**A Minor Project Report**

On

**NORMALIZATION OF DUPLICATE RECORDS FROM MULTIPLE SOURCES**

Submitted in partial fulfilment of requirements for the award of the Degree of

## BACHELOR OF ENGINEERING

in

## COMPUTER SCIENCE AND ENGINEERING

Under the guidance of

## Ms. I. KARTHIKA

## ASSISTANT PROFESSOR/CSE

Submitted By

## MOHAMEDNOWFAL.S(19BCS435)HARIHARAN.G (19BCS4303) DINESHWARAN (19BCS4018)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**M.KUMARASAMY COLLEGE OF ENGINEERING**

(Autonmous) **KARUR – 639 113** May,2022.

**NORMALIZATION OF DUPLICATE RECORDS FROM MULTIPLE RESOURCES**

# PROJECT REPORT

Submitted by

# REG.NO NAME

# Submitted for partial fulfillment of the award of Diploma in

**COMPUTER ENGINEERING**

# of

**DIRECTORATE OF TECHNICAL EDUCATION GOVERNMENT OF TAMILNADU**

**BONAFIDE CERTIFICATE**

Certified that this project report **“NORMALIZATION OF DUPLICATE RECORDS FROM MULTIPLE RESOURES”** is the bonafide work who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **REG.NO** | **NAME** |
|  |  |
|  |  |
|  |  |
|  |  |

## SIGNATURE SIGNATURE

**Mrs. M.SUGANYA M.E., Mr. K. RAMESHKUMAR M.E.,**

**PROJECT GUIDE /LECTURER HEAD OF THE DEPARTMENT**

Computer Engineering Computer Engineering

Kongu Hi-Tek Polytechnic college Kongu Hi-Tek Polytechnic College

Karur – 639201 Karur- 639201

Submitted for the Board Examination held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

# ACKNOWLEDGEMENT

I wish to thank the almighty for giving me such an opportunity. I express, my sincere gratitude to my parents who steadfastly behind me in all my efforts to the successful completion of this project.

I am extremely grateful to our respectable Chairman **Shri.R.Palanisamy** and Secretary **Mr.N.K.Kandasamy M.Sc., M.Phil., Ph.D** for giving me the opportunity to study in this prestigious institution. With profound sense of gratitude, I would like express my sincere thanks to our Principal **Mr. Ramesh M.E, MBA, Ph.D.,** and our Administrative Officer **Mr.V.P.Suresh B.A., B.L., MBA,** for providing me with moral help and encouragement for completing this project successfully.

I wish to place on the record the profound feeling of gratitude to the Head of the Department and our Overall Guide **Mr. K.RameshKumar M.E.,** for his encouragement throughout the project work. I wish to express my sincere thanks to our Project Guide **Mrs. Suagnya M.E.,** for their help and excellent guidance throughout my project.

I wish to thank all the staff members of our department who helped me a lot for the successful completion of this project. Finally, I would like to have a mention of thanks to my parents and friends who gave us the moral strength and source of incessant encouragement for completing my project successfully.

## NORMALIZATION OF DUPLICATE RECORDS FROM MULTIPLE RESOURCES

**ABSTRACT**

Data consolidation is a challenging issue in data integration. The usefulness of data increases when it is linked and fused with other data from numerous (Web) sources. The promise of big data hinges upon addressing several big data integration challenges, such as record linkage at scale, real-time data fusion, and integrating Deep Web. Although much work has been conducted on these problems, there is limited work on creating a uniform, standard record from a group of records corresponding to the same real-world entity. We refer to this task as record normalization. Such a record representation, coined normalized record, is important for both front-end and back-end applications. In this paper, we formalize the record normalization problem, present in- depth analysis of normalization granularity levels and of normalization forms (e.g., typical versus complete). We propose a comprehensive framework for computing the normalized record. The proposed framework includes a suit of record normalization methods, from naive ones, which use only the information gathered from records themselves, to complex strategies, which globally mine a group of duplicate records before selecting a value for an attribute of a normalized record. We conducted extensive empirical studies with all the proposed methods. We indicate the weaknesses and strengths of each of them and recommend the ones to be used in practice.

# ABBREVIATION

RDC Remote differential compression

CDV Content defined chunking

LMC Local Maximum Chunking

AE Asymmetric Extremum

RAM Rapid Asymmetric Maximum

SLA Service Level Agreement

AWS Amazon Web Services

VMs Virtual Machines

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **TITLE** | **PAGE** |
| **No.** |  | **No.** |
|  | Abstract | 1 |
|  | Abbreviations | 2 |
| 1 | Introduction | 4 |
|  | 1.1 Objective | 6 |
|  | 1.2 Aim |  |
| 2 | Literature Survey | 7 |
| 3 | Cloud Computing | 10 |
| 4 | Architecture | 11 |
| 5 | Modules | 12 |
|  | 5.1 Module Description & module diagrams |  |
|  | 5.2 User Authentication |  |
| 6 | Software description | 14 |
|  | 6.1 General |  |
|  | 6.2 Features of java |  |
|  | 6.3 Objectives of java |  |
|  | 6.4 Object oriented |  |
| 7 | Features | 21 |
|  | 7.1 Backup Software |  |
| 8 | Coding | 26 |
| 9 | Screenshot | 38 |
| 10 | Conclusion | 42 |

**CHAPTER -1 INTRODUCTION**

The Internet is one of the important technology advancements of recent times. Knowledge and media sharing are easier than before because the Internet provides data access everywhere and anywhere, and interconnects devices and machines around the world. However, the bandwidth requirement and the amount of data to be stored increase as the number of internet users increases. This has led to challenges researchers to develop high bandwidth networks and techniques to reduce the amount of data to be transferred.

A technique to reduce the amount of data transfer is to use compression. Data compression algorithms work by eliminating redundant data within a file or multiple files. In particular, a data compression technique called differential compression eliminates the amount of data to be transferred by sending only the differences between files. Differential compression works by comparing two files and only sends the differences. For example, when a user wants to update a file, the server compares and finds the difference between the old version and the new version. The server sends the difference, and the client is required to process the differences and the old file to get the new file. In this scenario, the differential compression technique is called local differential compression.

Remote differential compression (RDC) refers to another situation where the client has a file F1 and the server has a similar file F2. In this situation, the server and the client work together to discover the differences between the two files, and only the differences are sent. RSYNC by Trudgill et al. which is an RDC protocol, does remote differential compression by splitting a file into multiple fixed length chunks. Checksums or hash values are calculated from the chunks and sent to the server. The server does the same thing to the file that it believes to be similar and compares the hashes. Only chunks with different hashes are sent to the client, thus reducing the amount of data to be transferred.

The fix-sized chunk used in RSYNC has a problem. Bjørner et al. stated it cannot identify two identical files, F1 and F2, where F2 is F1 with a byte inserted at the beginning of the file. When F1 and F2 are processed using the RSYNC protocol described in the previous paragraph, no checksums of F1 matches the checksums of F2. This is called the byte shifting problem. Content defined chunking (CDC) algorithms fixed this problem by chunking the file based on the internal features of a file. Similar to RDC, CDC algorithms are also used in data deduplication to split a file into chunks for detecting duplicate data.

In the cloud storage ecosystem, differential compression is known as data deduplication. Data deduplication is important because it saves storage spaces by eliminating redundant data within files and between files. Data deduplication involves three main components: chunking, hashing, and comparing hashes to detect redundancy and it works by chunking the files into chunks and comparing the chunks by using a mathematical hash function. In a deduplication system, chunking defines the characteristic of the system because chunking algorithms used in the system can make the system more effective and faster.

One of the most known chunking algorithms is Rabin based chunking algorithm. Rabin based chunking algorithm uses Rabin rolling hash by Michael O. Rabin to find the cut- point. The rolling hash works by using a sliding window, and a hash value is calculated when the sliding window is moving. Rabin rolling hash consumes a considerable amount of time for calculating the hashes. Unlike Rabin based chunking, Local Maximum Chunking (LMC), which is proposed by Bjørner et al. uses byte values of a file and sliding window to determine the cut point. In this approach, no hash calculation is required but it uses comparisons for each byte processed. Another variant of the local maximum chunking is Asymmetric Extremum (AE) which uses a fixed length window and a variable-sized window to determine a cut point. The extreme value is located between the two windows. Unlike local maximum, AE does not use sliding window to find the cut points.

Thus, it requires fewer comparisons because the local maximum needs to find the extreme value in the sliding window each time the window slides. Although the number of comparisons is significantly reduced for AE compared to LMC, AE still does a lot of comparisons which might be expensive for performance limited devices. The aim of this work is to reduce the computational overhead of the existing CDC algorithm. We achieved the lower computational overhead by harnessing the statistical properties of the algorithm to minimize the number of comparisons.

This paper proposes a new Rapid Asymmetric Maximum (RAM) chunking algorithm which is based on the AE algorithm. Similar to AE, the algorithm uses two windows: a fixed- length window and a variable-sized window. Unlike AE, RAM uses different positions for the windows. The windows position is started with the fixed-length window and followed by the variable-sized window and the maximum sized byte. The cut point is located at the end of the chunk, which puts the maximum-valued byte at the end of the chunk. This configuration has different statistical properties than AE’s configurations, which reduces the number of comparisons. Thanks to the reduced number of comparisons, RAM performs better than AE in terms of deduplication throughput. In addition to the reduced number of comparisons, the

algorithm produces chunks with sizes distribution similar to AE’s. The contributions of this paper are as follows:

* Studies on multiple chunking algorithms;
* Proposes RAM, a fast and hash-less CDC algorithm;
* Analyses the properties of RAM and compares it with other CDC algorithms;
* Carried out performance evaluation of RAM and compares it with other CDC algorithms to learn the performance of RAM compared to other CDC algorithms when used in a deduplication system. The rest of this paper is organized as follows. Section 2 explains CDC algorithms, its problem, and our motivation. Section 3 discusses the design and analysis of our proposed RAM algorithm. In Section 4, we compare RAM with other CDC algorithms, followed by discussion of the results in Section 5. Lastly, we conclude our paper in Section 6.

# OBJECTIVE

In this paper, we proposed a new algorithm called Rapid Asymmetric Maximum (RAM) which improves the chunking throughput of AE by putting the extreme value at the boundary of the chunk. It has a low computational overhead which makes the algorithm faster than existing CDC algorithms. The low computation overhead of RAM reduces the cost of chunking process which makes chunking more attractive over AE for low performance devices such as mobile devices and IoT.

# AIM

* Proposes RAM, a fast and hash-less CDC algorithm for low performance devices.
* Carried out performance evaluation and compares it with other CDC algorithms.
* In some cases, RAM offers 26% to 40% higher byte saved per second compared to the other chunking algorithms.

# CHAPTER -2 LITERATURE SURVEY

**Title :** Towards a Self-Adaptive Data Management System for Cloud

Environments

**Author:** Alexandra Carpen-Amarie

**Year :** 2011

## Description:

More specifically, storage systems intended for very large scales have to address a series of challenges, such as a scalable architecture, data location transparency or high throughput under concurrent accesses. Although these requirements are the prerequisites for any efficient data management system, they also imply a high degree of complexity in the configuration and tuning of the system, with possible repercussions on the system’s availability and reliability. Such challenges can be overcome if the system is outfitted with a set of self-management mechanisms that enable an autonomic behavior, which can shift the burden of understanding and managing the system’s state from the human administrator to an automatic decision- making engine. However, self-adaptation is impossible without a deep and specific knowledge of the state of both the system and the infrastructure where the system is running. It heavily relies on introspection mechanisms, which play the crucial role of exposing the system’s behaviour accurately and in real time.

**Title :** Bringing introspection into blob seer: towards a self-adaptive distributed data

management system

**Author:** alexandru costan, jing cai

**Year :** 2011

## Description:

Introspection is the prerequisite of an autonomic behaviour, the first step towards a performance improvement and a resource-usage optimization for large-scale distributed systems. In Grid environments, the task of observing the application behaviour is assigned to monitoring systems. However, most of them are designed to provide general resource information and do not consider specific information for higher-level services. More precisely, in the context of data-intensive applications, a specific introspection layer is required to collect data about the usage of storage resources, about data access patterns, etc. This paper discusses the requirements for an introspection layer in a data-management system for large-scale distributed infrastructures. We focus on the case of Blob Seer, a large-scale distributed system

for storing massive data. The paper explains why and how to enhance BlobSeerwith introspective capabilities and proposes a three-layered architecture relying on the Mona LISA monitoring framework. We illustrate the autonomic behaviour of BlobSeer with a self- configuration component aiming to provide storage elasticity by dynamically scaling the number of data providers. Then we propose a preliminary approach for enabling self-protection for the BlobSeer system, through a malicious clients detection component. The introspective architecture has been evaluated on the Grid’5000 testbed, with experiments that prove the feasibility of generating relevant information related to the state and the behaviour of the system.

**Title :** Optimizing storage performance in public cloud platforms

**Author :** Jian-zong WANG, Peter VARMAN

**Year** 2011

## Description:

Cloud computing is an elastic computing model where users can lease computing and storage resources on demand from a remote infrastructure. It is gaining popularity due to its low cost, high reliability, and wide availability. With the emergence of public cloud storage platforms like Amazon, Microsoft, and Google, individual applications and enterprise storage are being deployed on Clouds. However, a serious impediment to its wider deployment is the relative lack of effective data management services. Our experiments, as well as industry reports, have shown that the performance and service-level agreement (SLA) cannot be guaranteed when the data is served over public Clouds. The relatively slow access to persistent data and large variability in cloud storage I/O performance can significantly degrade the performance of data-intensive applications. This paper addresses the issue of I/O performance fluctuation over public cloud platforms and we propose a middleware called CloudMW between the Cloud storage and clients to provide the storage services with better performance and SLA satisfaction. Some technologies, including data virtualization, data chunking, caching, and replication, are integrated into CloudMW to achieve a more stable and predictable performance, and permit flexible sharing of storage among the virtual machines (VMs). Experimental results based on Amazon Web Services (AWS) show that CloudMW is able to improve the stability and help provide better SLAs and data sharing for cloud storage.

**Title :** Dynamic Load Balancing on Web-server Systems

**Author :** Valeria Cardellini

**Year :**1999

## Description :

Popular Web sites can neither rely on a single powerful server nor on independent mirrored- servers to support the ever-increasing request load. Scalability and availability can be provided by distributed Web-server architectures that schedule client requests among the multiple server nodes in a user-transparent way. In this paper we will review the state of the art in load balancing techniques on distributed Web-server systems. We will analyze the eScience and limitations of the various approaches and their trade.

# CHAPTER -3 CLOUD COMPUTING

Cloud storage is a model of data storage in which the digital data is stored in logical pools, the physical storage spans multiple servers (and often locations), and the physical environment is typically owned and managed by a hosting company. Cloud storage is a service model in which data is maintained, managed, backed up remotely and made available to users over a network (typically the Internet).

Cloud storage is a cloud computing model in which data is stored on remote servers accessed from the Internet, or "cloud." It is maintained, operated and managed by a cloud storage service provider on storage servers that are built on virtualization techniques.

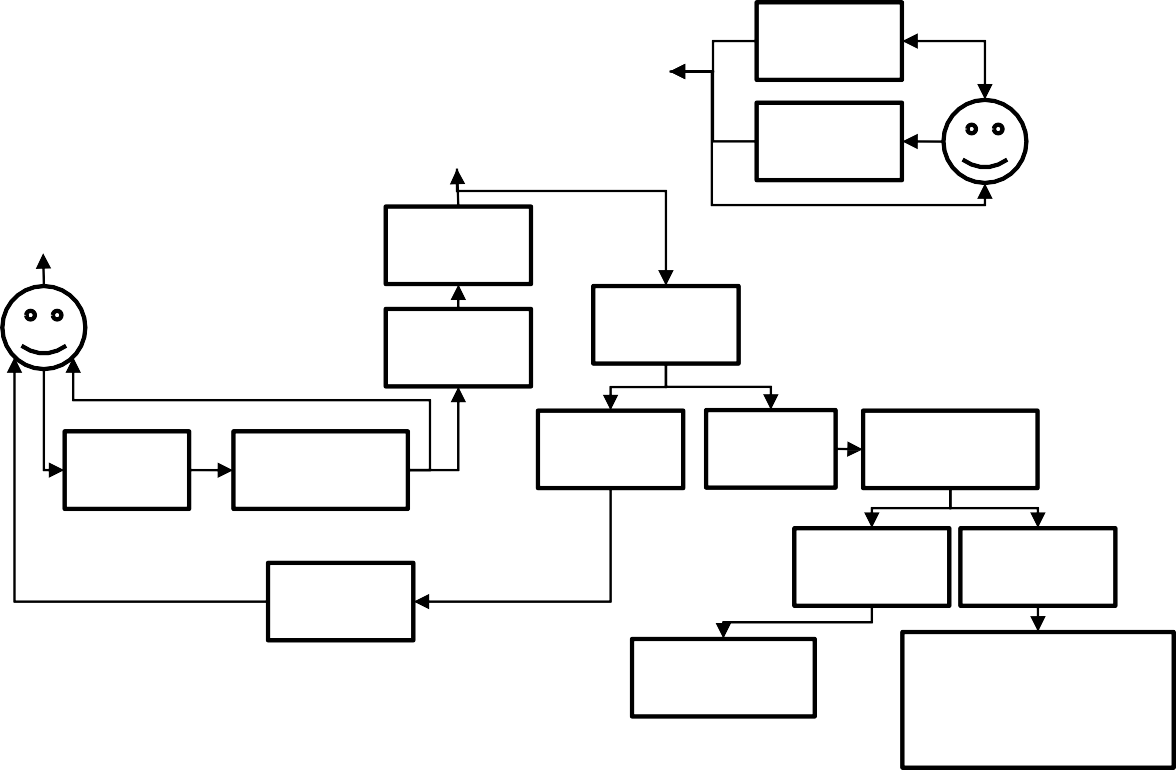
Cloud Files is an affordable, redundant, scalable, and dynamic storage service offering. The core storage system is designed to provide a safe, secure, automatically resizing, and network accessible way to store data. You can store an unlimited quantity of files ranging in size from a few bytes to extremely large.

The maximum storage allowance on BT Cloud depends on how much storage you get free with your BT Broadband package, and the extra storage you purchase: With limited BT Broadband or limited BT Infinity, you get 5GB of free BT Cloud storage. With Unlimited BT Broadband and Unlimited BT Infinity 1, you get 100GB free.

Cloud Storage is a service where data is remotely maintained, managed, and backed up. The service allows the users to store files online, so that they can access them from any location via the Internet. Now, let's look into some of the advantages and disadvantages of Cloud Storage.

# CHAPTER -4 ARCHITECTURE

Architecture diagram shows the relationship between different components of system. This diagram is very important to understand the overall concept of system. Architecture diagram is a diagram of a system, in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks. They are heavily used in the engineering world in hardware design, electronic design, software design, and process flow diagrams.



File

Vali

Aut

Use

Pula

Key

Sel

Che

Val

No

Auth

Yes

Du

No

Simil

Re

Simi

Non

Sen

Rese

Additional

# CHAPTER -5 MODULES

**MODULE DESCRIPTION & MODULE DIAGRAMS**

# AUTHENTICATION:

The process of identifying an individual usually based on a username and password. In security systems, Authentication merely ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual. In authentication module is used to security purpose. Here this module only for user, after registration user enter the username and password. This input is check into the database, whether input is correct or not. If input is correct then allow to next process otherwise consider as a non-authenticated user.

# REGISTER:

In this Module If he is a new user, he needs to enter the required data to register the form and the data will be stored in server for future authenmethtication purpose.

# FILE UPLOADING:

In this scheme user upload the files in the cloud server. Cloud can store multiple files.

Collect several files from the stored in the Cloud Server.

# CHUNKING ALGORITHM:

Chunking Algorithm In data deduplication, the basic idea is to split a file into blocks and applies hash functions to compute hash values. To check data duplication the client sends the hash key list to the server. The hash key for each chunk is used to determine if that chunk exists in the multiple locations by comparing hash keys. If there are same hash keys on another location, we assume that the chunk is duplicated. Therefore, we can prevent duplicated data blocks to be transferred. Generally, the chunking algorithms are divided into two; fixed length chunking and variable length chunking. The fixed length chunking approach achieves very fast data deduplication result but the performance is not good; because boundary shift problem degrades the deduplication performance. On other hand, variable length chunking achieves high degree of performance while causing high computation overhead and longer processing time.

# DE-DUPLICATION

Cloud can store and retrieve file. De-duplication has a removing duplicate file. Its will find out duplicate file. Deduplication The Admin is the data owner who performs deduplication by checking if the contents of two files are the same and stores only one of them. Here the data owner upload, download and update the files. Then the deduplication is performed by applying the RAM Algorithm

# RAPID ASYMMETRIC MAXIMUM ALGORITHM (RAM)

RAM algorithm design with a goal of achieving low computational overhead and byte shift-resistant algorithm, we proposed a boundary version of AE, called RAM. RAM is similar to AE because it also uses two windows: fixed and variable-sized windows. The placement of the windows is, however different from AE. In RAM, the fixed-sized window is located at the beginning of The algorithm works by searching a byte with the maximum value in the fixed- sized window. If the byte next to the fixed sized window has larger value than the one in the fixed-sized window, the byte is used as the maximum-valued byte and the cut-point is found. Otherwise, the algorithm moves to the next byte until it finds the larger byte as illustrated in the pseudo code of RAM . Thus the algorithm’s minimum chunk size is 1, where is the size of the fixed-sized Window. RAM reduces the computation time by searching the byte that is equal or larger than the current maximum value, while Reprocess all the bytes smaller or equal than the maximum-valued bytes. Since the probability that the next byte is smaller than the current maximum value is higher than the probability that the next byte is larger than the current maximum value, RAM enters the first condition less frequently than AE. This lowers RAM’s overhead.

# USER AUTHENTICATION

## REGISTRATION

If you are the new user going to login into the application then you have to register first by providing necessary details. After successful completion of sign up process, the user has to login into the application by providing username and exact password.

# GIN

The user has to provide exact username and password which was provided at the time of registration, if login success means it will take up to main page else it will remain in the login page itself.

# VIEW DETAILS

In this scheme user after the successful login goes to view the no of files in the cloud server. Each service has different set of files. This cloud server has collection of server which uniquely connected with the cloud server.

# FILE DOWNLOADING

In this scheme User uses to download the files in the cloud server. Each service has different set of files. User can collect several file by downloading, which are stored in the Cloud Server. This cloud server has collection of server cluster which uniquely connected with the cloud server

# CHAPTER -6 SOFTWARE DESCRIPTION

**GENERAL**

This chapter is about the software language and the tools used in the development of the project. The platform used here is JAVA. The Primary languages are JAVA, J2EE and J2ME. In this project J2EE is chosen for implementation.

# FEATURES OF JAVA

**THE JAVA FRAMEWORK**

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere".

Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web applicationsThe java framework is a new platform independent that simplifies application development internet. Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

# OBJECTIVES OF JAVA

To see places of Java in Action in our daily life, explore java.com. Why Software Developers Choose Java

Java has been tested, refined, extended, and proven by a dedicated community. And numbering more than 6.5 million developers, it's the largest and most active on the planet. With its versatility, efficiency, and portability, Java has become invaluable to developers by enabling them to:

* Write software on one platform and run it on virtually any other platform
* Create programs to run within a Web browser and Web services
* Combine applications or services using the Java language to create highly customized applications or services
* Write powerful and efficient applications for mobile phones, remote processors, low- cost consumer products, and practically any other device with a digital heartbeat Ways Software Developers Learn Java

Today, many colleges and universities offer courses in programming for the Java platform. In addition, developers can also enhance their Java programming skills by reading Sun's java.sun.com Web site, subscribing to Java technology-focused newsletters, using the Java Tutorial and the New to Java Programming Centre, and signing up for Web, virtual, or instructor-led courses.

# OBJECT ORIENTED

To be an Object-Oriented language, any language must follow at least the four characteristics.

# INHERITANCE

It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding addition a features as needed.

# ENCAPSULATION:

It is the mechanism of combining the information and providing the abstraction.

# POLYMORPHISM:

As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by the functions having the same name based on the signatures of the methods.

# DYNAMIC BINDING:

Sometimes we don't have the knowledge of objects about their specific types while writing our code. It is the way of providing the maximum functionality to a program about the specific type at runtime.

# JAVA SERVER PAGES - AN OVERVIEW

Java Server Pages or JSP for short is Sun's solution for developing dynamic web sites. JSP provide excellent server-side scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy.

JSP pages are efficient, it loads into the web server’s memory on receiving the request very first time and the subsequent calls are served within a very short period of time.

In today's environment most web sites servers’ dynamic pages based on user request. Database is very convenient way to store the data of users and other things. JDBC provide excellent database connectivity in heterogeneous database environment. Using JSP and JDBC its very creasy to develop database driven web application.

Java is known for its characteristic of "write once, run anywhere." JSP pages are platform Java Server Pages

Java Server Pages (JSP) technology is the Java platform technology for delivering dynamic content to web clients in a portable, secure and well-defined way. The JavaServer Pages specification extends the Java Servlet API to provide web application developers with a robust framework for creating dynamic web content on the server using HTML, and XML templates, and Java code, which is secure, fast, and independent of server platforms.

JSP has been built on top of the Server API and utilizes Servlet semantics. JSP has become the preferred request handler and response mechanism. Although JSP technology is going to be a powerful successor to basic Servlets, they have an evolutionary relationship and can be used in a cooperative and complementary manner.

Servlets are powerful and sometimes they are a bit cumbersome when it comes to generating complex HTML. Most servlets contain a little code that handles application logic and a lot more code that handles output formatting. This can make it difficult to separate and reuse portions of the code when a different output format is needed. For these reasons, web application developers turn towards JSP as their preferred servlet environment.

# EVOLUTION OF WEB APPLICATIONS

Over the last few years, web server applications have evolved from static to dynamic applications. This evolution became necessary due to some deficiencies in earlier web site design. For example, to put more of business processes on the web, whether in business-to- consumer (B2C) or business-to-business (B2B) markets, conventional web site design technologies are not enough. The main issues, every developer faces when developing web applications, are:

1. Scalability - a successful site will have more users and as the number of users is increasing firstly, the web applications have to scale correspondingly.
2. Integration of data and business logic - the web is just another way to conduct business, and so it should be able to use the same middle-tier and data-access code.
3. Manageability - web sites just keep getting bigger and we need some viable mechanism to manage the ever-increasing content and its interaction with business systems.
4. Personalization - adding a personal touch to the web page becomes an essential factor to keep our customer coming back again. Knowing their preferences, allowing them to configure the information they view, remembering their past transactions or frequent search keywords are all important in providing feedback and interaction from what is otherwise a fairly one- sided conversation.

Apart from these general needs for a business-oriented web site, the necessity for new technologies to create robust, dynamic and compact server-side web applications has been realized. The main characteristics of today's dynamic web server applications are as follows:

1. Serve HTML and XML, and stream data to the web client
2. Separate presentation, logic and data
3. Interface to databases, other Java applications, CORBA, directory and mail services
4. Make use of application server middleware to provide transactional support.
5. Track client sessions.

# BENEFITS OF JSP

One of the main reasons why the JavaServer Pages technology has evolved into what it is today and it is still evolving is the overwhelming technical need to simplify application design by separating dynamic content from static template display data. Another benefit of utilizing JSP is that it allows to more cleanly separate the roles of web application/HTML

designer from a software developer. The JSP technology is blessed with a number of exciting benefits, which are chronicled as follows:

1. The JSP technology is platform independent, in its dynamic web pages, its web servers, and its underlying server components. That is, JSP pages perform perfectly without any hassle on any platform, run on any web server, and web-enabled application server. The JSP pages can be accessed from any web server.
2. The JSP technology emphasizes the use of reusable components. These components can be combined or manipulated towards developing more purposeful components and page design. This definitely reduces development time apart from the At development time, JSPs are very different from Servlets, however, they are precompiled into Servlets at run time and executed by a JSP engine which is installed on a Web-enabled application server such as BEA WebLogic and IBM WebSphere.

# SERVLETS

Earlier in client- server computing, each application had its own client program and it worked as a user interface and need to be installed on each user's personal computer. Most web applications use HTML/XHTML that are mostly supported by all the browsers and web pages are displayed to the client as static documents.

A web page can merely displays static content and it also lets the user navigate through the content, but a web application provides a more interactive experience.

Any computer running Servlets or JSP needs to have a container. A container is nothing but a piece of software responsible for loading, executing and unloading the Servlets and JSP. While servlets can be used to extend the functionality of any Java- enabled server.

They are mostly used to extend web servers, and are efficient replacement for CGI scripts. CGI was one of the earliest and most prominent server side dynamic content solutions, so before going forward it is very important to know the difference between CGI and the Servlets.

# JAVA SERVLETS

Java Servlet is a generic server extension that means a java class can be loaded dynamically to expand the functionality of a server. Servlets are used with web servers and run inside a Java Virtual Machine (JVM) on the server so these are safe and portable.

Unlike applets they do not require support for java in the web browser. Unlike CGI, servlets don't use multiple processes to handle separate request. Servets can be handled by separate threads within the same process. Servlets are also portable and platform independent.

A web server is the combination of computer and the program installed on it. Web server interacts with the client through a web browser. It delivers the web pages to the client and to an application by using the web browser and he HTTP protocols respectively.

The define the web server as the package of large number of programs installed on a computer connected to Internet or intranet for downloading the requested files using File Transfer Protocol, serving e-mail and building and publishing web pages. A web server works on a client server model.

# COLLECTIONS:

The Java Collections API's provide Java developers with a set of classes and interfaces that makes it easier to handle collections of objects. In a sense Collection's works a bit like arrays, except their size can change dynamically, and they have more advanced behaviour than arrays. In this project we are using Array List for collecting the user input and saving values.

# THREAD:

In this project threading concept is very important. A thread is a sequential path of code execution within a program. And each thread has its own local variables, program counter and lifetime. Like creation of a single thread, we can also create more than one thread (multithreads) in a program using class Thread or implementing interface Run able to make our project efficient and dynamic. In our project we are using request process with the help of multi- threading concepts.

# SWINGS:

Swing, which is an extension library to the AWT, includes new and improved components that enhance the look and functionality of GUIs. Swing can be used to build Standalone swing guy apps as well as Servers and Applets. It employs model/view design architecture. Swing is more portable and more flexible than AWT.

# MYSQL

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home- brewed lexical analyser. The MySQL server software itself and the client libraries use dual- licensing distribution. Support can be obtained from the official manual.[24] Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third-party organizations exist to provide support and services. MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case" and that the "developer interfaces are there, and the documentation. It has also been tested to be a "fast, stable and true multi-user, multi-threaded SQL database server".

# CHAPTER – 7 FEATURES

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL

* + - * A broad subset of ANSI SQL 99, as well as extensions Cross-platform support
      * Stored procedures, using a procedural language that closely adheres to SQL/PSM[62]
      * Triggers Cursors
      * Updatable views
      * Online DDL when using the InnoDB Storage Engine.
      * Information schema
      * Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.[63]
      * A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
      * X/Open XA distributed transaction processing (DTP) support; two phase commits as part of this, using the default InnoDB storage engine
      * Transactions with save points when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
      * ACID compliance when using InnoDB and NDB Cluster Storage Engines [64]
      * SSL support
      * Query caching
      * Sub-SELECTs (i.e. nested SELECTs)

# BACKUP SOFTWARE

MySQL dump is a logical backup tool included with both community and enterprise editions of MySQL. It supports backing up from all storage engines. MySQL Enterprise

Backup is a hot backup utility included as part of the MySQL Enterprise subscription from Oracle, offering native InnoDB hot backup, as well as backup for other storage engines. XtraBackup is an open-source MySQL hot backup software program. Features include hot, non-locking backups for InnoDB storage, incremental backups, streaming, parallel- compressed backups, throttling based on the number of I/O operations per second, etc.

# HIGH AVAILABILITY SOFTWARE

MySQL Fabric is an integrated system for managing a collection of MySQL servers, and a framework on top of which high availability and database sharding is built. MySQL Fabric is open-source, and supports procedure execution in the presence of failure, providing an execution model usually called resilient execution. MySQL client libraries are extended so they are hiding the complexities of handling failover in the event of a server failure, as well as correctly dispatching transactions to the shards.

# OUD DEPLOYMENT

MySQL can also be run on cloud computing platforms such as Amazon EC2. Some common deployment models for MySQL on the cloud are:

# VIRTUAL MACHINE IMAGE

In this implementation, cloud users can upload a machine image of their own with MySQL installed, or use a ready-made machine image with an optimized installation of MySQL on it, such as the one provided by Amazon EC2.

# MYSQL AS A SERVICE

Some cloud platforms offer MySQL "as a service". In this configuration, application owners do not have to install and maintain the MySQL database on their own. Instead, the database service provider takes responsibility for installing and maintaining the database, and application owners pay according to their usage.

# USER INTERFACES:

Graphical user interfaces

A graphical user interface (GUI) is a type of interface that allows users to interact with electronic devices or programs through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation. GUIs are easier to learn than command-line interfaces (CLIs), which require commands to be typed on the keyboard.

Third-party proprietary and free graphical administration applications (or "front ends") are available that integrate with MySQL and enable users to work with database structure and data visually.

# MYSQL WORKBENCH

MySQL Workbench is the official integrated environment for MySQL. It was developed by MySQL AB, and enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, MySQL GUI Tools. Similar to other third-party packages, but still considered the authoritative MySQL front end, MySQL Workbench lets users manage database design & modelling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).

# ADMINER

Adminer (formerly known as phpMinAdmin) is a free MySQL front end for managing content in MySQL databases (since version 2, it also works on PostgreSQL, MS SQL, SQLite and Oracle SQL databases). Adminer is distributed under the Apache license (or GPL v2) in the form of a single PHP file (around 300 KiB in size), and is capable of managing multiple databases.

# DATABASE WORKBENCH

Database Workbench is a software application for development and administration of multiple relational databases using SQL, with interoperationality between different database systems,

Because Databases Workbench supports multiple database systems, it can provide software developers with the same interface and development environment for these otherwise different database systems and also includes cross database tools.

# DEBIT

Debit is a database editor, which can connect to an Oracle, DB2, MySQL and any database that provides a JDBC driver. It runs on Windows, Linux and Solaris. DBEdit is free and open source software and distributed under the GNU General Public License. The source code is hosted on Source Forge.

# SQLBUDDY

SQLBuddy is an open-source web-based application written in PHP intended to handle the administration of MySQL and SQLite with the use of a Web browser. The project places an emphasis on ease of installation and a simple user interface.

# SQLYOG

SQLyog is a GUI tool available in free as well as paid versions. Data manipulations (e.g., insert, update, and delete) may be done from a spreadsheet-like interface. Its editor has syntax highlighting and various automatic formatting options. Both raw table data and a result set from a query can be manipulated. Its data search feature uses Google-like search syntax and translates to SQL transparently for the user. It has a backup tool for performing unattended backups. Backups may be compressed and optionally stored as a file-per-table as well as identified with a timestamp.

# COMMAND-LINE INTERFACES

A command-line interface is a means of interacting with a computer program where the user issues commands to the program by typing in successive lines of text (command lines). MySQL ships with many command line tools, from which the main interface is the mysql client.

MySQL Utilities is a set of utilities designed to perform common maintenance and administrative tasks. Originally included as part of the MySQL Workbench, the utilities are a stand-alone download available from Oracle.

# APPLICATION PROGRAMMING INTERFACES:

Many programming languages with language-specific APIs include libraries for accessing MySQL databases. These include MySQL Connector/Net for integration with Microsoft's Visual Studio (languages such as C# and VB are most commonly used) and the JDBC driver for Java. In addition, an ODBC interface called MySQL Connector/ODBC allows additional programming languages that support the ODBC interface to communicate with a MySQL database, such as ASP or ColdFusion. The HTSQL – URL-based query method also ships with a MySQL adapter, allowing direct interaction between a MySQL database and any web client via structured URLs.

# RAM CODING

package CDC;

import java.io.BufferedInputStream; import java.io.File;

import java.io.FileInputStream; import java.io.IOException; import java.io.InputStream; import java.util.Hashtable;

# CHAPTER-8 CODING

import Bean.Login\_Bean;

import Implementations.User\_Implementations; import Interfaces.User;

public class RAM {

public static final int hconst = 69069; // good hash multiplier for MOD 2^32 public int mult = 1; // this will hold the p^n value

int[] buffer; // circular buffer - reading from file stream int buffptr = 0;

int segment = 0; // Once a chunk found, reset to 0 InputStream is;

int window;

public Hashtable<String, String> indexTable; public FileList list;

public Checksum sum; String path,username;

public RAM(String directory,String email) { indexTable = new Hashtable<String, String>(); sum = new Checksum();

// list = new FileList(Config.DIRECTORY); list = new FileList(directory+email); window = 5; // initial window size

path=directory; username=email;

}

// Initialize some value before start find chunks for each file public void initialize() {

mult = 1;

buffptr = 0;

}

public void setAll(File[] fileList) { for (File file : fileList) {

if (file.isFile() && !file.isHidden()) { initialize(); displayChunks(file);

}

}

}

public void displayChunks(File filelocation) { int mask = 1 << 13;

mask--; // 13 bit of '1's

File f = filelocation;

FileInputStream fs = null; // For sliding window

FileInputStream fsChunk = null; // For chunking the input stream BufferedInputStream bis = null;

try {

fs = new FileInputStream(f); fsChunk = new FileInputStream(f); bis = new BufferedInputStream(fs);

// BufferedInputStream is faster to read byte-by-byte from this.is = bis;

window

long length = bis.available(); long curr = length;

// get the initial 1k hash window // int hash = inithash(window);

curr -= bis.available(); // move the curr to next byte of the initial hash

byte[] chunk = null; String hashvalue = null; boolean firstChunk = true;

int count = 0; // count chunks

int duplicate = 0; // count duplicate long chk\_size=1024;

while (curr < length) {

if ((hash & mask) == 0) {

// window found - hash it, if (firstChunk == true) {

chunk = new byte[(int) chk\_size]; firstChunk = false;

} else {

chunk = new byte[segment];

}

if (fsChunk.read(chunk) != -1) {

// perform the hash on the chunk

hashvalue =

sum.naechunking(chunk,length,window);

//System.out.println(hashvalue);

// If not exist then save if(hashvalue!=null)

{

if (!indexTable.containsKey(hashvalue)) { indexTable.put(hashvalue, f.getName());

} else {

// found duplicated chunks

duplicate++;

}

}

}

segment = 0; count++;

}

// next window's hash // hash = nexthash(hash); curr++;

segment++;

}

Login\_Bean reg=new Login\_Bean();

//docreg.setid(id); reg.setram\_chks(count); if (duplicate != 0) { reg.setra\_dup(duplicate);

}

else

{

reg.setra\_dup(duplicate);

}

reg.setfilename(f.getName()); reg.setemail(username);

System.out.println(count + " chunks generated for: " + f.getName()); if (duplicate != 0) {

System.out.println(duplicate + " duplicated chunks in: "

+ f.getName());

}

User i=new User\_Implementations(); int status=i.fileupdateRam(reg);

bis.close();

} catch (Exception e) {

e.printStackTrace();

} finally {

// clean up

if (fs != null) {

try {

is.close();

fs.close();

} catch (IOException e) {

// TODO Auto-generated catch block e.printStackTrace();

}

}

}

}

public int nexthash(int prevhash) throws IOException { int c = is.read(); // next byte from stream

prevhash -= mult \* buffer[buffptr]; // remove the last value prevhash \*= hconst; // multiply the whole chain with prime prevhash += c; // add the new value

buffer[buffptr] = c; // circular buffer, 1st pos == lastpos buffptr++;

buffptr = buffptr % buffer.length;

return prevhash;

}

public int inithash(int length) throws IOException { buffer = new int[length]; // create circular buffer

int hash = 0;

// calculate the hash sum of p^n \* a[x] for (int i = 0; i < length; i++) {

int c = is.read();

if (c == -1) // file is shorter than the required window size break;

// store byte so we can remove it from the hash later buffer[buffptr] = c;

buffptr++;

buffptr = buffptr % buffer.length;

hash \*= hconst; // multiply the current hash with constant hash += c; // add byte to hash

if (i > 0) // calculate the large p^n value for later usage mult \*= hconst;

}

return hash;

}

}

# RABIN CODING

package CDC;

import java.io.BufferedInputStream; import java.io.File;

import java.io.FileInputStream; import java.io.IOException; import java.io.InputStream; import java.util.Hashtable;

import Bean.Login\_Bean;

import Implementations.User\_Implementations; import Interfaces.User;

public class Rabin {

public static final int hconst = 69069; // good hash multiplier for MOD 2^32 public int mult = 1; // this will hold the p^n value

int[] buffer; // circular buffer - reading from file stream int buffptr = 0;

int segment = 0; // Once a chunk found, reset to 0 InputStream is;

int window;

public Hashtable<String, String> indexTable; public FileList list;

public Checksum sum; String path=null; String username=null;

public Rabin(String directory,String email) { indexTable = new Hashtable<String, String>(); sum = new Checksum();

// list = new FileList(Config.DIRECTORY); list = new FileList(directory+email); window = 1024; // initial window size path=directory;

username=email;

}

// Initialize some value before start find chunks for each file public void initialize() {

mult = 1;

buffptr = 0;

}

public void setAll(File[] fileList) { for (File file : fileList) {

if (file.isFile() && !file.isHidden()) { initialize(); displayChunks(file);

}

}

}

public void displayChunks(File filelocation) { int mask = 1 << 13;

mask--; // 13 bit of '1's

File f = filelocation;

FileInputStream fs = null; // For sliding window

FileInputStream fsChunk = null; // For chunking the input stream BufferedInputStream bis = null;

try {

fs = new FileInputStream(f); fsChunk = new FileInputStream(f); bis = new BufferedInputStream(fs);

// BufferedInputStream is faster to read byte-by-byte from this.is = bis;

window

long length = bis.available(); long curr = length;

// get the initial 1k hash window // int hash = inithash(window);

curr -= bis.available(); // move the curr to next byte of the initial hash

byte[] chunk = null; String hashvalue = null; boolean firstChunk = true;

int count = 0; // count chunks

int duplicate = 0; // count duplicate

while (curr < length) {

if ((hash & mask) == 0) {

// window found - hash it, if (firstChunk == true) {

chunk = new byte[(int) curr]; firstChunk = false;

} else {

chunk = new byte[segment];

}

if (fsChunk.read(chunk) != -1) {

// perform the hash on the chunk hashvalue = sum.chunking(chunk);

// not exist then save

if (!indexTable.containsKey(hashvalue)) { indexTable.put(hashvalue, f.getName());

} else {

// found duplicated chunks duplicate++;

}

}

segment = 0; count++;

}

// next window's hash // hash = nexthash(hash); curr++;

segment++;

}

/////////////////////////////////////////////

Login\_Bean reg=new Login\_Bean();

//docreg.setid(id); reg.setrab\_chks(count); if (duplicate != 0) {

reg.setrab\_dup(duplicate);

}

else

{

reg.setrab\_dup(duplicate);

}

reg.setfilename(f.getName()); reg.setemail(username);

////////////////////////////////////////////

System.out.println(count + " chunks generated for: " + f.getName()); if (duplicate != 0) {

System.out.println(duplicate + " duplicated chunks in: "

+ f.getName());

}

User i=new User\_Implementations(); int status=i.fileupdateRab(reg);

bis.close();

} catch (Exception e) {

e.printStackTrace();

} finally {

// clean up

if (fs != null) {

try {

is.close();

fs.close();

} catch (IOException e) {

// TODO Auto-generated catch block e.printStackTrace();

}

}

}

}

public int nexthash(int prevhash) throws IOException { int c = is.read(); // next byte from stream

prevhash -= mult \* buffer[buffptr]; // remove the last value prevhash \*= hconst; // multiply the whole chain with prime prevhash += c; // add the new value

buffer[buffptr] = c; // circular buffer, 1st pos == lastpos buffptr++;

buffptr = buffptr % buffer.length;

return prevhash;

}

public int inithash(int length) throws IOException { buffer = new int[length]; // create circular buffer

int hash = 0;

// calculate the hash sum of p^n \* a[x] for (int i = 0; i < length; i++) {

int c = is.read();

if (c == -1) // file is shorter than the required window size break;

// store byte so we can remove it from the hash later buffer[buffptr] = c;

buffptr++;

buffptr = buffptr % buffer.length;

hash \*= hconst; // multiply the current hash with constant hash += c; // add byte to hash

if (i > 0) // calculate the large p^n value for later usage mult \*= hconst;

}

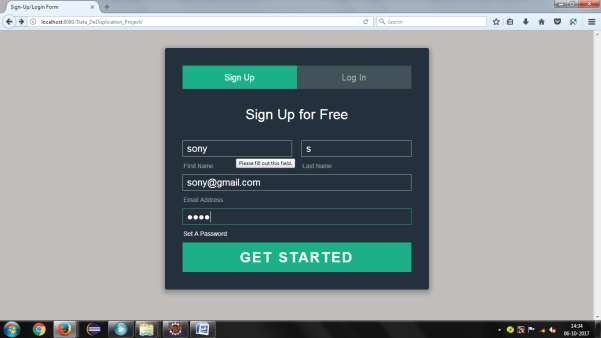
return hash;

}

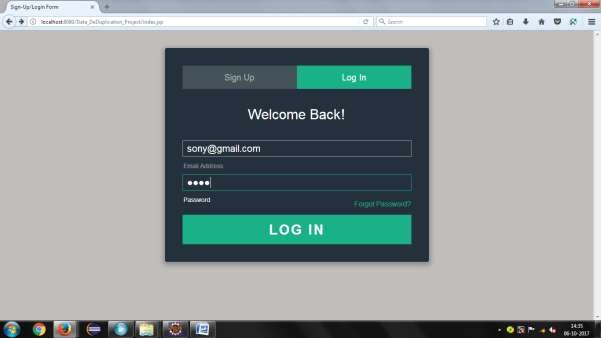
}

# CHAPTER-9 SCREEN SHOT

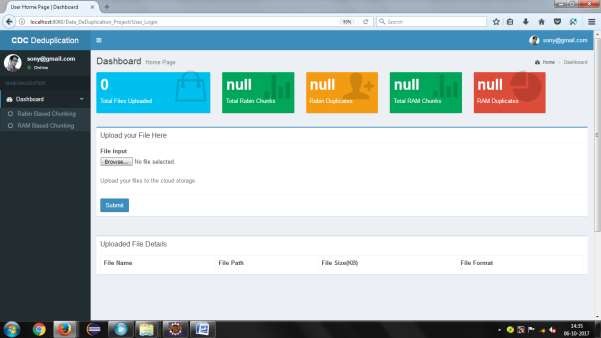
**SIGNUP**



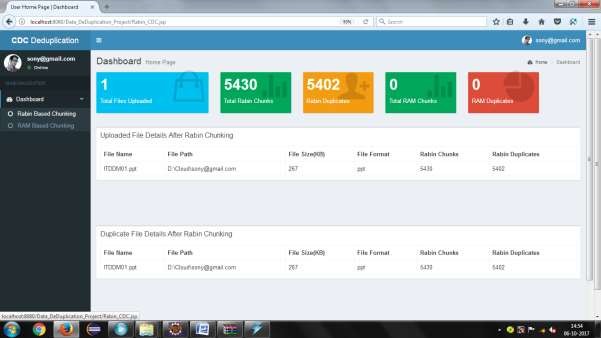
# LOGIN



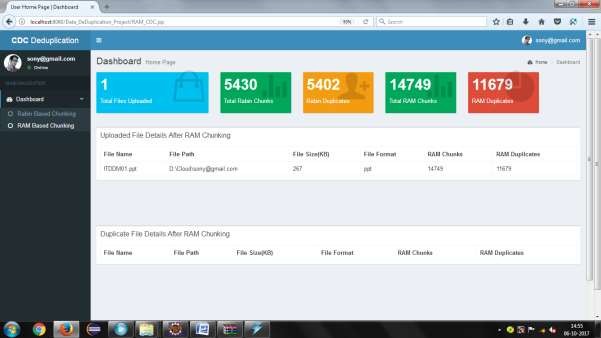
**HOME**



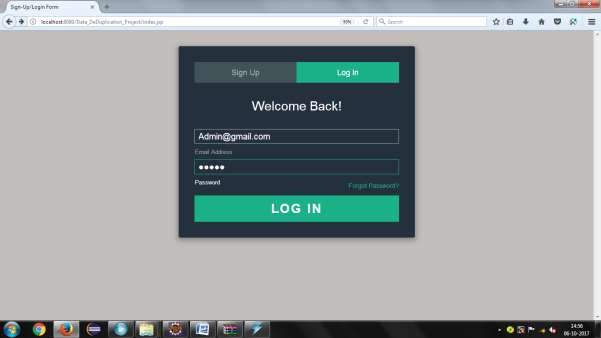
# RABIN BASED CHUNKING



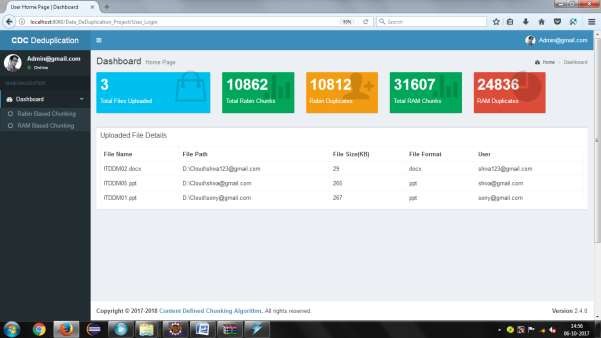
**RAM BASED CHUNKING**



# ADMIN LOGIN



**ADMIN PAGE**



# CHAPTER-10 CONCLUSION

In this paper, we have discussed the importance of content defined chunking for multiple applications and why it is better than fix-sized chunking. We proposed a new chunking algorithm, called Rapid Asymmetric Maximum (RAM) based on asymmetric chunking algorithm. We analysed and compared RAM with other chunking algorithms. Our results show that RAM offers lower computational overhead compared to other CDC algorithms. The main advantage of RAM is its low computation overhead which allows high chunking throughput. The high chunking throughput comes at the cost of higher chunk variance. The higher chunk variance produced by RAM is negligible compared to the performance gain over other chunking schemes based on local maximum chunking. In some cases, RAM offers 26% to 40% higher byte saved per second compared to the other chunking algorithms.