California State University, San Bernardino

CSUSB ScholarWorks

Theses Digitization Project

John M. Pfau Library

2007

Document imaging application

Ruchi Sukhija

Follow this and additional works at: https://scholarworks.lib.csusb.edu/etd-project

Part of the Databases and Information Systems Commons

Recommended Citation

Sukhija, Ruchi, "Document imaging application" (2007). *Theses Digitization Project*. 3217.
https://scholarworks.lib.csusb.edu/etd-project/3217

This Project is brought to you for free and open access by the John M. Pfau Library at CSUSB ScholarWorks. It has been accepted for inclusion in Theses Digitization Project by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.

DOCUMENT IMAGING APPLICATION

A Project

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Ruchi Sukhija

December 2007

DOCUMENT IMAGING APPLICATION

A Project

Presented to the

Faculty of

California State University,

San Bernardino

by

Ruchi Sukhija

December 2007

Approved by:

[redacted]

Dr. David Turner, Chair, Computer Science

11/28/07

Date

[redacted]

Dr. Tong Lai Yu

[redacted]

Dr. Ernesto Gomez

ABSTRACT

Nowadays, storing documents digitally is highly preferable, especially in medical facilities, as they are the ones who deal with the largest paper trails. Millions of documents are managed by a typical hospital, including records of patient visits, lab tests, x-ray reports, insurance information, proofs of payments (such as check copies), and cash receipts. As required by state law for audit purposes, these documents are archived for at least 7 to 10 years for every patient of the medical facility. Storing these document is a significant expense in terms of paper consumption and storage space. Also, the manual retrieval of documents is costly.

Document Imaging Application is proposed and developed to resolve this issue. By scanning the documents into an electronic repository, medical staff will be able to more easily store and locate these records. To make the application user friendly and facilitate staff access to patient medical records, the application is web-based.

However, this brings a great responsibility to the developer to insure that unauthorized access does not occur. The document imaging application uses the Oracle Application Server to implement a multitiered model.

ACKNOWLEDGMENTS

I extend my heartfelt acknowledgement to all the staff members of Computer Science Department of CSUSB. My profound acknowledgement goes to Dr. David Turner, Dr. Tong Yu, Dr.Earnest Gomez and Dr. Jospine Mendoza for their encouragement and guidance during this endeavor. I would like to specially thank my advisor, Dr. David Turner for giving me the inspiration and direction to write this paper.

My fond acknowledgement to my husband Paramjit S. Sukhija for his tireless encouragement and, to my children Jatin and Divya for their love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

There is a current need at the hospitals these days to store various documents in the digital format which helps them to access the patient's information from anywhere in the facility. This digital image repository is developed to support the collection of all patient related documents and replace the current manual process of archiving and investigation, which generates a large amount of paper trail. The example of health department and finance department is taken into consideration to demonstrate this project approach.

1.1 Current Business Process Definition

1.1.1 Patient Admission

Mostly patients visit the hospital for some kind of procedure and during the admission, hospital personnel enters the information of the patient in the computer and assign a unique medical record number(MRN) to that person.

1.1.2 Finance Department

The payment-processing group receives payments and payment documents from patients, insurance companies, Medicare, and other third parties. These documents are in various forms such as checks, cash receipts, credit card receipts, Explanation of Benefit, and remittance advice.

Once a payment is received, the payment information is entered manually into the accounting system by crediting the patient account. Once the account has been credited, the payment documents are then grouped into batches by deposit day and bundled together for archiving purposes. To comply with state regulations, paper documents are kept on file for 7 to 10 years.

Payments that cannot be readily associated with a patient account are posted to a holding account. Once a payment can be traced back to a patient account the amount is transferred from the holding account to the correct patient account.

The following diagram describes this process.

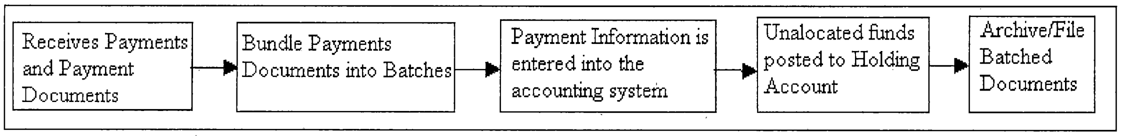

Receives Payments and Payment Documents
Bundle Payments Documents into Batches
Payment Information is entered into the accounting system
Unalocated funds posted to Holding Account
Archive/File Batched Documents

Figure 1. Posting Payment Information

<u>1.1.2.1 Payment Research</u>. Finance Service account reps must frequently retrieve specific payment documents for a specific service date. Payment disputes with insurance company or a patient inquiry regarding a payment

trigger this search. This is currently a very tedious process. First, a finance service account rep must look for the patient account number in account system and determine the service date. Once the account number and the service date are known, the finance service account rep must go through a large volume of paper documents to find the paper batch for the specific service date. Then, he/she must manually go through the batch to locate the payment document for a specific patient account. To support this, there are 5 associates in the department whose sole responsibility is to search the filing room for stored batches and locate batches for specific dates and hand deliver them to finance service account rep.

The following diagram describes this process.

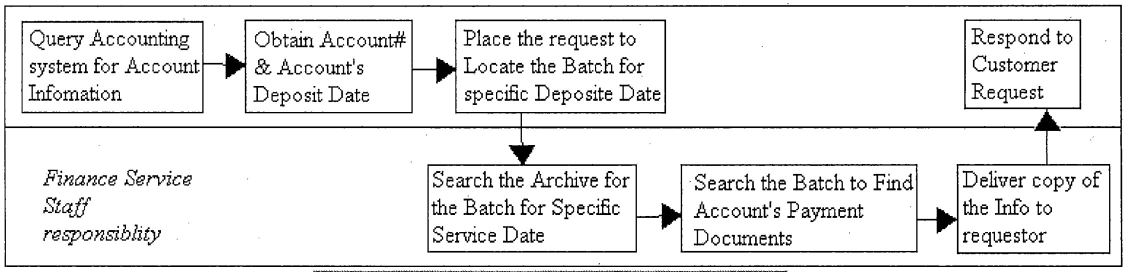

Query Accounting system for Account Infomation
Obtain Account# & Account's Deposit Date
Place the request to Locate the Batch for specific Deposite Date
Respond to Customer Request
Finance Service Staff responsiblity
Search the Archive for the Batch for Specific Service Date
Search the Batch to Find Account's Payment Documents
Deliver copy of the Info to requestor

Figure 2. Researching Payment Information

<u>1.1.2.2 Payment Document Types</u>. For any patient account, there are a variety of payment documents that are being archived. Some examples of these documents are EOBs,

remittance advices, checks, account statements, and payment notices.

The amount of documents archived on a daily basis is very and are of different types. Some of the examples are as following:

- Explanation of Benefits (EOB)
- Remittance Advice
- Credit Card and Cash Receipts
- Personal Checks
- Provider Checks
- Bank Deposit Slips
- Check Summary Reports (provided by providers)
- Patients' Account Statements
- Patients' Payment Notices
- Payment Summary Reports (provided by providers)
- Other documents (hand written letters by patients, providers' letters, collection notices, etc.)

1.1.3 Health Department

As payment-group deals with the patient related payment documents, Health information department(HID) deals with the patient related medical records. Following are some of the examples of medical documents HID deals with:

- PNR - Pre Natal Records: These are the pregnancy related documents which may contain ultrasound images to lab tests run on the patients.
- VMO - Verbal Medical Order: There are times when physicians prescribe medicines verbally for the patients. The assigned medical assistant fill the verbal medical order form and have it signed from the physician later for the record keeping purpose.
- ROI - Release Of Information: This document is to authorize the hospital to release the patient's health related information either to state or to other departments. This must be signed by the patient.
- HNP - History and Physical Information: These documents hold the historic health and physical information of the patient.

1.2 Document Imaging Overview

The initiative behind this project is to help the medical industry reduce the paper trail of any kind. The approach is to scan the documents from anywhere in the medical center and have the facility to view them online from anywhere within the hospital. By having this system, the users will have the following advantages.

- Ability to view the records anywhere in the hospital facility in a timelier manner.
- Ability to provide better patient care.
- Paper reduction.
- Permanent archival of records, reduce number of lost records.
- HIPAA compliance - Health Insurance Portability and Accountability Act address the security and privacy of health data. To comply with this state law, there is a privilege in this application to capture who is accessing the patient information for audit reasons.

The following process flow (Figure 3) demonstrates the process. The HP scanners can reside anywhere in the hospital, the few locations are mentioned in the figure. There is one server named shared_server dedicated to receive all the scanned files from the scanners. The ftp process picks up the files from the scanned_server and moves to the database server (database_server). As there are many tools available in the market for the file transfer, but to accomplish our approach, we will be using Appworx. Appworx is a tool to automate scripts periodically. Once the files are available on the server where 10g database is residing, database jobs can be scheduled to load them into the database. The users will be communicating with Oracle Application server from their browsers to view the files.

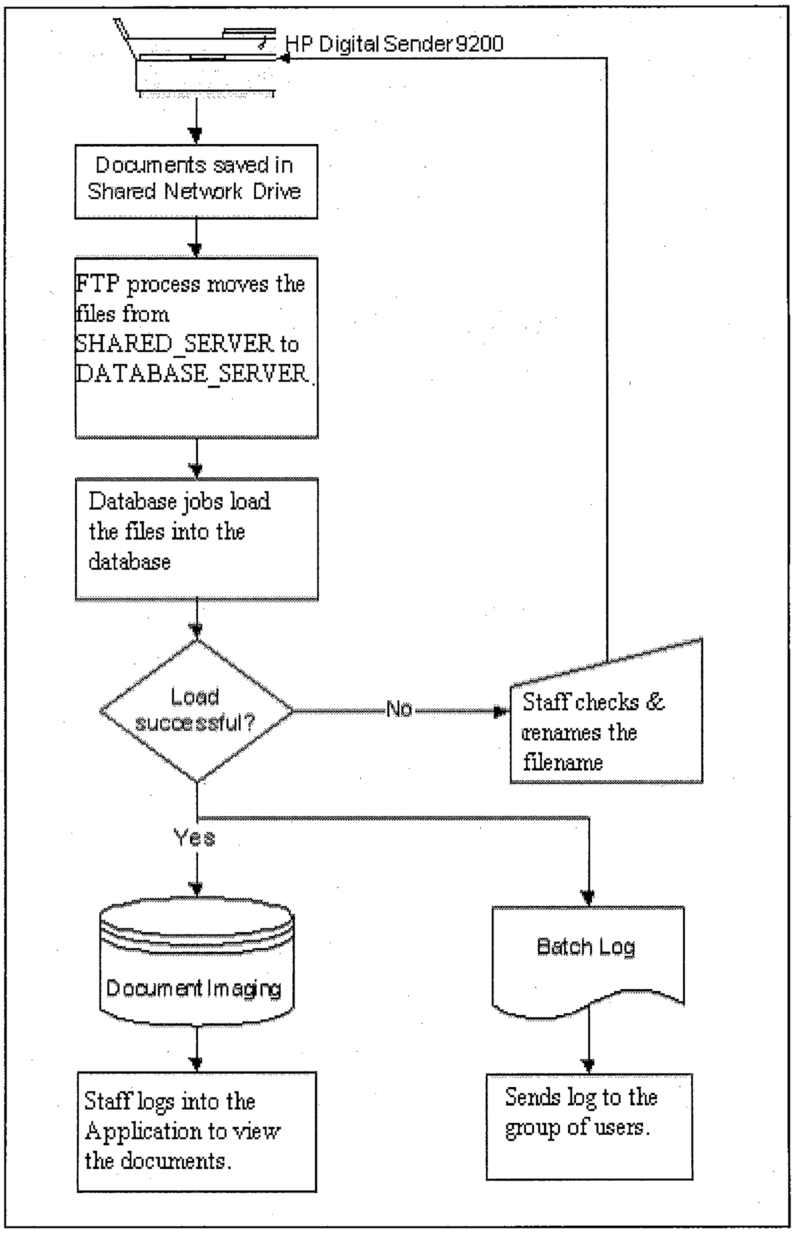

HP Digital Sender 9200
Documents saved in Shared Network Drive
FTP process moves the files from SHARED_SERVER to DATABASE_SERVER.
Database jobs load the files into the database
Load successful?
No
Staff checks & renames the filename
Yes
Document Imaging
Batch Log
Staff logs into the Application to view the documents.
Sends log to the group of users.

Figure 3. Application Overview

1.3 Application Server Overview

Application Server is an integral part of this project as users will be retrieving the loaded scanned documents from the database via application server. It will be handling all the browser based requests and responses. Therefore, discussing application server's security components and modules which communicate with underlying database will give an overall insight about the data flow.

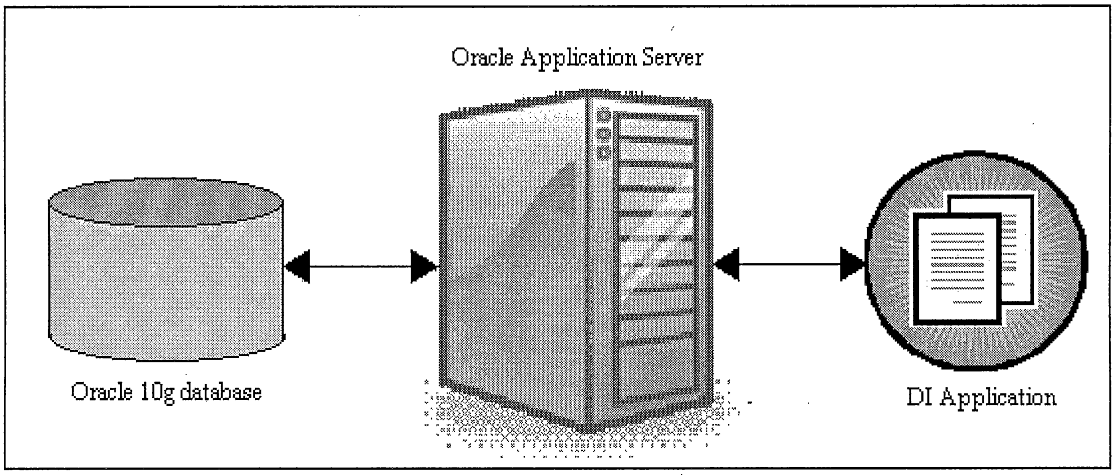


Figure 4. Data Flow

In addition to that, learning about Application Server is necessary for the business goals, especially in this era when the technology everywhere is based on multitier model. Application Server plays a major role in understanding the integration of key components and features and it addresses the following key solution areas.

- Deploying and managing J2EE applications.
- Deploying and managing portals and wireless-enabled applications.
- Accelerating performance with caching.
- Managing and securing the Web infrastructure.

I'll be discussing some of components of Oracle Application Server in the next chapter.

CHAPTER TWO

ORACLE APPLICATION SERVER

2.1 Overview

Oracle Application server is a standards-based application server that offers a fully integrated platform to develop, deploy, and administers Internet-based applications. Following are some of the solutions provided by Oracle Application server, especially for developers.

2.1.1 Hypertext Transfer Protocol Server, Java 2 Platform Enterprise Edition, and Web Services

Oracle HTTP server acts like a HTTP interface for all Oracle Application Server components. As Oracle Application Server is built on J2EE framework. It enables us to design, develop and deploy websites, and applications by using familiar languages.

2.1.2 Portals

Oracle Application Server provides the facility to create, and maintain enterprise portals. It has wizards available to maintain and publish the services.

2.1.3 Wireless

Oracle AS (Oracle Application Server) also provides development and deployment of applications in the wireless environment such as e-mails etc.

2.1.4 Caching

Oracle AS has a unique ability to cache both static and dynamic generated Web content. This feature really improves the performance and scalability of heavily loaded Web sites. It has number of features to ensure consistent response. These features include

- Page-fragment Caching
- Edge Side Includes(ESI)
- Edge Side Includes for Java support(JESI)
- Web-server load balancing
- Web Cache clustering

2.1.5 Business Intelligence

This is a great feature in Oracle Application Server as by using this, visitors can perform dynamic, ad hoc query reporting and analysis using a Web browser; and publishes high quality, dynamically generated reports on a secure platform.

2.1.6 Integration

By using Oracle AS, we can integrate any enterprise application, Web services and provide query access to many non-Oracle data sources.

2.1.7 Management and Security

Oracle AS monitors each individual Oracle Application Server instances to optimize them for performance and scalability. It uses encrypted secure sockets layer(SSL) connections, user and client based certificate-based authentication, and single sign-on across all the applications. In addition to that for security, it has LDAP directory that provides a single repository and administration environment for user accounts.

2.2 Architecture

Oracle AS architecture is based on a multitiered model as its components reside at different tiers and layers, with each tier made up of one or more servers. In general, number of tier and number of servers in each tier depend on the Oracle AS's implementation. The functional architecture (Figure 5) of Oracle Application Server is as follows:

- Web Tier: The listener listens on a specific port for the incoming requests. The Web Cache stores the pages that are accessed frequently and also, load balances to ensure the optimal results.

- Application Server Tier: This controls the business logic and portals within defines the Web page components and also, single sign-on controls security for the application server layer.
- Database Tier: This stores the metadata and acts like a repository for storage and retrieval of application data.

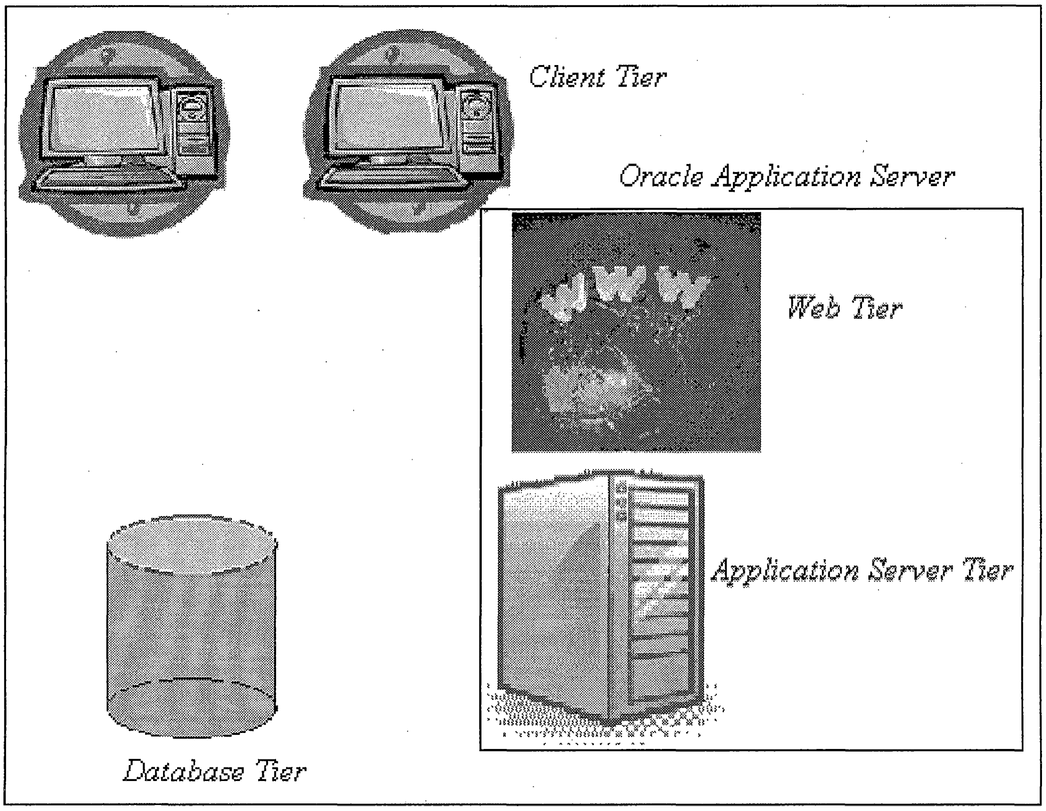


Figure 5. Oracle Application Server 10g Architecture

2.3 Oracle Hypertext Transfer Protocol Server

Oracle HTTP Server is based on the Apache Web Server and is the Web Server component of Oracle Application Server. Oracle HTTP server dispatches requests to invoke program logic written in Java, PL/SQL, PERL, PHP, or as CGI executables through a standard module architecture (Figure 6). Infact, Oracle HTTP server extends the functionality of Apache to provide SSL and HTTPS support also.

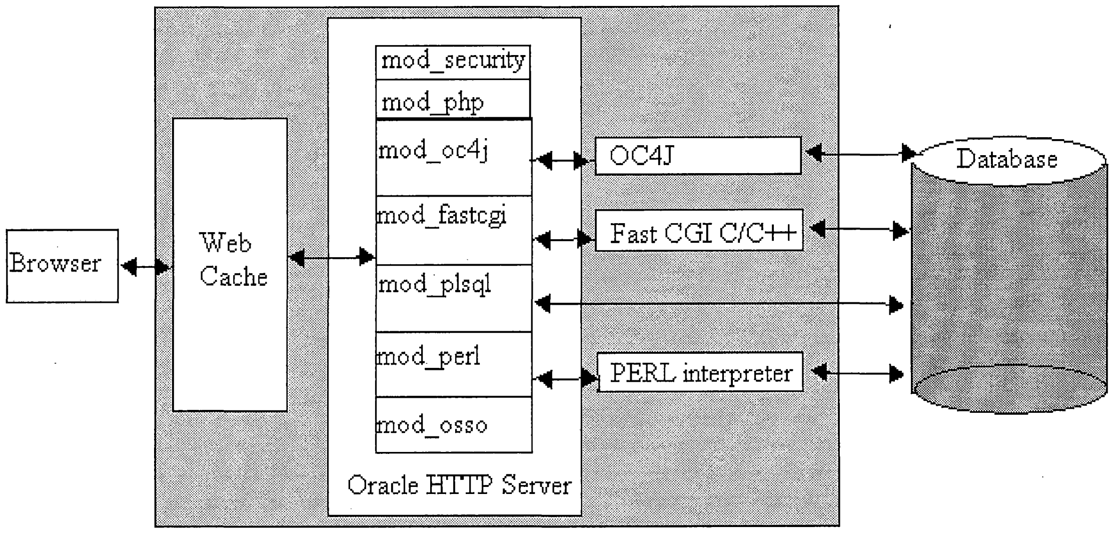


Figure 6. Oracle Hypertext Transfer Protocol Server

The functionality of its modules is as following.

- mod_security: Is an open source intrusion detection and prevention engine for Web applications.

- mod_php: This module enables PHP scripts to be executed in Oracle HTTP server.
- mod_oc4j: This module routes the communication between Oracle HTTP server and OC4J.
- mod_fastcgi: This supports CGI processes.
- mod_plsql: This module routes the requests for stored procedure to the database server. I'll be discussing this module in the later section.
- mod_perl: This module routes PERL requests to PERL interpreter.
- mod_osso: This module routes the requests to Oracle AS Single Sign-on.

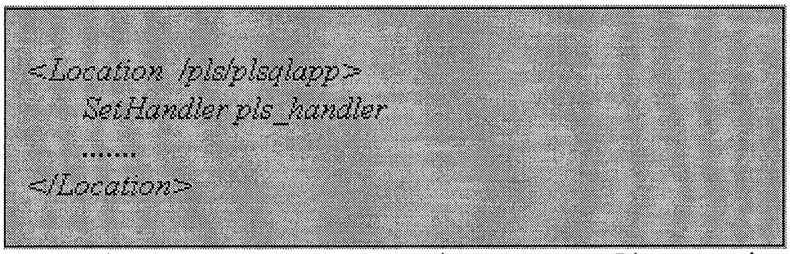
2.4 Module: mod_plsql

The mod_plsql module of Oracle HTTP server routes PL/SQL requests to the Oracle PL/SQL service which, in turn, delegates the servicing of requests to PL/SQL programs. Therefore in the nutshell, mod_plsql enables Oracle Application Server to connect to an Oracle database server and execute stored procedures. Each mod_plsql request is associated with a database access descriptor (DAD), which specifies the following information.

- The database alias
- A connect string if the database is remote

- A procedure for uploading and downloading documents.

Or we can say that a DAD is a set of values that specify how mod_plsql connects to a database server. The dads.conf file (Figure 7) contains the configuration parameters for the PL/SQL database access descriptor (DAD).



```
<Location /pls/plsqlapp>
    SetHandler pls_handler
    .......
</Location>
```

Figure 7. Database Access Descriptor Configuration File

The PL/SQL procedure which is invoked can perform some operations on the database and return the results to the users in the HTML format containing the data from the database. To invoke a PL/SQL stored procedure in a Web browser, the URL typically must be in the following format:

protocol://host[:port]/path/[package.proc_name[?query_string]]

- Protocol can be either http or https.
- Host is the domain-qualified name of the machine where the Web server is running.

- Port is the port at which the application server is listening.
- Path is the virtual path to handle PL/SQL requests that are mounted in a <Location> container for a specific DAD. This also includes the connection information.
- Package is the database PL/SQL package that contains the stored procedures.
- Proc_name is the name of the stand-alone procedure.
- ?query_string specifies parameters(if any) for the stored procedure.

The mod_plsql comes with the PL/SQL toolkit,a set of packages which can be used in the procedure to get information about the request, construct HTML tags, and return the header information to the client.

This Document Imaging Application is developed using the PL/SQL toolkit packages.

2.5 The Path of Hypertext Transfer Protocol Requests

Oracle HTTP server is an underlying deployment platform, and it provides a Web Listener for Oracle AS

Containers. The communication flow of HTTP requests is as following (Figure 8).

1. The browser sends a URL to the listener. The listener examines the URL and determines that the request is for module (in this case mod_plsql).
2. If authentication is required, the listener contacts an authorization module such as mod_ossl or mod_auth with the URL and browser credentials.
3. The authorization module validates the request and returns the result to the required module.
4. mod_plsql uses the database access descriptor(DAD) configuration values to determine how to connect to the database.
5. mod_plsql connects to the database, prepares the call parameters and invokes the PL/SQL procedure named in the database.
6. The PL/SQL procedure generates the HTML page that can include dynamic data accessed from the tables in the database, as well as static data.
7. The output from the procedure is returned by way of the response buffer to mod_plsql.

8. Oracle HTTP server sends the response back to the client.

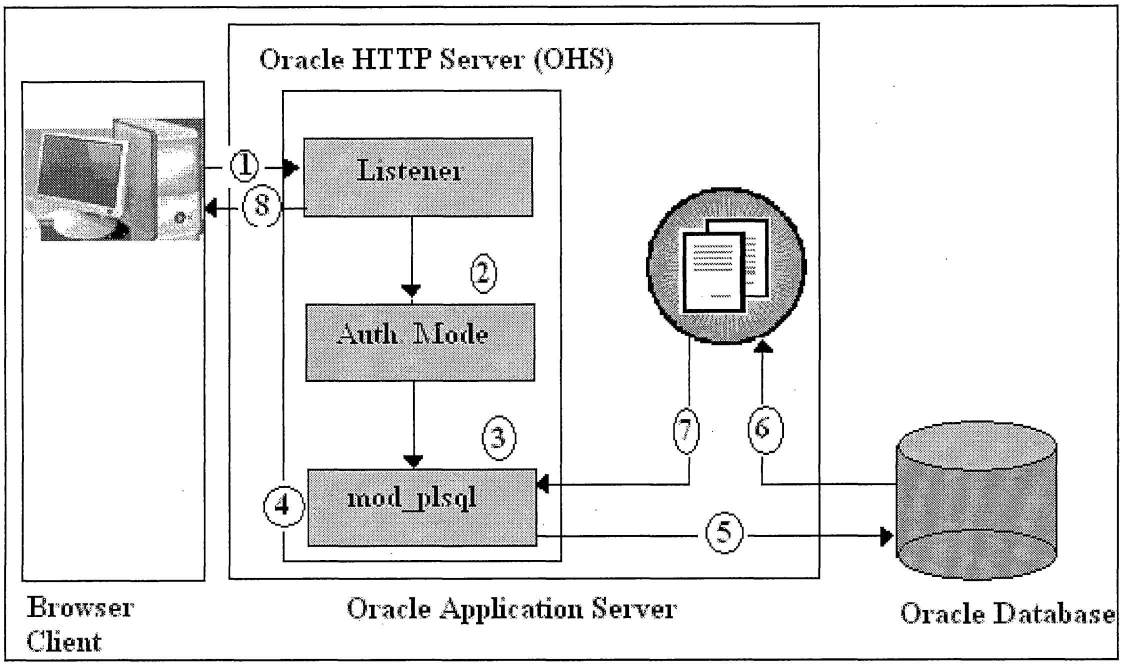


Figure 8. The Path of Hypertext Transfer Protocol Requests

2.6 Securing the Web Infrastructure

To protect against malicious intrusions, Oracle Application Server provides the following solutions.

- Secure socket layer (SSL) encryption can be used to protect the Web site.

  1. Oracle Internet Directory - It is an LDAP server that can be used to store the credentials required for the enterprise.

- Oracle Single Sign-On: This validates user credentials against Oracle Internet Directory.

After user sign on successfully, their credentials are automatically retrieved from Oracle Internet Directory when they launch any Oracle partner application.

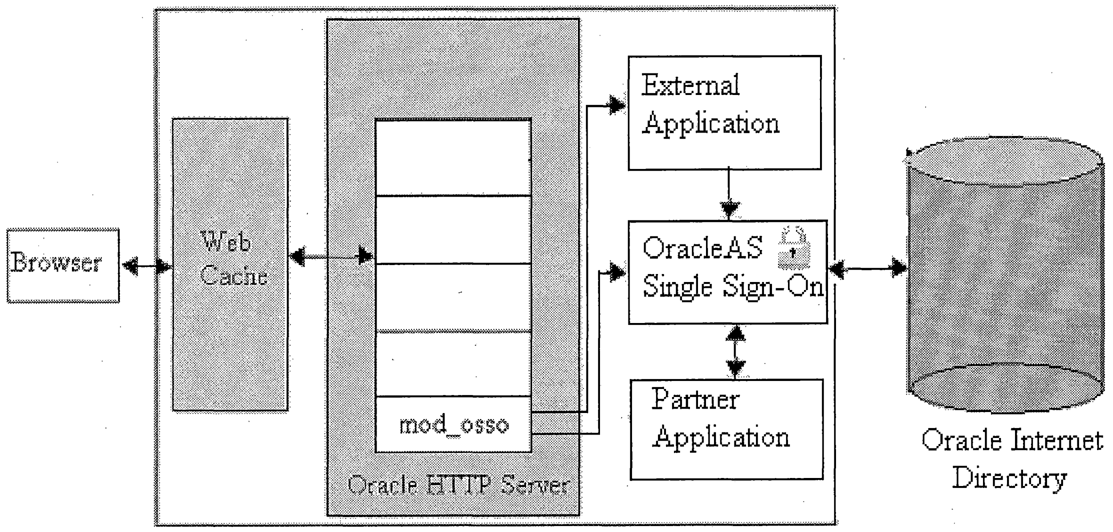


Figure 9. Oracle Internet Directory and Security

CHAPTER THREE

REQUIREMENT ANALYSIS

3.1 Use Case Analysis

The user community will consist of many people working on various locations and will be divided into two types of roles:

- Users scanning the documents (scanning role).
- Users viewing the scanned images (viewer role).

The number of users with scanning role, their location, and the amount of documents to be scanned determines the number of scanners required. Users with viewer role must have PCs capable of using a web-browser and monitors with a decent resolution.

3.1.1 Users Scanning the Documents

The users assigned with "Document Uploader" role will have the privilege to upload the documents, search, and views and delete the documents, view the daily batch log and change their passwords (Figure 10).

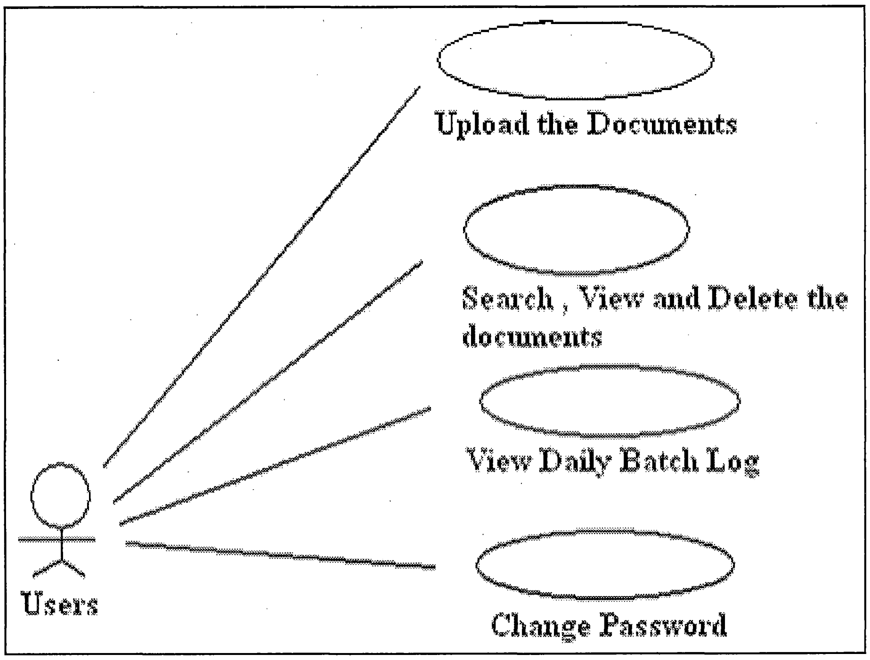


Figure 10. Case Diagram for Document Uploader Role

There are two ways the files can be loaded into the system for viewing purpose. One will be through scanners which is a bulk load. Assigned users will scan the files and system will do the rest for them. The other will be a drag and drop approach via application to load one file at a time in the system manually.

Following are the steps taken to load the documents into the system via scanners:

Table 1. Steps to Load the Document via Scanner

Step	User Action	Business Rules	System Response
1	Put the document in the feeder of the scanner.	None	None
2	Type the name of the file on the keypad of the scanner according to the business rule.	Files must be named in the following format <doctyp>_<MRN#>{_<keyvalue>} where <doctype> - the type of the document e.g VMO for verbal medical orders. <MRN#> - must be the valid MRN of the document related patient. {_<keyvalue>}- this is an optional field. But, if used, it must be a number field.	None
3	Hit "Start" and wait till scanning is complete.	None	Stores documents in the shared network drive (shared_server).
4	None	None	FTP process automatically moves the file from shared_server to database_server.
5	None	None	Database jobs loads these files into the database and sends the e-mail notification to users.
6	After getting e-mail notification, log in the application (see user manual for more detail) and verify the file load	None	Respond according to user query.

User manual describes the step by step process to load the document(s) via application.

3.1.2 User Viewing the Scanned Images

The users assigned with the "Document Viewing" role will be able to search and view the documents, view the daily batch log and change the password in the application (Figure 11).

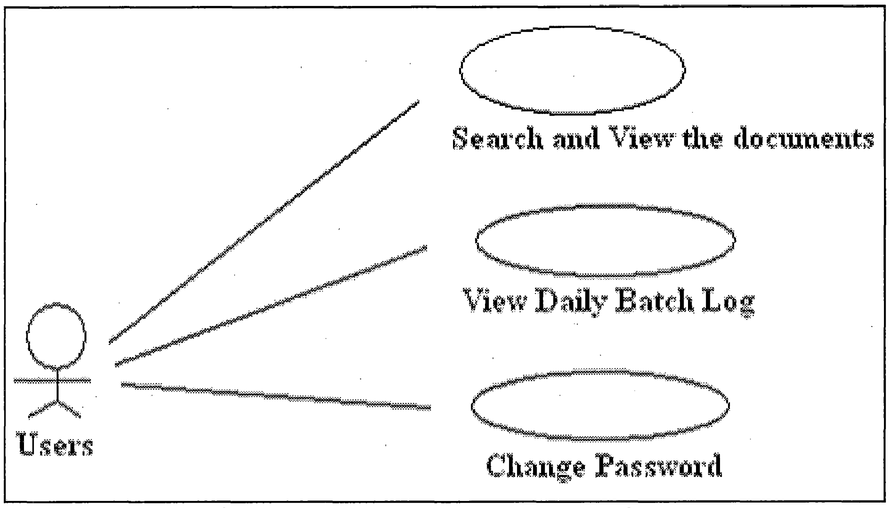


Figure 11. Case Diagram for Document Viewer Role

To view the documents via application, one must have the privilege to access it using the authentic username and password.

For complete application navigation with the screen shots please see the user manual.

3.2 System Architecture

Users will scan the documents via scanners and the documents will directly go to the shared server (see Figure 12). These scanners can reside at different locations within the hospital. Thereafter, FTP scripts will be scheduled to run every 5 minutes which will move the scanned documents from the shared server to the database server. Once the files will be available on the database server, the database jobs will be scheduled to load these files into the Oracle 10g database. At the same time, the database jobs will send the log to the specified users to state which files loaded successfully and which error out.

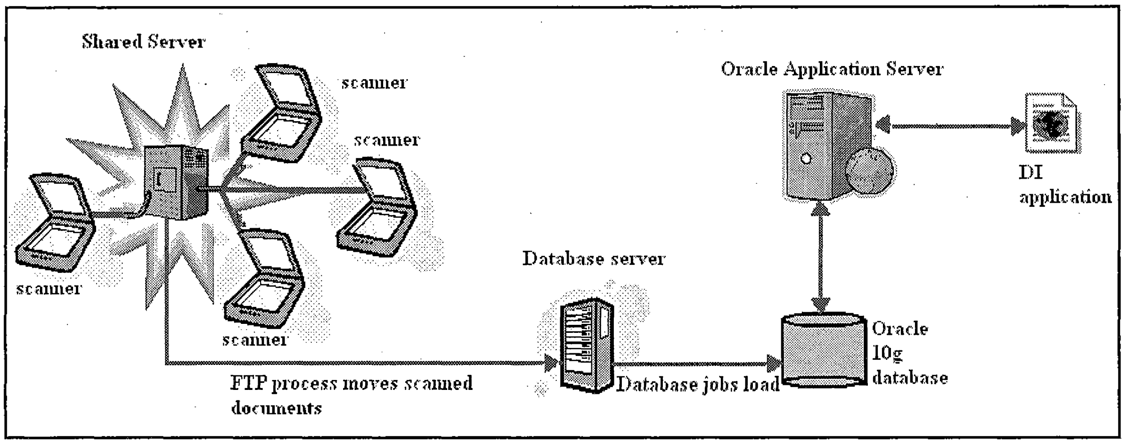


Figure 12. System Architecture

At any time after that, users will be able to search and view the scanned files in the database.

Hidden from the users, the communication with database will be handled by Oracle Application Server. Besides searching, viewing the scanned files, users will be able to change their passwords, view the daily batch log, and depending on their roles, they will be able to upload the document one by one via DI Application.

CHAPTER FOUR
TECHNICAL REQUIREMENT SPECIFICATION

4.1 Assumptions and Dependencies

A generic solution is proposed to accommodate all types of documents that may be scanned. Basic assumptions that will govern the application approach are:

- All patient related imaged documents will minimally be associated with a patient account or patient MRN.
- Each document type will have defined keywords that can be associated with the document types.

4.2 File Transfer Protocol/Batch Process

This process is to move the files from the shared server to the database server. As the database server is a Linux box, therefore shell scripts (see Appendix C) are used to perform the file transfer. I have used the tool "Appworx" to schedule the shell scripts for every 5 minutes.

Basically, scripts visit the shared server every 5 minutes and move the files to designated directories in the database server based on their type.

For example:

vmo_xxxxxxxxx_yyyyyy.pdf is of a type vmo(verbal medical order) where xxxxxxxxx is the MRN# and yyyyyy is the physician id. So, the script will check the type of the document and move it /database_server/vmo folder for further processing.

4.3 Database Package

As the chosen database for this project is Oracle 10g which comes with lot of standard packages for the use. One of them is DBMS_JOB, this standard package is provided to schedule the database batch jobs. This package comes with plenty of procedures and functions to run the processes smoothly. For example, DBMS_JOB.BACKGROUND_PROCESS: This tells whether the execution is a background process or the foreground process.

DBMS_JOB.BROKEN: This procedure is used to halt or re execute the execution of the process depending on the parameter e.g. DBMS_JOB.BROKEN(21,TRUE) where TRUE implies to pause the process with ID 21.

For scheduling the database job, I have used the SUBMIT function of the DBMS_JOB package (see Appendix D). The syntax of this is as following:

```
dbms_job.submit(
JOB        OUT BINARY_INTEGER,
WHAT       IN  VARCHAR2,
NEXT_DATE IN  DATE DEFAULT SYSDATE,
INTERVAL   IN  VARCHAR2 DEFAULT 'NULL',
NO_PARSE   IN  BOOLEAN DEFAULT FALSE);
```

JOB: is an out parameter which will be assigned to the process which we are scheduling. Therefore, to alter this scheduled process later, one has to always use this ID.

WHAT: is the user PL/SQL code which we want to execute.

NEXT_DATE: is the date when the PL/SQL code will execute again.

INTERVAL: This field is used to calculate the time of the execution of PL/SQL code.

NO_PARSE: is the flag to indicate Oracle whether to parse the procedure associated with the job or not. The default value is FALSE.

The PL/SQL code can be any package / procedure or function based on your business requirements. Oracle also provides HTP packages which actually generates the HTML tags for you.

4.4 Database Model

After gathering all the requirements from the end users, I have used Oracle designer tool to model the ER diagram and then the final database model. The advantage of this tool is that after you finish designing your database model, this tool has an option to generate the database level scripts. Therefore, you don't have to manually type those scripts and also, if at the later point you want to modify the model, you can just generate the scripts to update the model at any time.

As database design is the heart and core of the project; the success of the project relay on this. I am discussing the tables in this section one by one which will help to understand the business flow as well the need and function of them.

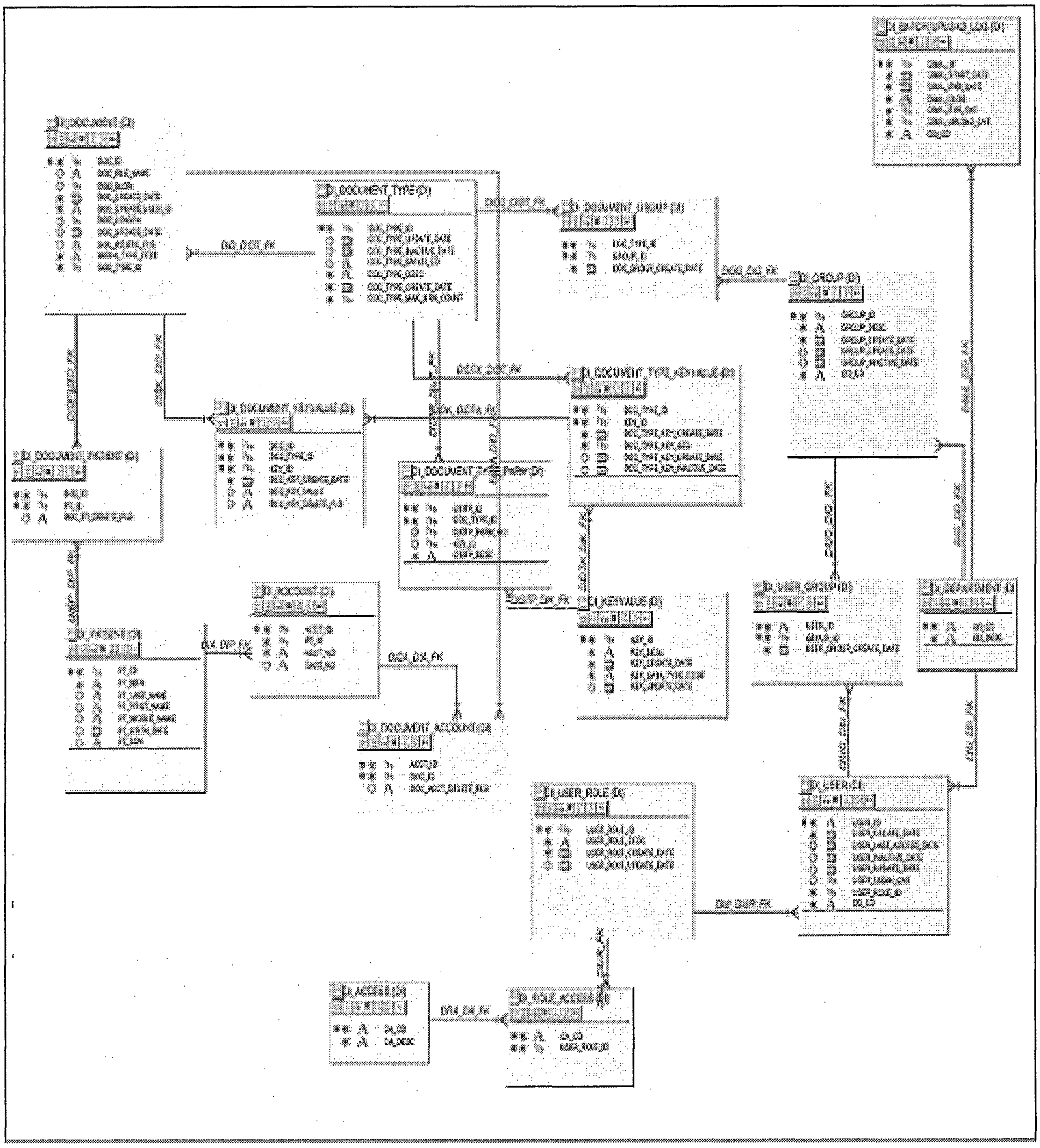


Figure 13. Server/Database Model

DI_Document: This table contains all the document related information.

Table 2. Document Imaging Document

Attributes	Comments	Type
Doc_Id	The unique key to identify the each document	Numeric
Doc_File_Name	The name of the document given by the user while scanning.	Character
Doc_Blog	The field to store the scanned content.	Blog
Doc_Create_Date	The date the document was scanned into the database	Date
Doc_Create_User_Id	The foreign key referencing to User_id of DI_User table, storing the user information who scanned the document.	Numeric
Doc_Length	The length or size of the scanned document.	Numeric
Doc_Update_Date	If any, the date of the last updation on the document record.	Date
Doc_Delete_Flg	This field is used as a flag to mark the record as a deleted rather than physically deleting the record from the database.	Character
Media_Type_Text	The type of the media text.	Character
Doc_Type_Id	The foreign key referencing to Doc_type_Id of DI_Document_Type table, telling the type of the document.	Numeric

DI_Patient: This table stores the patient information.

Table 3. Document Imaging Patient

Attributes	Comments	Type
PT_ID	Unique identifier to represent each patient record in the database.	Numeric
PT_MRN	Unique number used to represent the patient throughout the hospital.	Numeric
PT_Last_Name	Patient's last name	Character
PT_First_Name	Patient's first name	Character
PT_Middle_Name	Patient's middle name	Character
PT_Birth_Date	Patient's birthdate	Date
PT_ssn	Patient's Social Security number	Character

DI_Document_Patient: As DI_Document table stores many documents and DI_Patient table stores many patient's information. This table is used to record which document is associated with which patient, as to model many to many relationship.

Table 4. Document Imaging Document Patient

Attributes	Comments	Type
Doc_id	Unique key to identify each document	Numeric
PT_id	Unique key to identify each patient	Numeric
Doc_pt_delete_flg	Field is used to delete the association between document and the patient	Character

Relationship: This table stores Doc_id as foreign key from the di_document table, similarly, pat_id as foreign key from the di_patient table.

DI_Account: This table stores the account information of the patient. Account is the term used for the billing of the particular services and patient may visit the hospital for many health related services.

Table 5. Document Imaging Account

Attributes	Comments	Type
Acct_id	Unique key for the each account number	Numeric
Pt_id	Unique key referenced to the pt_id in DI_Patient table	Numeric
Acct_no	The unique number used throughout the hospital associated with the patient	Character
Case_no	Health related service code	Character

DI_Document_Account: This table is used to record the association between each document and the account number. This is very useful for finance department where they scan all the checks, receipts etc which are the proof of payment for the patient.

Table 6. Document Imaging Document Account

Attributes	Comments	Type
Acct_id	Unique key referenced to acct_id in DI_Account table	Numeric
Doc_id	Unique key as a foreign key referenced back to doc_id in DI_Document table	Numeric
Doc_Acct_Delete_Flg	The field to remove the association-record between document and the account	Character

DI_User: All the user and login related information is stored in this table.

Table 7. Document Imaging User

Attributes	Comments	Type
User_Id	The unique key to identify each user	Numeric
User_Create_Date	Date the login id was created in the database	Date
User_last_access_date	The last date user accessed the system	Date
User_Inactive_Date	If user is inactive then since when the user is being inactive	Date
User_login_cnt	How many times user has logged in the database	Numeric
User_role_id	What is the role assigned to the user in the database	Numeric
DD_Cd	The code of the department, user belongs to.	Character

DI_User_Role: All the application-roles related information is stored in this table.

Table 8. Document Imaging User Role

Attributes	Comments	Type
User_Role_Id	Unique key assigned to each role in the database	Numeric
User_Role_Desc	This field stores the description of the role in detail	Character
User_Role_Create_Date	The role created in the database	Date
User_Role_Update_Date	If the role is modified then it stores the date	Date

DI_Access: This table stores all the access information. This access is independent of the database access like select, delete, insert etc. It stores the application level access which application administrator can see and understand like delete file - access, create user access etc.

Table 9. Document Imaging Access

Attributes	Comments	Type
DA_Cd	Unique code assigned to each access	Character
DA_Desc	Description of the code	Character

DI_Role_Access: This table stores the information related to each role and the access granted to the role. This acts like a connecting table between DI_User_Role and DI_Access.

Table 10. Document Imaging Role Access

Attributes	Comments	Type
DA_Cd	Unique key referenced to DA_Cd/DI_Access	Character
User_Role_Id	Foreign key referenced to User_Role_id/DI_User_Role	Numeric

DI_Department: All the departments are stored in this table. There are only two attributes in this table. One is

DD_CD which defines the department code and another one is DD_Desc which defines the detailed description of the department. Considering the departments at the hospital for example Imaging department, this department may have many different type of documents to scan e.g MRI reports, x-ray reports, ultrasound images etc. Therefore , there may be many groups of users in one department depending upon the documents they are scanning. Apparently, there will be many users in one group. Based on these scenarios, the server model has the following group tables as well as the association between the group table and user table.

DI_Group: This table stores all the information related to each group in the department

Table 11. Document Imaging Group

Attributes	Comments	Type
Group_Id	Unique key used to identify each group	Numeric
Group_Desc	Detailed information about the group	Character
Group_Create_Date	When the group was created in the database	Date
Group_Update_Date	If any, when was the group information modified	Date
Group_Inactive_Date	If any, whether the group is active or inactive. If the group is active this field will be null.	Date
DD_Cd	Foreign key reference to DD_CD of DI_Department	Character

DI_Document_Group: These tables demonstrate which group scans what kind of documents.

Table 12. Document Imaging Document Group

Attributes	Comments	Type
Doc_Type_Id	Foreign key reference to Doc_Type_Id of DI_Document_Type	Numeric
Group_Id	Foreign key reference to Group_id of DI_Group.	Numeric
Doc_Group_Create_Date	When the group was authorize to scan this type of document	Date

DI_Document_Type: As the name suggests, this table stores the information about the document type. Doc_Type_Id is the primary key, which uniquely identifies each type of the document. Doc_Type_Desc stores the detailed information about the type of the document. Doc_Type_Update_Date is the date when the document type was modified. Doc_Type_Batch_Cd is the code under which the batch process will pick up these types of documents.Doc_Type_Max_Mrn_Count refers to the maximum number of MRN#s this type of document may have.

DI_Keyvalue: This table stores the information about all the keyvalue associated in the filename. For instance,if vmo_xxxxxxxxx_yyyyyy.pdf is the filename then yyyyyy is the keyvalue of the document. This either can be physician Id or account information depending upon the document. Therefore, information related to the keyvalue is stored in this table.

Table 13. Document Imaging Keyvalue

Attributes	Comments	Type
Key_Id	A key which uniquely identifies each keyvalue	Numeric
Key_Desc	The detailed description of the keyvalue	Character
Key_Create_Date	The date the keyvalue record was created in the database	Date
Key_Data_Type_code	The code which defines the type of data in the file.	Character
Key_Update_Date	If any, the date when the record was modified	Date

DI_Document_Type_Keyvalue: This table associates the type of the document with the keyvalues.

Table 14. Document Imaging Document Type Keyvalue

Attributes	Comments	Type
Doc_Type_Id	Foreign key references to Doc_Type_Id of the DI_Document_Type table	Numeric
Key_Id	Foreign key references to Key_id of the DI_Keyvalue	Numeric
Doc_Type_Key_Create_Date	The date when the type - keyvalue association was created	Date
Doc_Type_Key_Update_Date	If any, the date when the type keyvalue was modified	Date
Doc_Type_key_Inactive_Date	If this field is empty, that specifies the association between type and keyvalue is active	Date
Doc_Type_Key_SPO	Flag used to mark the type - keyvalue association as deleted instead of physically deleting the record.	Numeric

DI_Document_Type_Parm: This table holds the critical information about the document. It contains the parameter information of the filename which eventually tells about the document. For instance, considering the filename vmo_xxxxxxxxx_yyyyyy.pdf, the document with this name has three parameters; those are vmo, xxxxxxxxx, yyyyyy out of which vmo is the document type and rest of the two parameters can be defined as needed. For example, for vmo type document, the parameter two is defined as MRN # and the third parameter is the physician id. So, at the business level the whole filename tells us who is the physician with physician id yyyyyy who approved the verbal medical order (vmo) for this patient (xxxxxxxxx). The intention to have this table is to generalize the use of it. If we would like to extend the use of this database for scanning some other document example: HR records - in that scenario, we can have the Emp_zzzzzzzz_vvvvvv.pdf document where Emp will be the type of the document and zzzzzzzzz will be the employee Id as a second parameter and vvvvvv will be the keyvalue as a third parameter and which can be anything like benefit id or 401k document id etc.

Table 15. Document Imaging Document Type Parm

Attributes	Comments	Type
DIDTP_Id	A unique key to identify each document type parameter	Numeric
Doc_Type_Id	A foreign key referencing to doc_type_id of di_document_type	Numeric
DIDTP_Param_No	This field contains the information about the parameter number	Numeric
Key_id	A foreign key referencing to Key_id of Di_Keyvalue	Numeric
DIDTP_Desc	This field contains the description of the parameters	Character

DI_Document_Keyvalue: As there can be many documents with the same type and keyvalue, similarly, there may be many keyvalues associated with one document. Therefore, this is the table which associates one document with document type and the keyvalue.

Table 16. Document Imaging Document Keyvalue

Attributes	Comments	Type
Doc_Id	A foreign key referencing to Doc_Id of DI_Document	Numeric
Doc_Type_Id Key_Id	A foreign key referencing to the primary keys Doc_Type_Id & Key_id of DI_Document_Type_keyvalue	Numeric
Doc_Key_Create_Date	The date when the association was created	Date
Doc_Key_Value	It stores the value of the key from the name of the document e.g. yyyyyy	Character
Doc_Key_Delete_Flg	This field is to mark the record as a delete instead of physically deleting the record	Character

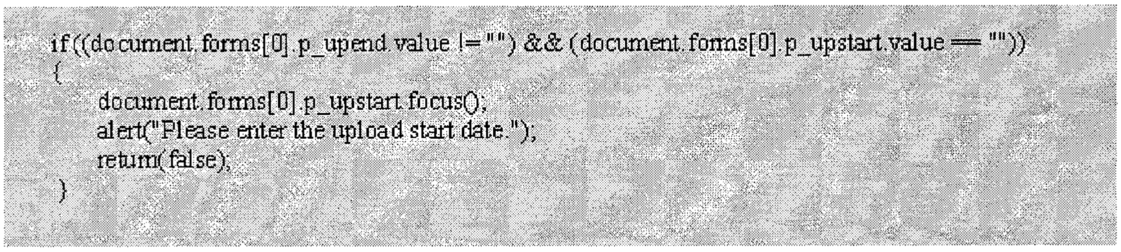
DI_Batch_Upload_log: This table stores the log of the batch i.e. in the batch, how many files were loaded successfully etc.

Table 17. Document Imaging Batch Upload Log

Attributes	Comments	Type
DBUL_Id	The unique key assigned to each database batch upload	Numeric
DBUL_Start_date	The date of the batch load run.	Date
DBUL_End_date	The date of the batch load end.	Date
DBUL_Clob	This field stores the detailed information about the names of the files loaded and the path of them	Clob
DBUL_File_Cnt	The number of files in the batch	Numeric
DBUL_Upload_Cnt	The number of the files loaded successfully into the database	Numeric
DD_CD	The foreign Key referencing to DD_CD of DI_department. It tells us for which department the batch process is running for.	Character

4.5 Programming Approach

The DI application is developed in PL/SQL toolkit which is actually a combination of HTML, JavaScript and PL/SQL code. The logic and the data manipulation (like select, delete or update) is done in PL/SQL language and JavaScript is used for browser initiated validation purpose.

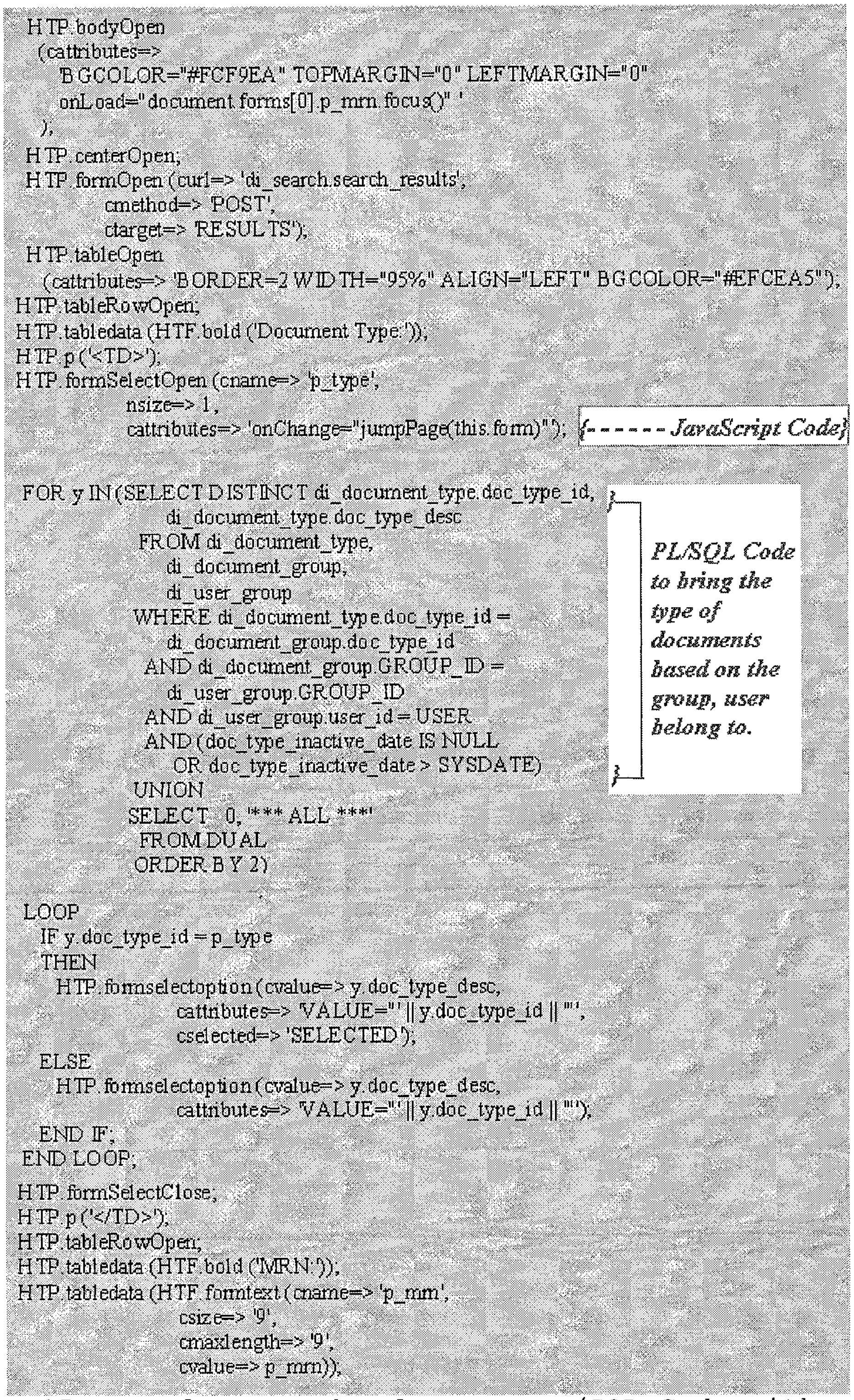


```
if((document.forms[0].p_upend.value != "") && (document.forms[0].p_upstart.value == ""))
{
      document.forms[0].p_upstart.focus();
      alert("Please enter the upload start date.");
      return(false);
}
```

Figure 14. Sample Javascript Code

Like in the above mentioned validation (see Figure 14), I am making sure if the user has entered the upload end date (p_upend.value) then the upload start date (p_upstart.value) can't be left black. User has to enter the date range as the search criteria for the query against the database.

For displaying the content in HTML format, I have used the standard package provided by the Oracle which is HTP package. There exist almost every procedure in this HTP package corresponding to each HTML tag for example



```
 HTP.bodyOpen
  (cattributes=>
    'BGCOLOR="#FCF9EA" TOPMARGIN="0" LEFTMARGIN="0"
    onLoad="document.forms[0].p_mrn.focus()" '
  );
 HTP.centerOpen;
 HTP.formOpen (curl=> 'di_search.search_results',
          cmethod=> 'POST',
          ctarget=> 'RESULTS');
 HTP.tableOpen
   (cattributes=> 'BORDER=2 WIDTH="95%" ALIGN="LEFT" BGCOLOR="#EFCEA5"');
HTP.tableRowOpen;
HTP.tabledata (HTF.bold ('Document Type:'));
HTP.p ('<TD>');
HTP.formSelectOpen (cname=> 'p_type',
            nsize=> 1,
            cattributes=> 'onChange="jumpPage(this.form)"');

 FOR y IN (SELECT DISTINCT di_document_type.doc_type_id,
                 di_document_type.doc_type_desc
              FROM di_document_type,
                 di_document_group,
                 di_user_group
             WHERE di_document_type.doc_type_id =
                 di_document_group.doc_type_id
              AND di_document_group.GROUP_ID =
                 di_user_group.GROUP_ID
              AND di_user_group.user_id = USER
              AND (doc_type_inactive_date IS NULL
                  OR doc_type_inactive_date > SYSDATE)
             UNION
            SELECT  0, '*** ALL ***'
              FROM DUAL
             ORDER BY 2)

 LOOP
   IF y.doc_type_id = p_type
   THEN
     HTP.formselectoption (cvalue=> y.doc_type_desc,
                 cattributes=> 'VALUE="' || y.doc_type_id || '"',
                 cselected=> 'SELECTED');
   ELSE
     HTP.formselectoption (cvalue=> y.doc_type_desc,
                 cattributes=> 'VALUE="' || y.doc_type_id || '"');
   END IF;
 END LOOP;
HTP.formSelectClose;
HTP.p ('</TD>');
HTP.tableRowOpen;
HTP.tabledata (HTF.bold ('MRN:'));
HTP.tabledata (HTF.formtext (cname=> 'p_mrn',
                 csize=> '9',
                 cmaxlength=> '9',
                 cvalue=> p_mrn));
```

{------ JavaScript Code}
PL/SQL Code to bring the type of documents based on the group, user belong to.

Figure 15. Sample Procedural Language/SQL Code with Hypertext Transfer Protocol Tags

HTP.HTMLOPEN, HTP>HTMLCLOSE which corresponds to <HTML> , </HTML> tags respectively. HTP.P can also be used, which displays the text in the HTML page, for example HTP.p ('this is a sample ') which is equivalent to <p>this is a sample</p>. Figure 15 is a good example to understand how JavaScript, HTML and PL/SQL are working together.

The approach in this project is to categories the functionality in procedure or function and then bundle them up in the packages. And as mentioned in the Application Server section, we have the leverage to call these packages or procedure directly from the browser with the condition that DAD (Database Access Descriptor) is set up properly.

To elaborate this programming approach considers the database model, there must exists user in the database who has access to this schema, in my case, DI_OWNER. Therefore, via Application Server GUI - Graphic User Interface, we can create DAD with username as DI_OWNER, its password and the connecting string say DI_DATABASE which is alias of the database where DI Schema is residing. I have created package DI_BATCH_PDF_LOAD (see Appendix B) which contains many procedures underneath. If user clicks on the "Upload Document" link in the HTML page using browser then the following request will be executed:

http://cslxowrl.edu:7777/di_batch_pdf_load.upload_file

where

cslxowrl.edu is the host where the web server is running.

7777 is port for the listener

/di_batch_pdf_load.upload_file is the DAD call.

Therefore in the nutshell, the Application Server will connect to the DI_DATABASE using username as DI_OWNER and its password and sends command to execute the DI_BATCH_PDF_LOAD.UPLOAD_FILE procedure in the database and returns the queried data with HTML content back to the user.

4.6 Error Handling and Security Concerns

As users will be scanning the medical records of the patients, this application is developed to address the following concerns:

4.6.1 Potential Problem # 1

There is a possibility that while scanning, the users mistyped the MRN#, in that scenario, the wrong file will get loaded into the database.

4.6.2 Solution

To avoid this kind of error to occur, the system handles this issue in two ways. Firstly, there will be only specific responsible users who would be able to scan the documents and the role assigned to the group of users is "Document Uploader". Rest will be assigned "Document Viewer". By doing this, we can minimize the number of cases where wrong MRN# being assigned to the file.

Secondly, while uploading the document, the system validates the MRN provided in the filename against the patient repository of the hospital. If there are no matches then the system won't load the file into the database and what it does is to rename it to *<Medical Record Type>*_*<MRN>*.PDF. BAD and sends the notification back to the users saying: System couldn't load the files into the database, please check the name or otherwise contact the support person. They can rename the files with proper MRN#s; therefore, in the next run of jobs, after validation, that file can be loaded into the system.

4.6.3 Security Concerns

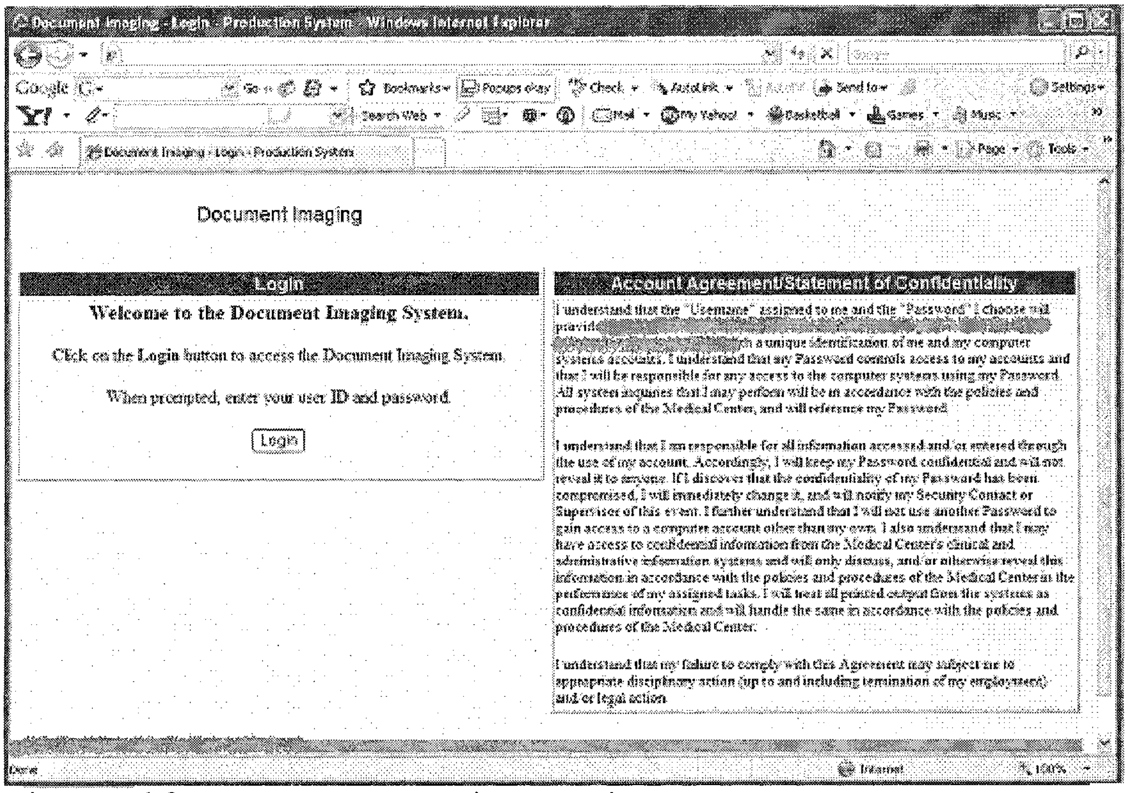
The major security concern is how to prevent unauthorized person accessing the patient's medical records. There is definitely the security improvised by Oracle Application Server, but, at the application level, the system addresses this concern in two ways. Firstly, by using the database authorization which implies restricting the unauthorized access to the database where the patient medical records are stored. Therefore, only person who has legitimate username and password can only access the database. In addition to that, the system automatically expires the session of the user, if not active for more than certain time which forces the users to login again, this minimizes the chances of desktop misuse. Secondly, The database is residing on our internal servers. And these servers are being protected by the firewall. No body will be able to access these servers from outside world.

CHAPTER FIVE

USERS MANUAL

5.1 Accessing the Document Imaging System

To log in to the DI system, double click the icon "DI system" on your desktops. The Document Imaging Login Screen will be displayed (Figure 16). Click the Login button. In the resulting dialog box (Figure 17), enter your username and password, and click on the OK button.


Document Imaging - Login - Production System - Windows Internet Explorer
Document Imaging
Login
Login
Account Agreement/Statement of Confidentiality
Internet
100%

Welcome to the Document Imaging System.

Click on the Login button to access the Document Imaging System.

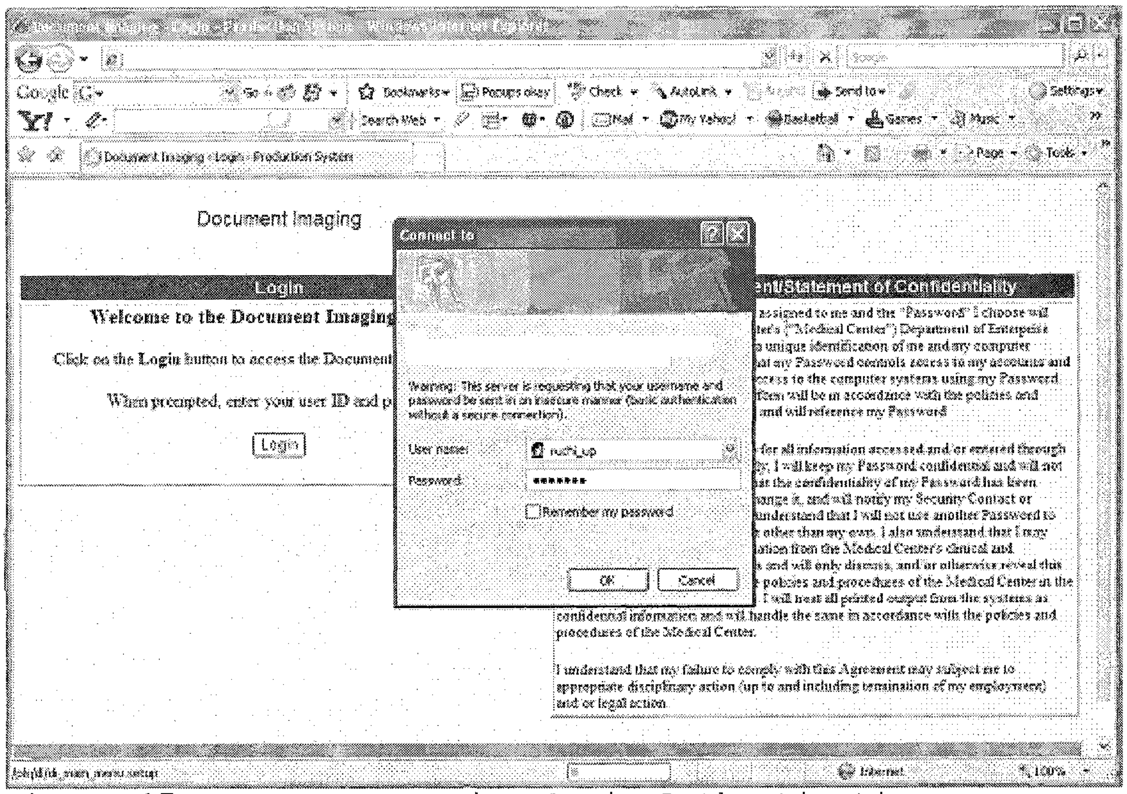
When prompted, enter your user ID and password.

I understand that the "Username" assigned to me and the "Password" I choose will provid[redacted]h a unique identification of me and my computer systems accounts. I understand that my Password controls access to my accounts and that I will be responsible for any access to the computer systems using my Password. All system inquiries that I may perform will be in accordance with the policies and procedures of the Medical Center, and will reference my Password.

I understand that I am responsible for all information accessed and/or entered through the use of my account. Accordingly, I will keep my Password confidential and will not reveal it to anyone. If I discover that the confidentiality of my Password has been compromised, I will immediately change it, and will notify my Security Contact or Supervisor of this event. I further understand that I will not use another Password to gain access to a computer account other than my own. I also understand that I may have access to confidential information from the Medical Center's clinical and administrative information systems and will only discuss, and/or otherwise reveal this information in accordance with the policies and procedures of the Medical Center in the performance of my assigned tasks. I will treat all printed output from the systems as confidential information and will handle the same in accordance with the policies and procedures of the Medical Center.

I understand that my failure to comply with this Agreement may subject me to appropriate disciplinary action (up to and including termination of my employment) and/or legal action.

Figure 16. Document Imaging Login Screen



Welcome to the Document Imaging

Click on the Login button to access the Document

When prompted, enter your user ID and p

Warning: This server is requesting that your username and password be sent in an insecure manner (basic authentication without a secure connection).

User name: ruchi_up

Password: •••••••

Remember my password

ent/Statement of Confidentiality

assigned to me and the "Password" I choose will
ter's ("Medical Center") Department of Enterprise
a unique identification of me and my computer
at my Password controls access to my accounts and
ccess to the computer systems using my Password
tem will be in accordance with the policies and
and will reference my Password

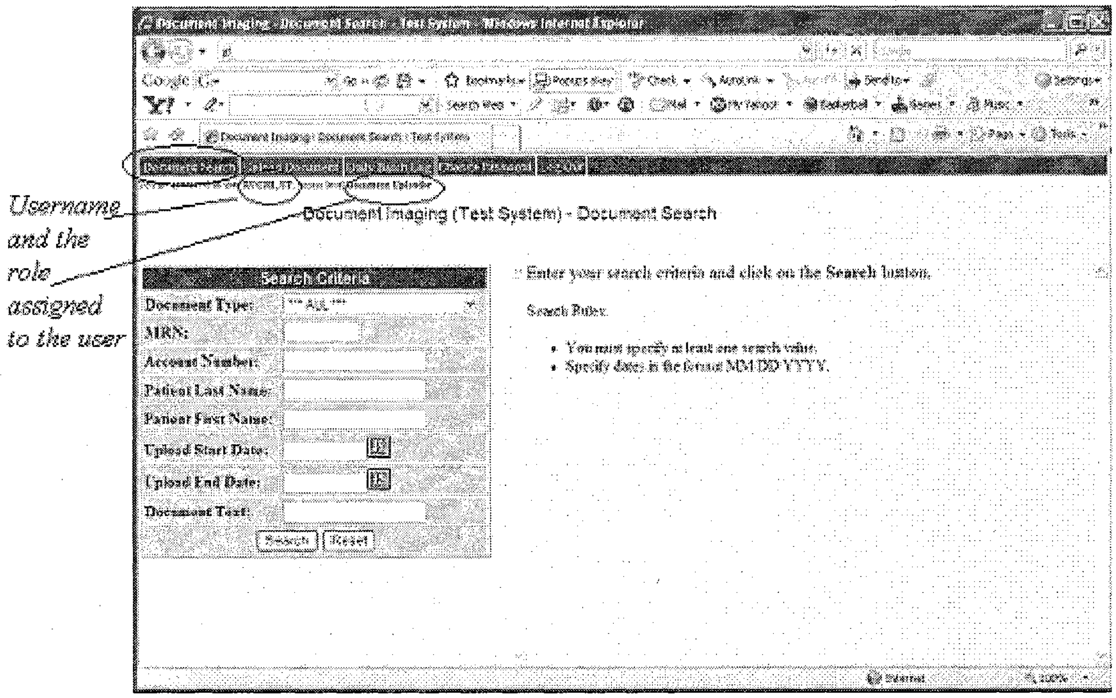
for all information accessed and/or entered through
y, I will keep my Password confidential and will not
at the confidentiality of my Password has been
ange it, and will notify my Security Contact or
understand that I will not use another Password to
other than my own. I also understand that I may
ation from the Medical Center's clinical and
s and will only discuss, and/or otherwise reveal this
policies and procedures of the Medical Center at the
I will treat all printed output from the systems as
confidential information and will handle the same in accordance with the policies and procedures of the Medical Center.

I understand that my failure to comply with this Agreement may subject me to appropriate disciplinary action (up to and including termination of my employment) and/or legal action.

Figure 17. Document Imaging Login Authentication

5.2 Performing the Document Search

After successfully logging in to the system, you will be taken to the Document Imaging screen. To navigate to the Search screen manually at any time, please click on the Document Search on the top menu. Just right below the top menu, you can see username and the role assigned. The navigation in the top menu will be based on the role assigned to the user (see Figure 18 and Figure 19).


Document Imaging - Document Search - Test System - Windows Internet Explorer
Username and the role assigned to the user
Search Criteria
Search
Reset

Document Imaging (Test System) - Document Search

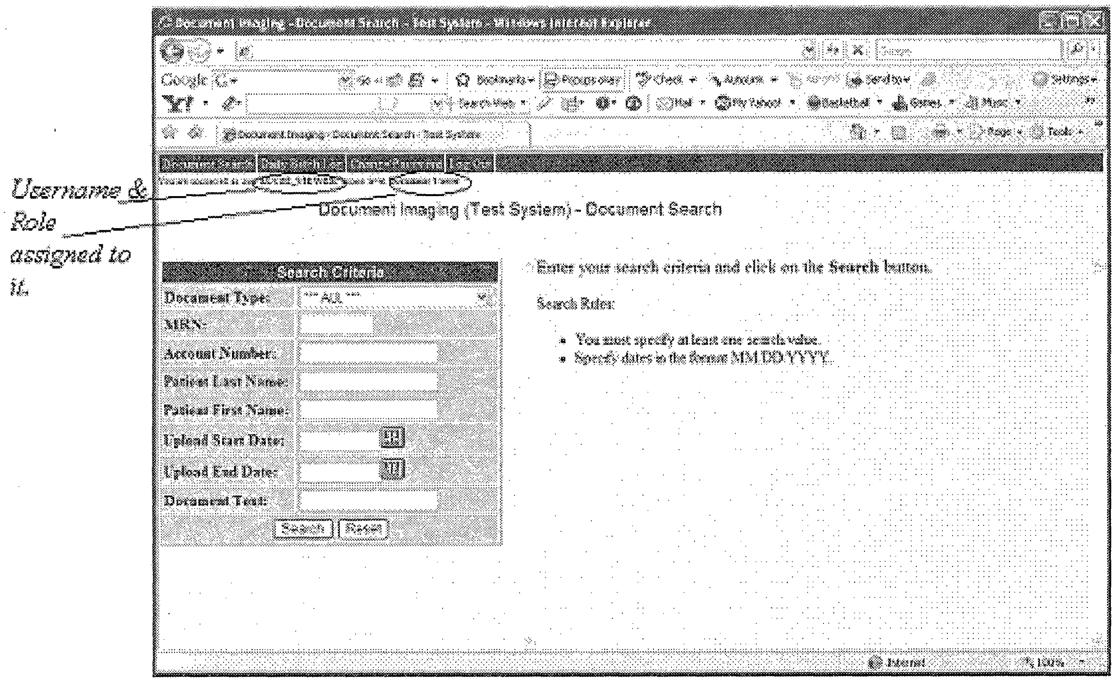
Document Type:	*** ALL ***
MRN:	
Account Number:	
Patient Last Name:	
Patient First Name:	
Upload Start Date:	
Upload End Date:	
Document Text:	

:: Enter your search criteria and click on the Search button.

Search Rules:

- You must specify at least one search value.
- Specify dates in the format MM/DD/YYYY.

Figure 18. Document Imaging Search Screen



Document Imaging (Test System) - Document Search

Search Criteria

Document Type:	*** ALL ***
MRN:	
Account Number:	
Patient Last Name:	
Patient First Name:	
Upload Start Date:	
Upload End Date:	
Document Text:	

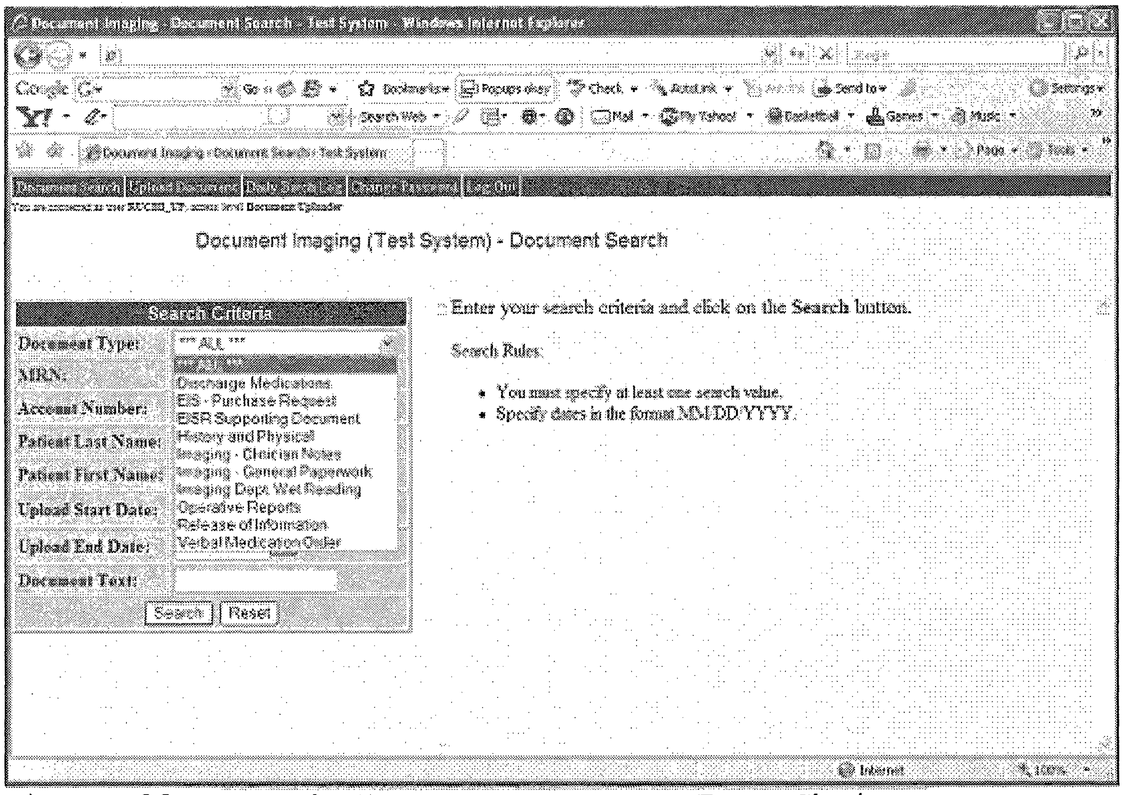
Enter your search criteria and click on the Search button.

Search Rules:

- You must specify at least one search value.
- Specify dates in the format MM/DD/YYYY.

Figure 19. Document Imaging Search Screen

To retrieve the desired files from the database, you can enter any known field related to the file(s). The document type is a drop down field, you can select any choice to search for that type of document (Figure 20).The choices may be different per username as it depends which group name your username belongs to.


Document Imaging - Document Search - Test System - Windows Internet Explorer

Document Search | Upload Documents | Daily Status Log | Change Password | Log Out

Document Imaging (Test System) - Document Search

Search Criteria

Document Type: *** ALL ***

*** ALL ***
Discharge Medications
E/S - Purchase Request
E/SR Supporting Document
History and Physical
Imaging - Clinician Notes
Imaging - General Paperwork
Imaging Dept. Wet Reading
Operative Reports
Release of Information
Verbal Medication Order

MRN:

Account Number:

Patient Last Name:

Patient First Name:

Upload Start Date:

Upload End Date:

Document Text:

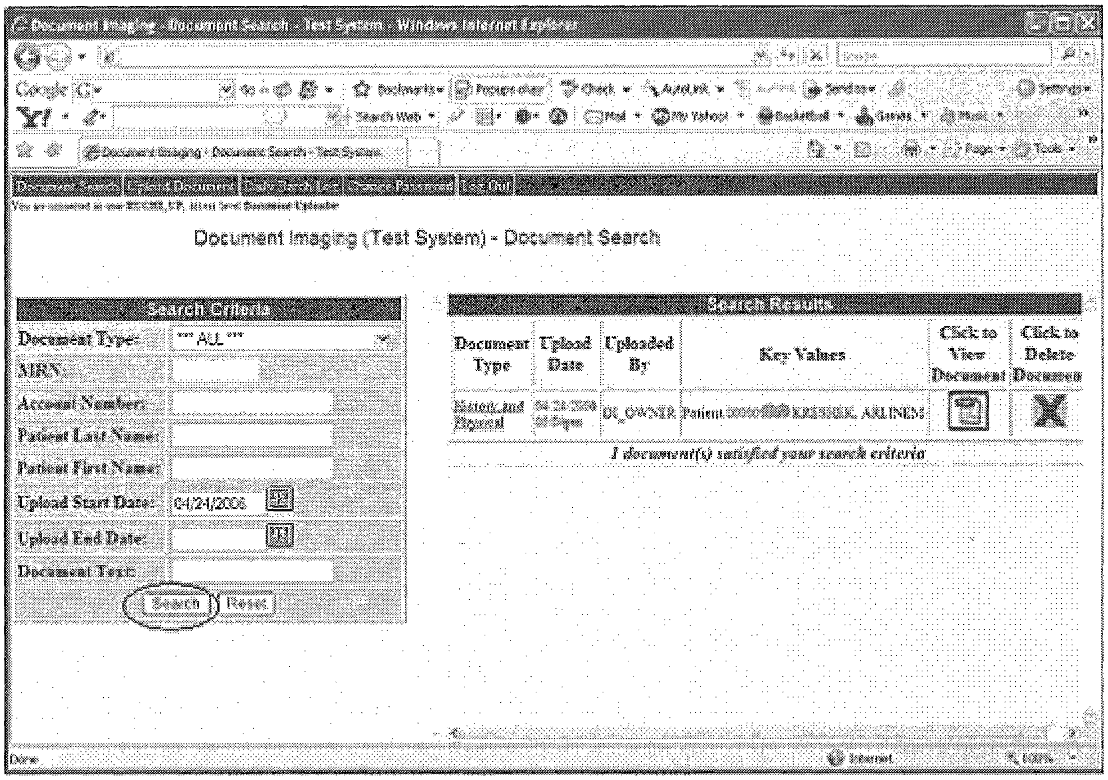
Search | Reset

Enter your search criteria and click on the **Search** button.

Search Rules:

- You must specify at least one search value.
- Specify dates in the format MM/DD/YYYY.

Figure 20. Search Screen - Document Type Choices



Document Imaging (Test System) - Document Search

Search Criteria

Document Type:	*** ALL ***
MRN:	
Account Number:	
Patient Last Name:	
Patient First Name:	
Upload Start Date:	04/24/2006
Upload End Date:	
Document Text:	

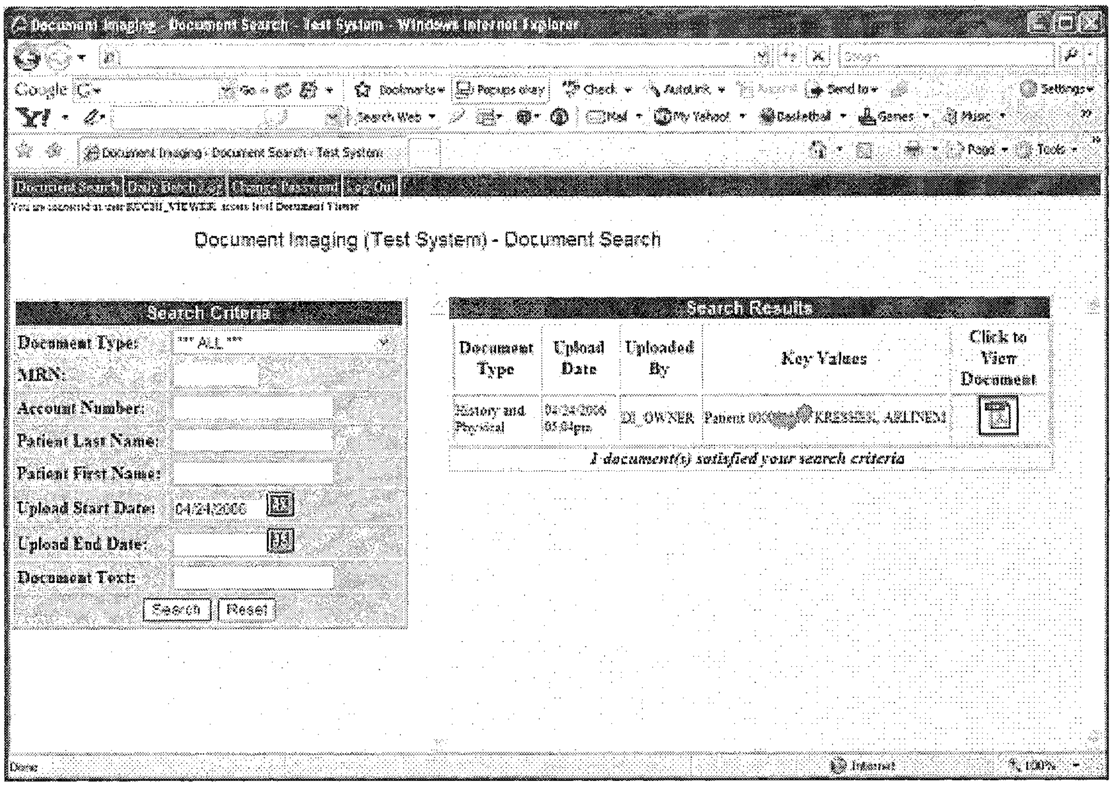
Search Results

Document Type	Upload Date	Uploaded By	Key Values	Click to View Document	Click to Delete Documen
History and Physical	04-24-2006 [illegible]	DI_OWNER	Patient [illegible] [redacted] KRESSICK, ARLINE		

1 document(s) satisfied your search criteria

Figure 21. Search Result for Document Uploader Role

After entering any criteria just hit "Search" on your screen. All the files satisfying your criteria will appear on the right hand side of the screen (Figure 21). Depending on your role, you might have only selecting privileges. Figure 22, if you have document viewer role.



Document Search | Daily Batch Log | Change Password | Log Out

Document Imaging (Test System) - Document Search

Search Criteria

Document Type:	*** ALL ***
MRN:	
Account Number:	
Patient Last Name:	
Patient First Name:	
Upload Start Date:	04/24/2006
Upload End Date:	
Document Text:	

Search | Reset

Search Results

Document Type	Upload Date	Uploaded By	Key Values	Click to View Document
History and Physical	04/24/2006 05:04pm	DI_OWNER	Patient 00[redacted]KRESSES, ARLINEM	

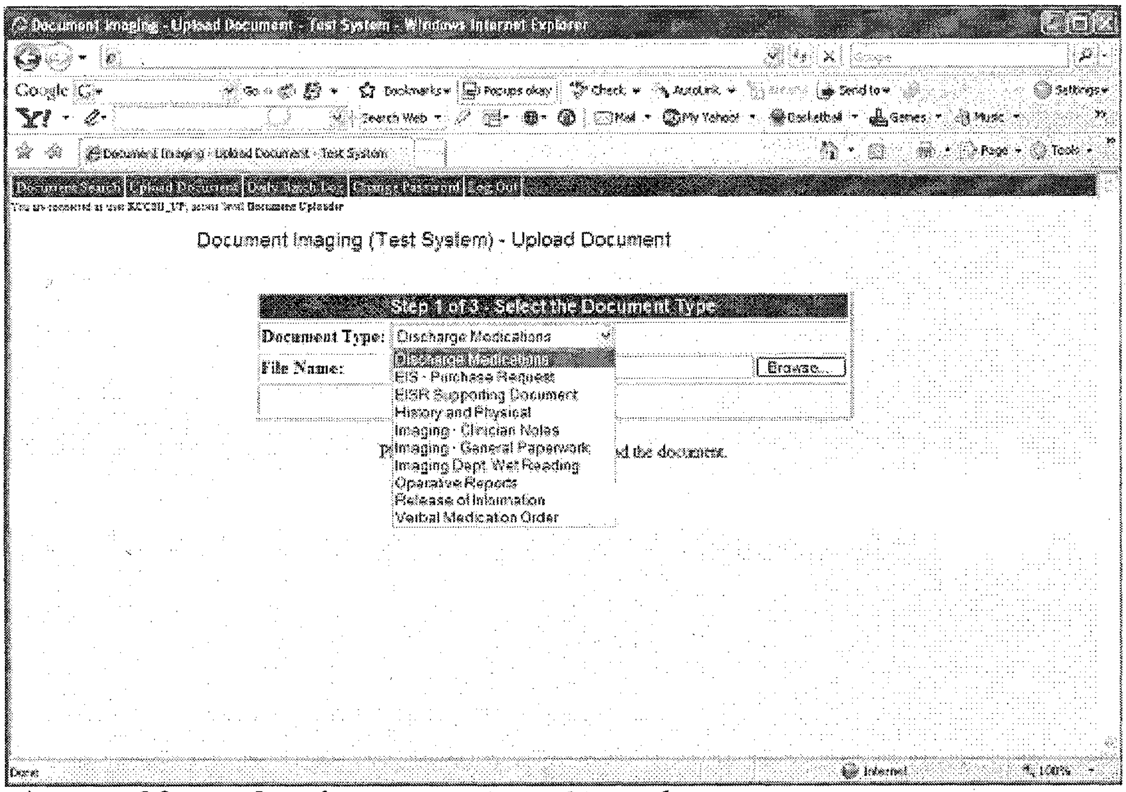
1 document(s) satisfied your search criteria

Figure 22. Search Screen Results for Document Viewer Role

As you can see, only Document uploader will have the privilege to delete the document from the database, if necessary. The "reset" button will clear the screen.

5.3 Uploading the New Document Manually

This tab is to manually upload the file into the database which is different from the automatic batch load. This is a three step process to load any file from your desktop into the database see Figure 23, Figure 24 and Figure 25


Document Imaging - Upload Document - Test System - Windows Internet Explorer
Document Search
Upload Document
Daily Batch Log
Change Password
Log Out
Browse...

Document Imaging (Test System) - Upload Document

Step 1 of 3 - Select the Document Type

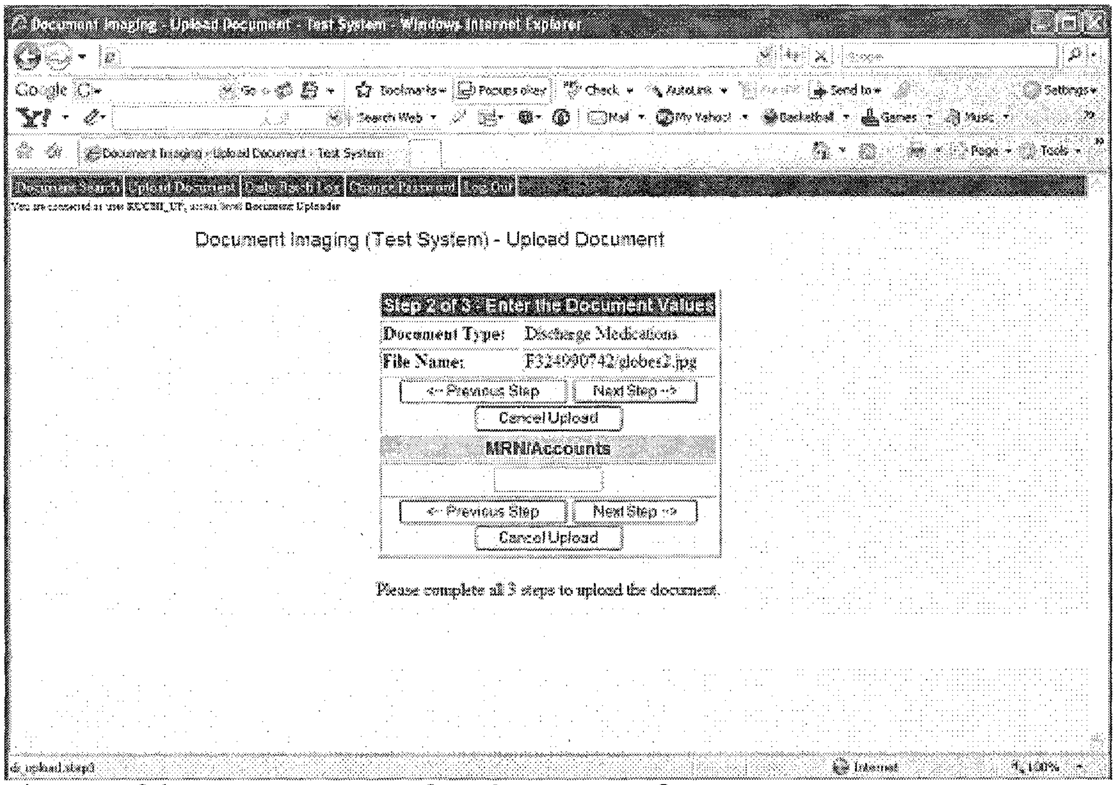
Document Type: Discharge Medications

File Name:

Discharge Medications
EIS - Purchase Request
EISR Supporting Document
History and Physical
Imaging - Clinician Notes
Imaging - General Paperwork
Imaging Dept. Wet Reading
Operative Reports
Release of Information
Verbal Medication Order

Figure 23. Upload Document - Step 1

First select the type of the document, you are uploading followed by the path of the filename by using the browse.



Document Imaging (Test System) - Upload Document

Step 2 of 3 - Enter the Document Values

Document Type: Discharge Medications

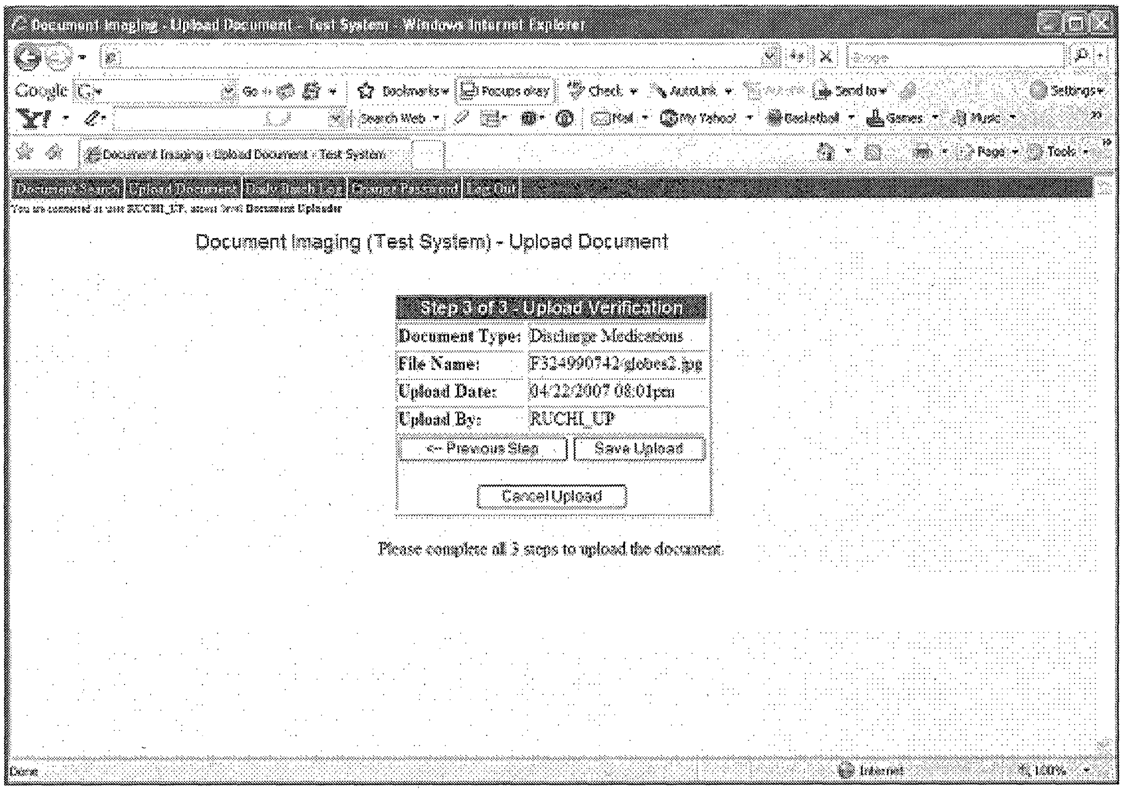
File Name: F324990742/globes2.jpg

MRN/Accounts

Please complete all 3 steps to upload the document.

Figure 24. Document Upload - Step 2

You can associate, if any, the name of the file with the MRN or the account in this screen. Choose any of the given options on the screen accordingly like "Next Step" to complete the upload, "Cancel Upload" to cancel the upload process, "Previous Step" to go back to the previous screen.



Document Imaging (Test System) - Upload Document

Step 3 of 3 - Upload Verification	
Document Type:	Discharge Medications
File Name:	F324990742/globes2.jpg
Upload Date:	04/22/2007 08:01pm
Upload By:	RUCHI_UP

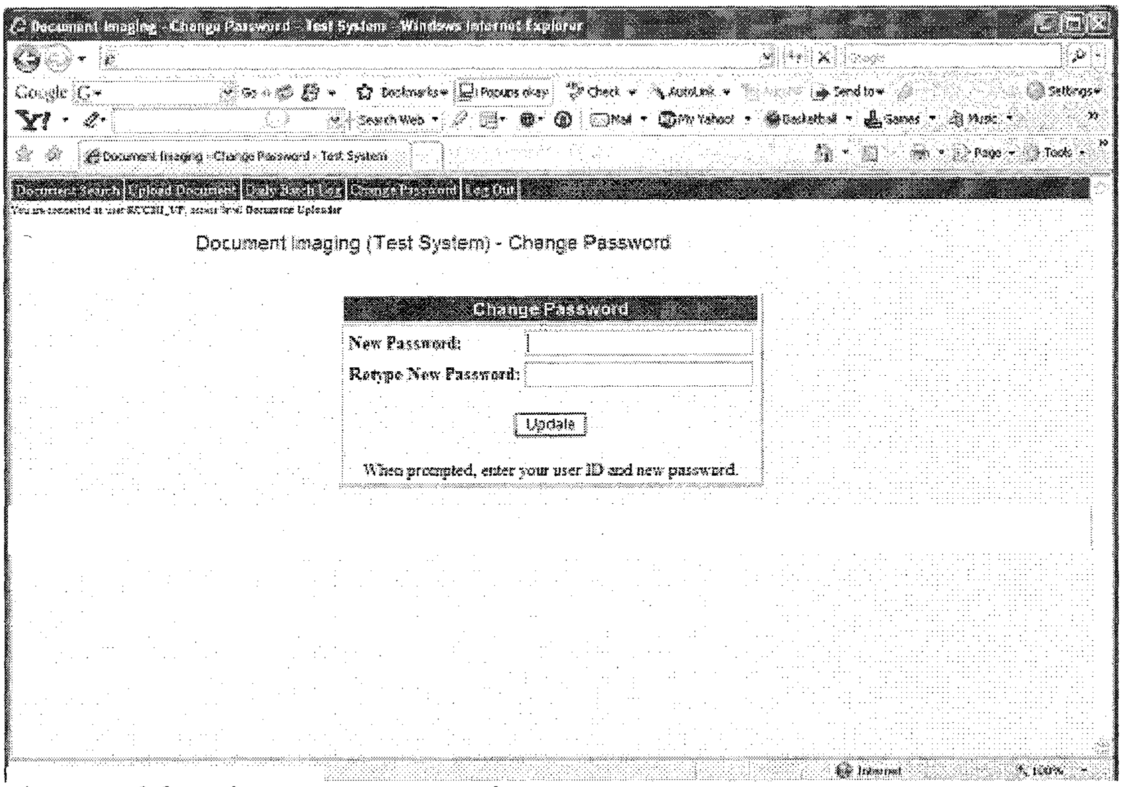
Please complete all 3 steps to upload the document.

Figure 25. Document Upload - Step 3

The third step is the confirmation step to verify that this is the file, you really want to upload into the database. After clicking "Save Upload" the document will be stored into the database.

5.4 Changing the Password

Users will have the privilege to change their passwords at any time. Just click on the "Change Password" in the top menu and following screen (Figure 26) will appear. Just type the same password in the both fields and click "Update". The password will be changed and to login thereafter, you have to use the new password.



Document Imaging (Test System) - Change Password

Change Password

New Password:

Retype New Password:

When prompted, enter your user ID and new password.

Figure 26. Change Password

CHAPTER SIX

CONCLUSIONS

6.1 Summary

Efficient applications can save companies millions of dollars in resources. These resources either can be personnel hours or the physical space. This application was developed to be very flexible, it allows customization according to the business flow. This project can be used as a reference when developing custom applications using Oracle tools or Oracle Application Server. But, for sure, there is always more room for exploration and erudition especially in the field of Oracle Application Server.

APPENDIX A

DEFINITIONS, ACRONYMS AND ABBREVIATIONS

Appworx
Tool used to schedule the shell scripts on specific time.

Shell Scripts
Scripts written in shell to move the files between servers.

PL/SQL
Programming language used to retrieve, insert and manipulate data on the Oracle database.

Database job
The DBMS_JOB package is used to schedule the jobs at the database level.

APPENDIX B

SAMPLE APPLICATION CODE

```
/* -------- Sample Package - Body Declaration --------------*/

PACKAGE DI_BATCH_PDF_LOAD IS
    PROCEDURE MAIN
    (
        p_directory IN VARCHAR2 DEFAULT '/apps/file/vmo',
        p_dept_code IN VARCHAR2 DEFAULT 'VMO',
        p_file_code IN VARCHAR2 DEFAULT 'VMO',
        p_logical_logdir IN VARCHAR2,
        p_logfile_name IN VARCHAR2,
        p_email_group IN mail_pkg.array_typ
    );

    PROCEDURE UPLOAD_FILE
    (
        p_filename IN VARCHAR2,p_logical_logdir IN VARCHAR2,
        p_logfile UTL_FILE.FILE_TYPE, p_dept_code IN VARCHAR2
    );

    /*
    this procedure show bad files with .BAD extention in the given location. When
shown the file extention is hidden and shown as PDF so that it will be easier for user
to rename.
    */

    PROCEDURE show_pdf_file
    (
        old_filename VARCHAR2 := NULL,
        new_filename VARCHAR2 := NULL,
        file_location VARCHAR2 := NULL
    );

    PROCEDURE view_pdf_file
    (
        loc IN VARCHAR2,
        file_name IN VARCHAR2
    );
END DI_BATCH_PDF_LOAD;
```

/* ----------- Sample Body Code --------------*/

```
------------------------------------------------------------------------------------------------------------
                                   Upload PDF files.
------------------------------------------------------------------------------------------------------------
    PROCEDURE upload_file (
        p_filename IN VARCHAR2, p_logical_logdir IN VARCHAR2,
        p_logfile IN UTL_FILE.FILE_TYPE,
        p_dept_code IN VARCHAR2
        )
    IS
        l_filename  di_document.doc_file_name%TYPE;
        l_blob    di_document.doc_blob%TYPE;
        l_docid   di_document.doc_id%TYPE;
        l_bfile   BFILE;
        l_ptid di_patient.pt_id%TYPE;
        l_doctype   di_document_type.doc_type_id%TYPE;
        l_indxINTEGER;
        l_posn    INTEGER;
        l_posn1 INTEGER;
        l_posn2 INTEGER;
        l_filetypeVARCHAR2 (100);
        l_parmcnt   INTEGER := 0;
        l_parms t_parms;
        l_case    VARCHAR2 (100);
        l_mrndi_patient.pt_mrn%TYPE;

        PRAGMA AUTONOMOUS_TRANSACTION;

    BEGIN
        l_filename := UPPER (p_filename);
        -- Parse file name, separated by underscores.
        -- Maximum of 10 parameters in file name.
        <<extract_keyvalues>>
          FOR l_indx IN 1 .. 10
          LOOP
            l_posn1 := INSTR (l_filename, '_');
            l_posn2 := INSTR (l_filename, '.');
            IF l_posn1 = 0 AND l_posn2 = 0
            THEN
              l_filetype := l_filename;
              EXIT;
            ELSE
              l_parmcnt := l_parmcnt + 1;
              IF (l_posn1 < l_posn2) AND (l_posn1 > 0)
              THEN
                l_posn := l_posn1;
              ELSE
                l_posn := l_posn2;
          END IF;

l_parms (l_indx) := SUBSTR (l_filename, 1, l_posn - 1);
l_filename := SUBSTR (l_filename, l_posn + 1);
END IF;

END LOOP extract_keyvalues;

-- Check if parsing was successful.
IF (l_parmcnt < 2) OR (l_filetype IS NULL)
THEN
   ROLLBACK;
   log_message (
          'Incorrect File Name.', NULL, NULL,
          p_dept_code, 'File name: ' || p_filename
          );
   RAISE upload_failed;

-- Additional validation for MDN documents.
ELSIF
(l_parms (1) IN ('MDNE', 'MDNP', 'MDNC', 'MDNX')) AND
(l_parms (2) <> l_parms (3))
THEN
      ROLLBACK;
      log_message
      (
      'For MDN Files, parameter2 and parameter3 specified
      in the file name must match .',
      NULL,
      NULL,
      p_dept_code,
      'File name: ' || p_filename
      );
   RAISE upload_failed;
END IF;

-- Determine the document type using parameter #1.
<<get_document_type>>
BEGIN
   SELECT doc_type_id
      INTO l_doctype
      FROM di_document_type
   WHERE doc_type_batch_cd = l_parms (1);

   EXCEPTION
          WHEN NO_DATA_FOUND
          THEN
                 ROLLBACK;
          log_message
```

```
        (
            'No Document Type found for column
            doc_type_batch_cd of ' ||l_parms (1),
            SQLCODE,
            'DI_DOCUMENT_TYPE',
            p_dept_code,
            'File name: ' || p_filename
        );
        RAISE upload_failed;

        WHEN OTHERS
        THEN
                ROLLBACK;
                log_message (
                        SUBSTR (SQLERRM, 1, 500),
                        SQLCODE, 'DI_DOCUMENT_TYPE',
                        p_dept_code, 'File name: ' || p_filename
                        );
                RAISE upload_failed;

END get_document_type;

-- Check if media type exists.
<<add_media_type>>
BEGIN
    INSERT INTO di_media_type
        (media_type_text,
        media_type_create_date,
        media_type_mimetype_text)
    VALUES (l_filetype, SYSDATE, '*' );

EXCEPTION
WHEN DUP_VAL_ON_INDEX
THEN
    NULL; -- Media type exists. Don't need to insert.

WHEN OTHERS
THEN
    ROLLBACK;
    log_message (
    'Error when adding media type for file type of ' ||
    l_filetype ||SUBSTR (SQLERRM, 1, 500), SQLCODE,
    'DI_MEDIA_TYPE', p_dept_code, 'File name: ' ||
    p_filename
    );
    RAISE upload_failed;

END add_media_type;
-- Insert the document to table DI_DOCUMENT.
```

```
     SELECT di_doc_id_seq.NEXTVAL
        INTO l_docid
        FROM DUAL;

  <<add_document>>
     BEGIN
        INSERT INTO di_document
           (doc_id, doc_create_date, doc_create_user_id,
           doc_file_name, doc_blob,
           doc_length, doc_type_id, media_type_text)
        VALUES (l_docid, SYSDATE, USER, p_filename,
              EMPTY_BLOB (),
              DBMS_LOB.getlength (l_blob), l_doctype,
              l_filetype)RETURNING doc_blob INTO l_blob;
        -- need to map logical directory
        l_bfile := BFILENAME (p_logical_logdir,p_filename);
        DBMS_LOB.fileopen (l_bfile);
        DBMS_LOB.loadfromfile (l_blob, l_bfile,
        DBMS_LOB.getlength (l_bfile));
        DBMS_LOB.fileclose (l_bfile);
        UPDATE di_document
           SET doc_length = dbms_lob.getlength(doc_blob)
        WHERE doc_id = l_docid;
     EXCEPTION WHEN OTHERS
     THEN
        ROLLBACK;
        log_message
        ('Error when adding document. ' ||
        SUBSTR (SQLERRM, 1, 500),
        SQLCODE, 'DI_DOCUMENT',
        p_dept_code, 'File name: ' || p_filename);

           IF DBMS_LOB.ISOPEN(l_bfile) = 1
           THEN
              DBMS_LOB.fileclose (l_bfile);
           END IF;
           RAISE upload_failed;
        END add_document;

        <<validate_document_keyvalues>>
           FOR x IN (SELECT *
                    FROM di_document_type_parm
                    WHERE
                    (doc_type_id = l_doctype) AND
                    (didtp_parm_no IS NOT NULL)
                       ORDER BY didtp_parm_no)
           LOOP
              IF x.didtp_desc = 'MRN'
              THEN
```

```
          -- Insert document keyvalues.
          -- Handle MRN keyvalue.
          -- If length=9 then value is MRN.
          -- If length=12 then value is CASE and MRN.

          IF LENGTH (l_parms (x.didtp_parm_no)) = 9
          THEN
             l_mrn := l_parms (x.didtp_parm_no);
             l_ptid :=
                 process_mrn (p_filename, l_mrn, l_docid,
                 p_dept_code);
          ELSIF LENGTH (l_parms (x.didtp_parm_no)) = 12
          THEN
             l_case := SUBSTR (l_parms
                     (x.didtp_parm_no), 1, 3);
             l_mrn := SUBSTR (l_parms (x.didtp_parm_no),
                     4, 9);
             l_ptid := process_mrn (p_filename, l_mrn,
                     l_docid, p_dept_code);
             process_acct (p_filename, l_ptid, l_case,
                     l_mrn, l_docid, p_dept_code);
          ELSE
             ROLLBACK;
             log_message
             ('Invalid MRN number. ', NULL, NULL,
             p_dept_code, 'File name: ' || p_filename);
             RAISE upload_failed;
       END IF;
    ELSE
       -- Handle other keyvalues.
       IF (l_parms.EXISTS (x.didtp_parm_no)) AND
          (l_parms (x.didtp_parm_no) IS NOT NULL)
       THEN
          <<add_document_keyvalues>>
             BEGIN
                 INSERT INTO di_document_keyvalue (
                     doc_id, doc_type_id, key_id,
                     doc_key_create_date, doc_key_value
                     )
             VALUES (
                     l_docid, l_doctype, x.key_id,
                     SYSDATE, l_parms (x.didtp_parm_no)
                     );
          EXCEPTION
             WHEN DUP_VAL_ON_INDEX
             THEN
                 NULL; -- Don't need to add.
             WHEN OTHERS
```

```
                    THEN
                        ROLLBACK;
                        log_message
                            ('Error when adding document key values.'
                            ||SUBSTR (SQLERRM, 1, 500), SQLCODE,
                        'DI_DOCUMENT_KEYVALUE',
                        p_dept_code, 'File name: ' || p_filename
                        );
                    RAISE upload_failed;
                END add_document_keyvalues;
            END IF;
        END IF;
END LOOP validate_document_keyvalues;

COMMIT;
-- Display list of MRN.
-- Will be included in email log to users.
-- replace: write to a file . use utl_file
-- DBMS_OUTPUT.PUT_LINE('-*-*-');

FOR x IN (SELECT B.PT_LAST_NAME || ', ' ||
                B.PT_FIRST_NAME || ' ' ||
                B.PT_MIDDLE_NAME PTNAME, B.PT_MRN
                FROM DI_DOCUMENT_PATIENT A, DI_PATIENT
                B
                WHERE A.PT_ID = B.PT_ID
                    AND A.DOC_ID = l_docid
                ORDER BY B.PT_MRN)
LOOP
IF (l_case IS NULL)
THEN
        CASE p_dept_code
        WHEN 'ESR'
            THEN
        log_di_run
        (
        'DI_BATCH_PDF_LOAD',
        'MRN: '|| x.pt_mrn || ' Patient: ' || x.ptname,
        SYSDATE
        );
        ELSE
        UTL_FILE.PUT_LINE
        (p_logfile, 'MRN: ' || x.pt_mrn || ' Patient: ' || x.ptname);
    END CASE;
ELSE
    CASE p_dept_code
    WHEN 'ESR'
        THEN
    log_di_run
```

```
                (
                    'DI_BATCH_PDF_LOAD',
                    'MRN: ' || x.pt_mrn || ' Case: ' || l_case || '
                    Patient: ' || x.ptname,
                SYSDATE
                );
                ELSE
                UTL_FILE.PUT_LINE (p_logfile, 'MRN: ' || x.pt_mrn ||
                ' Case: ' || l_case || '
                Patient: ' || x.ptname);
                END CASE;
        END IF;
    END LOOP;
END – upload_file ;
```

APPENDIX C

SAMPLE APPWORX JOB

```
#!/bin/ksh

#-- Deleting all the existing PDF files from the server
#-- so that the files will be loaded into the database
#-- only one time .

cd /server1/apps/dcm/data
rm -f /server1/apps/dcm/data/*.PDF
rm -f /server1/apps/dcm/data/*.pdf
smbclient //shared_server/dcm rsukhija -U ACCOUNTS\\sukhijar << EOF
prompt
mget *.pdf
rename *.pdf *.pdf.OK
quit
EOF


#delete *.pdf
cd /server2/apps/dcm/data
#rm *.pdf

#-- Logging into the designated server to copy the files

ftp -n databaseserver << EOF
quote user ruchisukhija
quote pass sukhijar
cd /apps/file/dcm
bin
prompt
```

APPENDIX D

SAMPLE DATABASE JOB

```
variable a number;
execute
dbms_job.submit
(:a,
'di_batch_pdf_load.main("/apps/file/roi",
                         "ROI",
                         "ROI",
                         "ROI_DIR",
                         "upload_roi_files.log",
                         mail_pkg.array_typ("sukhijar@cshs.org"));',
trunc(SYSDATE)+ 6/24,
'trunc (sysdate) + decode (to_char (sysdate, "d"), 6, 3, 1) + 6/24');
commit;
```

REFERENCES

[1] Elmasri, Ramez and Navathe, Shamkant, Fundamentals of Database Systems, Fourth edition, 2004

[2] Moncur, Michael, Teach Yourself JavaScript, Sams, Third Edition, 2002

[3] Brown, Bradley, Oracle 9i Web Development, Osborne McGraw-Hill, 2001

[4] Urman, Scott, Oracle 8 PL/SQL Programming, Osborne McGraw-Hill, 1997

[5] Banerjee, Saurabh, Oracle Application Server 10g, Oracle University, Second Edition, November 2005

[6] Holm, Bjarki, Carnell, John, Stubbs, Tomas, Sarang, Poornachandra, Mukhar, Kevin, Singh, Sant, Goodman, Jaeda, Marcotte, Ben, Naranjo, Maurico, Raj, Anand, Piermanini, Mark, Oracle 9i Java Programming: Solutions for Developers Using PL/SQL and Java, Wrox, 2003