        {
26            echo '<h3>Thank You for Uploading a New Package!</h3>';
27        }
28        else if(isset($_GET['ref']) && $_GET['ref'] == 'rm')
29        {
30            echo '<h3>Package Removed Successfully!</h3>';
31        }
32        else if(isset($_GET['ref']) && $_GET['ref'] == 'ped')
33        {
34            echo '<h3>Package Edited Successfully!</h3>';
35        }
36        else if(isset($_GET['ref']) && $_GET['ref'] == 'ued')
37        {
38            echo '<h3>User account details Edited Successfully!</h3>';
39        }
40        printNavOptions($user);
41    ?>
42    <div id="details">
43        Name:&nbsp; <?php echo $user['name'];?><br />
44        Email Address:&nbsp; <?php echo $user['email']; ?><br />
45        Member Since:&nbsp; <?php echo $user['joindate'];?><br />
46        Packages Uploaded:&nbsp; <?php echo $user['packages'];?><br />
47        <a href="manageUsers.php?id=<?php echo $user['id']; ?>&ref=prof">
48            Edit</a><br /><br />
49        <h3>Your Uploaded Packages:</h3>
50        <?php
51        tabs($auth);
52        echo '</div>';
53    }
54    else // Invalid session
55    {
56        redirect('loginPage.php');
57    }
58    printFooter();
59 ?>
</body></html>

```

F.9 upload.php

```
1 <?php
2     include_once 'includes.php';
3     startPage('Upload Package');
4     echo '<body>';
5
6     printHeader();
7     $user = getAuthenticatedUser();
8
9     // Display if Logged in
10    if($user)
11    { ?>
12        <h3>Welcome&nbsp; <?php echo $user['name']; ?>!</h3>
13        <h3>Upload a Package to Share:</h3>
14
15        <p>Choose a file:</p>
16        <form enctype="multipart/form-data" action="uploader.php" onsubmit=""
17            method="POST">
18            <input type="hidden" name="MAX_FILE_SIZE" value="1000000" />
19            <input name="uploadedPackage" type="file" /><br />
20            <input type="submit" value="Upload Package" />
21        </form><br />
22        <?php
23        printNavOptions($user);
24    }
25    else
26    { ?>
27        <h3>You must be logged in to upload a package.</h3><br />
28        <a href="loginPage.php">Login</a><br />
29        <a href="home.php">Home</a><br />
30    }
31    <?php
32    printFooter();
33 ?>
</body></html>
```

F.10 managePackages.php

```

1 <?php
2 include_once 'includes.php';
3 startPage('Manage Packages');
4 echo '<body>';
5 printHeader();
6
7 $user = getAuthenticatedUser();
8 if($user)
9 {
10     if(isset($_POST['ref']) && $_POST['ref'] == 'prof')
11     {
12         redirect('profilepage.php');
13     }
14
15     echo "<h3>Welcome, " . $user['name'] . "! </h3>";
16     printNavOptions($user);
17     if(isset($_GET['ref']) && $_GET['ref'] == 'rm')
18     {
19         echo '<h3>Package Removed Successfully </h3>';
20     }
21     else if(isset($_GET['ref']) && $_GET['ref'] == 'ed')
22     {
23         echo '<h3>Package Edited Successfully </h3>';
24     }
25     else if(isset($_GET['ref']) && isset($_GET['id']) && ($_GET['ref'] ==
        'maned' || $_GET['ref'] == 'prof'))
26     {
27         // Open the connection with MySQL
28         $conn = dbconnect();
29
30         $edid = mysql_real_escape_string($_GET['id']);
31         $query = "SELECT * FROM 'metadata' WHERE 'id' = '$edid'";
32         $row = mysql_query($query);
33         $pack = mysql_fetch_array($row);
34         $packExists = mysql_num_rows($row);
35         dbclose($conn);
36
37         if($packExists == 1 && ($user['class'] == 1 || $pack['authid'] ==
            $user['id']))
38         { ?>
39             <h3>Editing a Package:</h3>
40             <form name="edit" action="peditor.php?ref=<?php echo $_GET['
                ref']; ?>" method="post">
41                 <input type="hidden" name="id" value=" <?php echo $edid;?>
                    " /><br />
42                 <tr><td>Package Name:</td><td><input type="text" name="
                    name" value=" <?php echo $pack['name'];?>" /> </td><br
                    /></tr>
43                 <tr><td>Description: </td><td><input type="text" name = "
                    desc" value=" <?php echo $pack['desc'];?>" /> </td><br
                    /></tr>
44                 <input type="submit" value="Save Changes" />
45             </form>
46             <p>(note: changes will only take effect if you click save)</p>
47         <?php
48         }
49     }
50     if($user['class'] == 1)

```

```
51     {
52         echo '<h3>Here are our uploaded packages:</h3>';
53         tabs();
54     }
55 }
56 else
57 {
58     echo '<p>You must be logged in to manage metadata packages! &nbsp; You
59         will be redirected to the login page in a few seconds...</p>';
60     redirect('loginPage.php', 5);
61 }
62 printFooter(); ?>
</body></html>
```


F.11 manageUsers.php

```

1 <?php
2     include_once 'includes.php';
3     startPage('Manage Users');
4     echo '<body>';
5     printHeader();
6     $user = getAuthenticatedUser();
7     if($user)
8     {
9         echo "<h3>Welcome, " . $user['name'] . "! </h3>";
10        printNavOptions($user);
11        if(isset($_GET['ref']) && $_GET['ref'] == 'ed')
12        {
13            echo '<h3>User Account Details Edited Successfully</h3>';
14        }
15        else if(isset($_GET['ref']) && $_GET['ref'] == 'rm')
16        {
17            echo '<h3>User Account Removed Successfully</h3>';
18        }
19        else if(isset($_GET['ref']) && $_GET['ref'] == 'edf')
20        {
21            echo '<h3>There was an error editing user account details</h3>';
22        }
23        else if(isset($_GET['id']) && isset($_GET['ref']) && $_GET['ref'] == '
24            man')
25        {
26            // Open the connection with MySQL
27            $conn = dbconnect();
28            $sedid = mysql_real_escape_string($_GET['id']);
29            $query = "SELECT * FROM 'users' WHERE 'id' = '$sedid'";
30            $row = mysql_query($query);
31            $edu = mysql_fetch_array($row);
32            $eduExists = mysql_num_rows($row);
33
34
35            if($eduExists == 1 && ($user['class'] == 1))
36            {
37                $pw = getPasswordHash(getPasswordSalt(), 'MDSSUser-' . $edu['
38                    name']);
39                $query = "UPDATE 'users' SET 'pw' = '$pw' WHERE 'id' = '$sedid'
40                    ";
41                $row = mysql_query($query);
42            }
43            dbclose($conn);
44            echo '<h3>Password reset successfully.</h3>';
45        }
46        else if(isset($_GET['id']) && isset($_GET['ref']) && ($_GET['ref'] ==
47            'maned' || $_GET['ref'] == 'prof'))
48        // TODO: validate new email changes with javascript
49        {
50            // Open the connection with MySQL
51            $conn = dbconnect();
52            $sedid = mysql_real_escape_string($_GET['id']);
53            $query = "SELECT * FROM 'users' WHERE 'id' = '$sedid'";
54            $row = mysql_query($query);
55            $edu = mysql_fetch_array($row);
56            $eduExists = mysql_num_rows($row);

```

```

55     dbclose($conn);
56
57     if($SeduExists == 1 && ($user['class'] == 1 || $Sedu['id'] == $user[
58         'id']))
59     {
60         // TODO: javascript to verify
61         ?>
62         <h3>Editing User&nbsp;<?php echo $Sedu['id'];?>: </h3>
63         <form name="edit" action="ueditor.php?ref=<?php echo $_GET
64             ['ref'];?>" onsubmit="return checkRequired("new")"
65             method="post">
66             <input type="hidden" name="id" value=" <?php echo
67                 $Sedu['id'];?>" /><br />
68             <tr><td>Name:</td><td><input type="text" name="name"
69                 value="<?php echo $Sedu['name'];?>" /> </td><td><
70                 div id="namecheck"> </div></td></tr>
71             <tr><td>Email: </td><td><input type="text" name = "
72                 email" value="<?php echo $Sedu['email'];?>" /> </td>
73                 <td><div id="emailcheck"> </div></td></tr>
74                 <br />
75                 <?php
76                 if($user['class'] == 0)
77                 { ?>
78                     <tr><td>Current Password: </td><td><input type="
79                         password" name = "pass" /> </td><td><div id="
80                         passcheck"> </div></td></tr>
81
82                 <?php
83                 } ?>
84                 <tr><td>New Password: </td><td><input type="password"
85                     name = "newpw" /> </td></tr>
86                 <tr><td>Confirm New Password: </td><td><input type="
87                     password" name = "newpw2" /> </td></tr>
88                 <?php if($user['class'] == 1)
89                 { ?>
90                     <tr><td>User Class: <br /></td><td>
91                     <input type="radio" name = "class" value="admin"
92                         <?php if($Sedu['class'] == 1) echo 'checked';?>
93                         />Admin</td></tr>
94                     <input type="radio" name = "class" value="user"<?
95                         php if($Sedu['class'] == 0) echo 'checked';?>
96                         />User</td></tr>
97
98                 <?php
99                 } ?>
100                <input type="submit" value="Save Changes" />
101                <a href="<?php
102                    if($_GET['ref'] == 'maned')
103                    {
104                        echo 'manageUsers';
105                    }
106                    else if($_GET['ref'] == 'prof')
107                    {
108                        echo 'profilepage';
109                    }
110                ?>.php"><input type="button" name="cancel" value="
111                    Cancel" /></a>
112            </form>
113            <p>(note: changes will only take effect if you click save)
114            </p>
115        <?php
116    }

```

```

97     }
98
99     if($user['class'] == 1 && !isset($_GET['ref']) || ($_GET['ref'] != '
        maned' && $_GET['ref'] != 'prof' )
100     { ?>
101         <script type="text/javascript">
102             function confirmDelete(name){
103                 var confirmed = confirm("Are you sure? This will remove user '"
                    + name + "' and all their packages forever.");
104                 return confirmed;
105             }
106     </script>
107         Here are our registered users:<br />
108         <table border = "5px">
109             <tr><td> name </td>
110             <td> email </td>
111             <td> class </td>
112             <td> join date</td>
113             <td> packages </td>
114             <td> Actions</td></tr>
115
116             <?php
117                 // Open the connection with MySQL
118                 $conn = dbconnect();
119                 $query = "SELECT * FROM 'users '";
120                 $table = mysql_query($query);
121                 while($row = mysql_fetch_array($table))
122                 {
123                     echo '<td>' . $row['name'] . '</td>';
124                     echo '<td>' . $row['email'] . '</td>';
125                     if($row['class'] == 1)
126                     {
127                         echo '<td>Admin</td>';
128                     }
129                     else
130                     {
131                         echo '<td>User</td>';
132                     }
133                     echo '<td>' . $row['joindate'] . '</td>';
134                     echo '<td>' . $row['packages'] . '</td>';
135                     echo '<td><a href="manageUsers.php?id=' . $row['id'] . '&
                        ref=man">Reset PW</a>&nbsp;';
136                     echo '<a href="manageUsers.php?id=' . $row['id'] . '&ref=
                        maned">Edit</a>&nbsp;';
137                     echo '<a href="removeUser.php?id=' . $row['id'] . '&ref=
                        manrm" onclick="return confirmDelete(\'' . $row['name'
                        ] . '\')">Remove</a></td></tr>';
138                 }
139                 dbclose($conn);
140             echo '</table>';
141         }
142     }
143 }
144 else
145 {
146     // No such user or corrupt session
147     redirect('loginPage.php');
148 }
149 printFooter();
150 ?>
151 </body></html>

```

F.12 search.php

```

1 <?php
2
3 // if we got something through $_POST
4 if(isset($_POST['search']) && isset($_POST['type1']) && isset($_POST['type2']))
5     {
6         include_once 'includes.php';
7         // Open the connection with MySQL
8         $conn = dbconnect();
9         $word = '%' . mysql_real_escape_string($_POST['search']) . '%';
10        dbclose($conn);
11
12        if($_POST['type1'] == 'true')
13        {
14            $query = "SELECT * FROM 'metadata' WHERE 'name' LIKE '$word' ORDER BY
15                    'dls' DESC LIMIT 10";
16        }
17        else if($_POST['type2'] == 'true')
18        {
19            $query = "SELECT * FROM 'metadata' WHERE 'auth' LIKE '$word' ORDER BY
20                    'dls' ";
21        }
22        else
23        {
24            $query = "SELECT * FROM 'metadata' WHERE 'desc' LIKE '$word' ORDER BY
25                    'dls' DESC LIMIT 10";
26        }
27        tabcontent('main', $query);
28
29        /*
30        // code for bolding the search term in a result
31        if($isEntry >= 1 && isset($row)) {
32            $send_result = '<td>';
33            foreach($row as $r) {
34                $res = $r['name'];
35                // bold the search word in result
36                $bold = '<span class="found">' . $word . '</span>';
37                $send_result .= '<tr>' . str_ireplace($word, $bold, $res) . '</tr>
38                >';
39            }
40            $send_result .= '</td>';
41            echo $send_result;
42        } else {
43            echo '<li>No results found</li>';
44        }
45        */
46    }
47 }
48 ?>

```

F.13 download.php

```
1 <?php
2 include_once 'includes.php';
3 startPage('Download');
4 echo '<body>';
5 function downloadFile($fullPath, $filename)
6 {
7     // Must be fresh start
8     if(headers_sent())
9     {
10         die('headers Sent');
11     }
12
13     // Required for some browsers
14     if(ini_get('zlib.output_compression'))
15     {
16         ini_set('zlib.output_compression', 'Off');
17     }
18
19     // File Exists?
20     if(file_exists($fullPath))
21     {
22         // Parse Info / Get Extension
23         $fsize = filesize($fullPath);
24         $ctype="application/x-zip-compressed";
25
26         header("Pragma: public"); // required
27         header("Expires: 0");
28         header("Server: ");
29         header("X-Powered-By: ");
30         header("Cache-Control: must-revalidate, post-check=0, pre-check=0");
31         header("Cache-Control: private", false); // required for certain browsers
32         header("Content-Type: " . $ctype);
33         header("Content-Disposition: attachment; filename=\"\" . $filename . "\"");
34
35         header("Content-Transfer-Encoding: binary");
36         header("Content-Length: " . $fsize);
37         ob_clean();
38         flush();
39         readfile($fullPath);
40     }
41     else
42     {
43         die('File Not Found');
44     }
45 }
46 if(isset($_GET['id']))
47 {
48     // Open the connection with MySQL
49     $conn = dbconnect();
50
51     $id = mysql_real_escape_string($_GET['id']);
52     $query = "SELECT * FROM 'metadata' WHERE 'id' = '$id'";
53     $table = mysql_query($query);
54     $isEntry = mysql_num_rows($table);
55     $row = mysql_fetch_array($table);
56     $dls = $row['dls'] + 1;
57
58     if($isEntry == 1)
```

```
59     {
60         downloadFile($row['path'], $row['name']);
61         $query = "UPDATE 'metadata' SET 'dls' = '$dls' WHERE 'id' = '$id'";
62         $result = mysql_query($query);
63     }
64     dbclose($conn);
65 }
66 ?>
67 </body></html>
```

F.14 addUser.php

```

1 <?php
2     include_once 'includes.php';
3     startPage('User Registration');
4     echo '<body>';
5     printHeader();
6
7     if(isset($_POST['name']) && isset($_POST['email']) && isset($_POST['pass'
8         ]))
9     {
10         // Open the connection with MySQL
11         $conn = dbconnect();
12         $name = trim(mysql_real_escape_string($_POST['name']));
13         $email = trim(mysql_real_escape_string($_POST['email']));
14         $pass = trim($_POST['pass']);
15
16         if($email == '')
17         {
18             echo '<p>Please enter an email address! &nbsp; You will be
19                 redirected to the previous page in a few seconds...</p>';
20             redirect('newaccount.php', 5);
21         }
22         else if($pass == '')
23         {
24             echo '<p>Please enter a password! &nbsp; You will be redirected to
25                 the previous page in a few seconds...</p>';
26             redirect('newaccount.php', 5);
27         }
28         else if($name == '')
29         {
30             echo '<p>Please enter a username! &nbsp; You will be redirected to
31                 the previous page in a few seconds...</p>';
32             redirect('newaccount.php', 5);
33         }
34         else
35         {
36             // Check if the user already exists in the table
37             $query = "SELECT * FROM 'users' WHERE 'email' = '$email'";
38             $result_user_check = mysql_query($query);
39             $isEntry = mysql_num_rows($result_user_check);
40
41             // Add user if not already in the database
42             if($isEntry == 0)
43             {
44                 // get a new hash for a password
45                 $hash = getPasswordHash(getPasswordSalt(), $pass);
46
47                 $defaultPackages = 0;
48                 $joinDate = date(time());
49
50                 $query = "INSERT INTO 'users' ('name', 'email', 'pw', 'class',
51                     'joindate', 'packages') VALUES ('$name', '$email', '$hash
52                     ', '0', FROM_UNIXTIME('$joinDate'), '$defaultPackages')";
53                 mysql_query($query);
54
55                 echo '<p>New User Added! Try to log in now <a href="loginPage.
56                     php">here</a><p>';
57             }
58         }
59     }
60 }
61
62 else
63 { ?>

```

```
54         <p>A user account with that email address already exists.  
          Please log-in <a href="loginPage.php">here</a>.</p>  
55     <p>If you forgot your password click <a href="pwRecovery.php">  
          here</a> to recover.</p>  
56     <p>If you would like to create a new user account please  
          choose a different email <a href="newaccount.php">here</a  
          >.</p>  
57     <?php  
58         }  
59     }  
60     dbclose($conn);  
61     }  
62     else  
63     {  
64         redirect('home.php');  
65     }  
66     printFooter();  
67     ?>  
68 </body>  
69 </html>
```


F.15 login.php

```
1 <?php
2     Session_start();
3     include_once 'includes.php';
4     $val = isAuthenticated();
5     if($val) // Someone is already logged in - log them out
6     {
7         // Unset all of the session variables.
8         $_SESSION = array();
9
10        Session_destroy();
11    }
12    ob_start();
13    startPage('Login');
14    ob_end_clean();
15    echo '<body>';
16
17    // Open the connection with MySQL
18    $conn = dbconnect();
19    $email = mysql_real_escape_string($_POST['email']);
20    $pass = $_POST['pass'];
21
22    $query = "SELECT * FROM 'users' WHERE 'email' = '$email'";
23    $hash = mysql_query($query);
24    $isEntry = mysql_num_rows($hash);
25    $row = mysql_fetch_array($hash);
26
27    if(($isEntry != 1) || !comparePassword($pass, $row['pw']))
28    {
29        redirect('loginPage.php?ref=bad');
30    }
31    else
32    {
33        $seshid = getPasswordHash(getPasswordSalt(), date(time()));
34
35        $_SESSION['id'] = $seshid;
36        $_SESSION['name'] = $row['name'];
37        $_SESSION['email'] = $email;
38
39        redirect('profilepage.php');
40    }
41    dbclose($conn);
42 ?>
43 </body>
44 </html>
```

F.16 uploader.php

```

1 <?php
2 include_once 'includes.php';
3 startPage('Uploader');
4 echo '<body>';
5
6 $person = getAuthenticatedUser();
7
8 if($person)
9 {
10     $okay = false;
11     $accepted_types = array('application/zip', 'application/x-zip-compressed',
12                             'multipart/x-zip', 'application/s-compressed');
13
14     if(isset($_FILES))
15     {
16         $file = $_FILES["uploadedPackage"];
17     }
18     else
19     {
20         redirect('home.php');
21         die("No file.");
22     }
23     $type = $file["type"];
24     foreach($accepted_types as $mime_type) {
25         if($mime_type == $type) {
26             $okay = true;
27             break;
28         }
29     }
30     if($okay && ($file["size"] < 1000000))
31     {
32         if($file["error"] > 0)
33         {
34             echo "Error: " . $file["error"] . "<br />";
35         }
36         else
37         {
38             echo "Upload: " . $file["name"] . "<br />";
39             echo "Type: " . $file["type"] . "<br />";
40             echo "Size: " . ($file["size"] / 1024) . " Kb<br />";
41             $blacklist = array(".php", ".phtml", ".php3", ".php4");
42             foreach ($blacklist as $item)
43             {
44                 if(preg_match("/$item$/i", $file['name']))
45                 {
46                     echo "We do not allow uploading PHP files\n";
47                     exit;
48                 }
49             }
50             // add new generated filename to path
51             $target_path = str_replace('\\', '/', $GLOBALS['uploadpath']) .
52                 getPasswordHash(getPasswordSalt(), $file['name']) . ".zip";
53
54             if(move_uploaded_file($file['tmp_name'], $target_path))
55             {
56                 echo "The file " . basename($file['name']) . " has been
57                 uploaded";

```

```

58 // Extract the package info from the header.xml file
59 // TODO: handle gracefully if this fails
60 unzip($target_path);
61
62 // open the metadata.xml file
63 // we don't really care about the contents
64 // just check that it isn't corrupted.
65 $xml = dirname(__FILE__) . '/unzip/metadata.xml';
66 ob_start();
67 $x = simplexml_load_file($xml);
68 ob_end_clean();
69
70 if($x == FALSE || empty($x))
71 {
72     deleteFile($target_path);
73     redirect('home.php', 5);
74     die('<br /><h3>WARNING! Bad xml file. Please check your
75         file and try again.</h3>');
76 }
77 // now open the header.xml file which contains the info we
78 // want
79 $xml = dirname(__FILE__) . '/unzip/header.xml';
80 ob_start();
81 $x = simplexml_load_file($xml);
82 ob_end_clean();
83
84 if($x == FALSE || empty($x))
85 {
86     deleteFile($target_path);
87     redirect('home.php', 5);
88     die('<br /><h3>WARNING! Bad xml file. Please check your
89         file and try again.</h3>');
90 }
91 else
92 {
93     // Open the connection with MySQL
94     $conn = dbconnect();
95
96     // $uuid = $x->attributes()->uuid;
97     $pubdate = mysql_real_escape_string($x->dateCreated);
98     $name = mysql_real_escape_string($x->name);
99     $desc = mysql_real_escape_string($x->description);
100     $omrsver = mysql_real_escape_string($x->openmrsVersion);
101     $ver = mysql_real_escape_string($x->version);
102     // $group = mysql_real_escape_string($x->group);
103     $suburl = mysql_real_escape_string($x->subscriptionUrl);
104     // TODO: capture MDS version here
105
106     $auth = mysql_real_escape_string($person['name']);
107     $authid = $person['id'];
108     $update = date(time());
109
110     $query = "SELECT * from 'metadata' WHERE 'name' = '$name'
111         AND 'omrsver' = '$omrsver' AND 'ver' = '$ver'";
112     $row = mysql_query($query);
113     $isEntry = mysql_num_rows($row);
114     if($isEntry >= 1)
115     {
116         deleteFile($target_path);
117         redirect('upload.php', 5);
118         die("<h3>We're sorry that package has already been
119             shared. Please try a different package.</h3>");
120     }
121 }

```

```
116     }
117
118     $query = "INSERT INTO 'metadata' ('name', 'path', 'auth',
119     'authid', 'pubdate', 'update', 'desc', 'omrsver', 'ver',
120     'suburl', 'dls')";
121     $query .= "VALUES ('$name', '$target_path', '$auth', '$authid', '$pubdate', FROM_UNIXTIME('$update'), '$desc', '$omrsver', '$ver', '$suburl', '0')";
122     mysql_query($query);
123     $query = "UPDATE 'users' SET 'packages' = 'packages'+1
124     WHERE 'id' = '$authid'";
125     mysql_query($query);
126     dbclose($conn);
127     }
128     redirect('profilePage.php?ref=up', 5);
129     }
130     else
131     {
132     echo '<p>There was an error uploading the file, please try
133     again!</p>';
134     redirect('home.php', 5);
135     }
136     }
137     }
138     else
139     {
140     echo '<p>Invalid file. Please try again.</p>';
141     redirect('home.php', 5);
142     }
143     }
144     ?>
</body>
</html>
```

F.17 peditor.php

```

1 <?php
2     ini_set('display_errors',1);
3     error_reporting(E_ALL);
4
5     Session_start();
6     include_once 'includes.php';
7
8     $user = getAuthenticatedUser();
9     // if Logged in
10    if($user)
11    {
12        if(isset($_POST['id']))
13        {
14            // Open the connection with MySQL
15            $conn = dbconnect();
16
17            $pid = mysql_real_escape_string($_POST['id']);
18            $query = "SELECT * FROM 'metadata' WHERE 'id' = '$pid'";
19            $row = mysql_query($query);
20            $pack = mysql_fetch_array($row);
21            $packExists = mysql_num_rows($row);
22
23            // close the connection with MySQL
24            dbclose($conn);
25
26            if($packExists == 1 && ($user['class'] == 1 || $pack['auth'] ==
                $user['name']))
27            {
28                if(isset($_POST['name']) && isset($_POST['desc']))
29                {
30                    // Open the connection with MySQL
31                    $conn = dbconnect();
32
33                    $name = trim(mysql_real_escape_string($_POST['name']));
34                    $desc = trim(mysql_real_escape_string($_POST['desc']));
35
36                    $ver = $pack['ver'];
37                    $omrsver = $pack['omrsver'];
38
39                    $query = "SELECT * from 'metadata' WHERE 'name' = '$name'
                        AND 'ver' = '$ver' AND 'omrsver' = '$omrsver'";
40                    $row = mysql_query($query);
41                    $isEntry = mysql_num_rows($row);
42                    if($isEntry >= 1 && $pack['name'] != $name)
43                    {
44                        if(isset($_GET['ref']))
45                        {
46                            if($_GET['ref'] == 'prof')
47                            {
48                                redirect('profilepage.php', 5);
49                            }
50                            else if($_GET['ref'] == 'maned')
51                            {
52                                redirect('managePackages.php', 5);
53                            }
54                        }
55                        else
56                        {
57                            redirect('home.php', 5);
58                        }
59                    }
60                }
61            }
62        }
63    }

```

```

59         die("We're sorry a package with that name has already
60             been shared. Please try a different name.");
61     }
62     $query = "UPDATE 'metadata' SET 'name' = '$name', 'desc'
63             = '$desc' WHERE 'id' = '$pid'";
64     mysql_query($query);
65     // close the connection with MySQL
66     dbclose($conn);
67
68     $target_path = $pack['path'];
69
70     openZip($target_path);
71
72     $xml = dirname(__FILE__) . '/unzip/header.xml';
73     $x = simplexml_load_file($xml);
74
75     // TODO: decide which other fields to edit here
76     $x->name = $name;
77     $x->description = $desc;
78
79     if(file_put_contents($xml, $x->asXML()) == false)
80     {
81         redirect('managePackages.php', 5);
82         die('Failed to edit xml.');
```

```

83     }
84
85     writeZip($target_path, $xml, 'header.xml');
86 }
87 }
88 if(isset($_GET['ref']))
89 {
90     if($_GET['ref'] == 'prof')
91     {
92         redirect('profilepage.php?ref=ped');
```

```

93     }
94     else if($_GET['ref'] == 'maned')
95     {
96         redirect('managePackages.php?ref=ed');
```

```

97     }
98 }
99 else
100 {
101     redirect('home.php');
```

```

102 }
103 }
104 }
105 else
106 {
107     redirect('home.php');
```

```

108 }
109 ?>
```

F.18 ueditor.php

```

1 <?php
2     ini_set('display_errors',1);
3     error_reporting(E_ALL);
4
5     Session_start();
6     include_once 'includes.php';
7
8     $user = getAuthenticatedUser();
9     if($user) // if Logged in
10    {
11        if(isset($_POST['id']))
12        {
13            // Open the connection with MySQL
14            $conn = dbconnect();
15
16            $uid = mysql_real_escape_string($_POST['id']);
17            $query = "SELECT * FROM 'users' WHERE 'id' = '$uid'";
18            $row = mysql_query($query);
19            $edu = mysql_fetch_array($row);
20            $uExists = mysql_num_rows($row);
21
22            if($uExists == 1 && ($user['class'] == 1 || $edu['id'] == $user['
23                id']))
24            {
25                if(isset($_POST['name']) && isset($_POST['email']) && isset(
26                    $_POST['newpw']) && isset($_POST['newpw2']))
27                {
28                    if($_POST['newpw'] != $_POST['newpw2'])
29                    {
30                        if(isset($_GET['ref']))
31                        {
32                            if($_GET['ref'] == 'prof')
33                            {
34                                redirect('profilepage.php', 5);
35                            }
36                            else if($_GET['ref'] == 'maned')
37                            {
38                                redirect('manageUsers.php', 5);
39                            }
40                        }
41                        else
42                        {
43                            redirect('home.php', 5);
44                        }
45                    }
46                    die("All passwords must match. Please check your
47                        typing and try again.");
48                }
49                $name = trim(mysql_real_escape_string($_POST['name']));
50                $email = trim(mysql_real_escape_string($_POST['email']));
51                $query = "SELECT * from 'users' WHERE 'name' = '$name'";
52                $row = mysql_query($query);
53                $isEntryN = mysql_num_rows($row);
54                $query = "SELECT * from 'users' WHERE 'email' = '$email'";
55                $row = mysql_query($query);
56                $isEntryE = mysql_num_rows($row);

```

```

57     if(($isEntryN >= 1 && ($name != $edu['name'])) || (
58         $isEntryE >= 1 && ($email != $edu['email'])))
59     {
60         if(isset($_GET['ref']))
61         {
62             if($_GET['ref'] == 'prof')
63             {
64                 redirect('profilepage.php', 5);
65             }
66             else if($_GET['ref'] == 'maned')
67             {
68                 redirect('manageUsers.php', 5);
69             }
70         }
71         else
72         {
73             redirect('home.php', 5);
74         }
75         die("We're sorry a user with that name or email
76             already exists. Please try a different name or
77             email.");
78     }
79     else
80     {
81         $authid = $edu['id'];
82         $query = "UPDATE 'metadata' SET 'auth' = '$name' WHERE
83             'authid' = '$authid'";
84         mysql_query($query);
85         $query = "UPDATE 'users' SET 'name' = '$name', 'email'
86             = '$email' WHERE 'id' = '$uid'";
87         mysql_query($query);
88         if($_POST['newpw'] != '')
89         {
90             if($user['class'] == 1 || (isset($_POST['pass'])
91                 && (comparePassword(trim($_POST['pass']),
92                     $user['pw']) == TRUE)))
93             {
94                 $newpw = getPasswordHash(getPasswordSalt(),
95                     trim($_POST['newpw']));
96                 $query = "UPDATE 'users' SET 'pw' = '$newpw'
97                     WHERE 'id' = '$uid'";
98                 mysql_query($query);
99             }
100             else
101             {
102                 if(isset($_GET['ref']))
103                 {
104                     if($_GET['ref'] == 'prof')
105                     {
106                         redirect('profilepage.php', 5);
107                     }
108                     else if($_GET['ref'] == 'maned')
109                     {
110                         redirect('manageUsers.php', 5);
111                     }
112                 }
113             }
114         }
115     }

```



```

109         redirect('home.php', 5);
110     }
111     die("All passwords must match. Please check
112         your typing and try again.");
113 }
114 if($user['class'] == 1 && isset($_POST['class']))
115 {
116     $class = ($_POST['class'] == 'admin') ? 1 : 0;
117     $query = "UPDATE 'users' SET 'class' = '$class'
118             WHERE 'id' = '$uid'";
119     mysql_query($query);
120 }
121 }
122 }
123 // close connection to MySQL
124 dbclose($conn);
125 if(isset($_GET['ref']) && $_GET['ref'] == 'maned')
126 {
127     redirect('manageUsers.php?ref=ed');
128 }
129 // TODO: indicate failures here
130 else if(isset($_GET['ref']) && $_GET['ref'] == 'maned' && false)
131 {
132     redirect('manageUsers.php?ref=edf');
133 }
134 else if(isset($_GET['ref']) && $_GET['ref'] == 'prof')
135 {
136     redirect('profilepage.php?ref=ued');
137 }
138 else
139 {
140     redirect('home.php');
141 }
142 }
143 }
144 }
145 else
146 {
147     redirect('home.php');
148 }
149 ?>

```

F.19 deletePackage.php

```

1 <?php
2 include_once 'includes.php';
3 startPage('Deleting Package...');
4 echo '<body>';
5
6 if(isset($_GET['id']))
7 {
8     $person = getAuthenticatedUser();
9
10    if($person)
11    {
12        // Open the connection with MySQL
13        $conn = dbconnect();
14
15        $id = mysql_real_escape_string($_GET['id']);
16        $query = "SELECT * FROM 'metadata' WHERE 'id' = '$id'";
17        $table = mysql_query($query);
18        $isEntry = mysql_num_rows($table);
19        $row = mysql_fetch_array($table);
20        dbclose($conn);
21
22        if($isEntry == 1 && ($row['auth'] == $person['name'] || $person['class
23            ''] == 1))
24        {
25            if(isset($_GET['ref']) && $_GET['ref'] == 'prof')
26            {
27                deleteFile($row['path'], 'profilePage.php');
28            }
29            else if(isset($_GET['ref']) && $_GET['ref'] == 'man')
30            {
31                deleteFile($row['path'], 'managePackages.php');
32            }
33            else
34            {
35                deleteFile($row['path']);
36            }
37            $conn = dbconnect();
38            $query = "DELETE FROM 'metadata' WHERE 'id' = '$id'";
39            $result = mysql_query($query);
40
41            $id = $row['authid'];
42            $query = "UPDATE 'users' SET 'packages' = 'packages'-1 WHERE 'id'
43                = '$id'";
44            $result = mysql_query($query);
45            dbclose($conn);
46        }
47    }
48    if(isset($_GET['ref']) && $_GET['ref'] == 'prof')
49    {
50        redirect('profilePage.php?ref=rm');
51    }
52    else if(isset($_GET['ref']) && $_GET['ref'] == 'man')
53    {
54        redirect('managePackages.php?ref=rm');
55    }
56    else
57    {
58        redirect('home.php');

```

```
59     }
60   }
61   else
62   {
63     redirect ( 'home.php' );
64   }
65   ?>
66 </body></html>
```

F.20 removeUser.php

```

1 <?php
2     include_once 'includes.php';
3     startPage('Remove User');
4     echo '<body>';
5
6     $person = getAuthenticatedUser();
7     if(isset($_GET['id']) && $person)
8     {
9         if($person['class'] == 1)
10        {
11            // Open the connection with MySQL
12            $conn = dbconnect();
13
14            $id = mysql_real_escape_string($_GET['id']);
15
16            $query = "SELECT * FROM 'users' WHERE 'id' = '$id'";
17            $tmp = mysql_query($query);
18            $isAuth = mysql_num_rows($tmp);
19            $user = mysql_fetch_array($tmp);
20
21            $auth = $user['name'];
22
23            if($isAuth == 1)
24            {
25                $query = "SELECT * FROM 'metadata' WHERE 'auth' = '$auth'";
26                $table = mysql_query($query);
27                $isEntry = mysql_num_rows($table);
28
29                if($isEntry >= 1)
30                {
31                    while($row = mysql_fetch_array($table))
32                    {
33                        deleteFile($row['path']);
34                        $pid = $row['id'];
35                        $query = "DELETE FROM 'metadata' WHERE 'id' = '$pid'";
36                        $result = mysql_query($query);
37                    }
38                }
39                $query = "DELETE FROM 'users' WHERE 'id' = '$id'";
40                $result = mysql_query($query);
41            }
42            dbclose($conn);
43        }
44        if(isset($_GET['ref']) && $_GET['ref'] == 'manrm')
45        {
46            redirect('manageUsers.php?ref=rm');
47        }
48        else
49        {
50            redirect('home.php');
51        }
52    }
53 ?>
54 </body></html>

```

F.21 Exit pages

logout.php

```

1 <?php
2     include_once 'includes.php';
3     startPage('Logging Out...');
4     echo '<body>';
5     printHeader();
6     $val = isAuthenticated();
7     if($val)
8     {
9         // Unset all of the session variables.
10        $_SESSION = array();
11
12        session_destroy();
13    }
14    ?>
15    <div id="banner">
16        <p>You have successfully logged out. &nbsp; You will be redirected to
17            the main page in a few seconds... </p>
18    </div>
19    <?php
20        printFooter();
21        redirect('home.php', 3); ?>
22 </body></html>

```

pwRecovery.php

```

1 <?php
2     startPage('Password Recovery');
3     echo '<body>';
4     printHeader();?>
5     <div id="registration">
6         <p>Metadata Sharing Server does not offer password recovery services
7             at this time! Please email an admin to reset your password.</p>
8         <p>Enter your email:</p>
9         <form name="new" action="" method="post">
10            <table>
11                <tr><td>Email:</td><td><input type="text" name = "email"
12                    /></td></tr>
13            </table>
14            <input type="submit" value="Submit" />
15        </form>
16    </div><br />
17    <?php
18        $user = getAuthenticatedUser();
19        if($user)
20        {
21            printNavOptions($user);
22        }
23        else
24        { ?>
25            Login <a href="loginPage.php">here</a>!<br />
26            <a href="home.php">Home</a>
27        <?php }
28        printFooter();?>
29 </body></html>

```



```

55     if ( isset( $_GET[ 'iSortCol_0' ] ) )
56     {
57         $sOrder = "ORDER BY ";
58         for ( $i=0 ; $i<intval( $_GET[ 'iSortingCols' ] ) ; $i++ )
59         {
60             if ( $_GET[ 'bSortable_' . intval( $_GET[ 'iSortCol_' . $i ] )
61                 ] == "true" )
62             {
63                 $sOrder .= " " . $aColumns[ intval( $_GET[ '
64                     iSortCol_' . $i ] ) ] . " " .
65                     mysql_real_escape_string( $_GET[ '
66                         sSortDir_' . $i ] ) . ", ";
67             }
68         }
69         $sOrder = substr_replace( $sOrder, "", -2 );
70         if ( $sOrder == "ORDER BY" )
71         {
72             $sOrder = "";
73         }
74     }
75     /*
76     * Filtering
77     * NOTE this does not match the built-in DataTables filtering which
78     * does it
79     * word by word on any field. It's possible to do here, but concerned
80     * about efficiency
81     * on very large tables, and MySQL's regex functionality is very
82     * limited
83     */
84     $sWhere = "";
85     if ( isset( $_GET[ 'sSearch' ] ) && $_GET[ 'sSearch' ] != "" )
86     {
87         $sWhere = "WHERE (";
88         for ( $i=0 ; $i<count( $aColumns ) ; $i++ )
89         {
90             $sWhere .= " " . $aColumns[ $i ] . " LIKE '%" .
91                 mysql_real_escape_string( $_GET[ 'sSearch' ] ) . "%'
92                 OR ";
93         }
94         $sWhere = substr_replace( $sWhere, "", -3 );
95         $sWhere .= ')';
96     }
97     /* Individual column filtering */
98     for ( $i=0 ; $i<count( $aColumns ) ; $i++ )
99     {
100         if ( isset( $_GET[ 'bSearchable_' . $i ] ) && $_GET[ 'bSearchable_' .
101             $i ] == "true" && $_GET[ 'sSearch_' . $i ] != '' )
102         {
103             if ( $sWhere == "" )
104             {
105                 $sWhere = "WHERE ";
106             }
107             else
108             {
109                 $sWhere .= " AND ";
110             }
111             $sWhere .= " " . $aColumns[ $i ] . " LIKE '%" .
112                 mysql_real_escape_string( $_GET[ 'sSearch_' . $i ] ) . "%'

```

```

107         };
108     }
109
110
111     /*
112     * SQL queries
113     * Get data to display
114     */
115     $sQuery = "
116         SELECT SQL_CALC_FOUND_ROWS ".str_replace(" ", " ", implode(
117             "' ', '", $aColumns))." '
118         FROM   $sTable
119         $sWhere
120         $sOrder
121         $sLimit
122         ";
123     $rResult = mysql_query( $sQuery, $gaSql['link'] ) or die(mysql_error()
124     );
125     /* Data set length after filtering */
126     $sQuery = "
127         SELECT FOUND_ROWS()
128     ";
129     $rResultFilterTotal = mysql_query( $sQuery, $gaSql['link'] ) or die(
130     mysql_error());
131     $aResultFilterTotal = mysql_fetch_array($rResultFilterTotal);
132     $iFilteredTotal = $aResultFilterTotal[0];
133     /* Total data set length */
134     $sQuery = "
135         SELECT COUNT('".$sIndexColumn."')
136         FROM   $sTable
137     ";
138     $rResultTotal = mysql_query( $sQuery, $gaSql['link'] ) or die(
139     mysql_error());
140     $aResultTotal = mysql_fetch_array($rResultTotal);
141     $iTTotal = $aResultTotal[0];
142
143     /*
144     * Output
145     */
146     $output = array(
147         "sEcho" => intval($_GET['sEcho']),
148         "iTotalRecords" => $iTTotal,
149         "iTotalDisplayRecords" => $iFilteredTotal,
150         "aaData" => array()
151     );
152     while ( $aRow = mysql_fetch_array( $rResult ) )
153     {
154         $row = array();
155         for ( $i=0 ; $i<count($aColumns) ; $i++ )
156         {
157             if ( $aColumns[$i] == "version" )
158             {
159                 /* Special output formatting for 'version'
160                 column */
161                 $row[] = ($aRow[ $aColumns[$i] ]=="0") ? '-' :
162                     $aRow[ $aColumns[$i] ];
163             }
164             else if ( $aColumns[$i] != ' ' )
165             {

```



```
164                                     /* General output */
165                                     $row [] = $aRow[ $aColumns[$i] ];
166                                     }
167                                 }
168                                 $output['aaData'][] = $row;
169                             }
170                             }
171                             echo json_encode( $output );
172 ?>
```

F.23 Javascript

formvalidate.js

```
function isBlank(x) {
    "use strict";
    return (x === null || x === "");
}
function checkRequired(form) {
    "use strict";
    var emailAlert = false, emailblankAlert = false, nameblankAlert = false;
    var passblankAlert = false, name = document.forms[form][0];
    var email = document.forms[form][1], pass = document.forms[form][2];
    var atpos = email.value.indexOf("@"), dotpos = email.value.lastIndexOf(".");
    var emailcheck = document.getElementById("emailcheck");
    var namecheck = document.getElementById("namecheck");
    var passcheck = document.getElementById("passcheck");
    if (isBlank(email.value)) {
        email.style.border = '1px Solid Red';
        emailblankAlert = true;
        emailcheck.innerHTML = '';
    } else {
        if (atpos < 1 || dotpos < atpos + 2 || dotpos + 2 >= email.value.length) {
            email.style.border = '1px Solid Red';
            emailAlert = true;
            emailcheck.innerHTML = '';
        } else {
            email.style.border = '1px Solid Gray';
            emailcheck.innerHTML = '';
        }
    }
    if (isBlank(name.value)) {
        name.style.border = '1px Solid Red';
        nameblankAlert = true;
        namecheck.innerHTML = '';
    } else {
        name.style.border = '1px Solid Gray';
        namecheck.innerHTML = '';
    }
    if (isBlank(pass.value)) {
        pass.style.border = '1px Solid Red';
        passblankAlert = true;
        passcheck.innerHTML = '';
    } else {
        pass.style.border = '1px Solid Gray';
        passcheck.innerHTML = '';
    }
    if (emailAlert) {
        alert("Please enter a valid email.");
        return false;
    } else {
        if (nameblankAlert || emailblankAlert || passblankAlert) {
            return false;
        }
    }
    return true;
}
```

loginvalidate.js

```
function isBlank(x) {
  "use strict";
  return (x === null || x === "");
}
function loginRequired(form) {
  "use strict";
  var emailAlert = false, emailblankAlert = false, passblankAlert = false;
  var email = document.forms[form][0], pass = document.forms[form][1];
  var atpos = email.value.indexOf("@"), dotpos = email.value.lastIndexOf(".");
  var emailcheck = document.getElementById("emailcheck"), passcheck;
  if (isBlank(email.value)) {
    email.style.border = '1px Solid Red';
    emailblankAlert = true;
    emailcheck.innerHTML = '';
  } else {
    if (atpos < 1 || dotpos < atpos + 2 || dotpos + 2 >= email.value.length) {
      email.style.border = '1px Solid Red';
      emailAlert = true;
      emailcheck.innerHTML = '';
    } else {
      email.style.border = '1px Solid Gray';
      emailcheck.innerHTML = '';
    }
  }
  passcheck = document.getElementById("passcheck");
  if (isBlank(pass.value)) {
    pass.style.border = '1px Solid Red';
    passblankAlert = true;
    passcheck.innerHTML = '';
  } else {
    pass.style.border = '1px Solid Gray';
    passcheck.innerHTML = '';
  }
  if (emailAlert) {
    alert("Please enter a valid email.");
    return false;
  } else {
    if (emailblankAlert || passblankAlert) {
      return false;
    }
  }
  return true;
}
```