A MAJOR PROJECT ON

ENABLING CLOUD STORAGE AUDTING WITH VERIFIABLE OUTSOURCING OF KEY UPDATES

A dissertation submitted in partial fulfilment of the requirements for awarding the degree of

Bachelor of Technology

in

INFORMATION TECHNOLOGY

Submitted by

T Akhil – (18B81A1262)

A Charan – (18B81A1268)

D V S S Mihir – (18B81A12B2)

Under the esteemed guidance of

Dr.B.Vikranth

Professor, IT Department

CVR College of Engineering

DEPARTMENT OF INFORMATION TECHNOLOGY

Vastunagar, Mangalpally (V), Ibrahimpatnam (M), R.R. District, PIN-501 510

2021-2022

Cherabuddi Education Society's
CVR COLLEGE OF ENGINEERING
(An Autonomous Institution)
ACCREDITED BY NATIONAL BOARD OF ACCREDITATION, AICTE
(Approved by AICTE & Govt. of Telangana and Affiliated to JNT University)
Vastunagar, Mangalpalli (V), Ibrahimpatan (M), R.R. District, PIN - 501 510
Web : http://cvr.ac.in, email : info@cvr.ac.in
Ph : 08414 - 252222, 252369, Office Telefax : 252396, Principal : 252396 (O)

CERTIFICATE

This is to certify that the Project Report entitled **"Enabling cloud storage auditing of verifiable outsourcing of key updates"** is a bonafide work done and submitted by **T .Akhil (18B81A12B2)**, **A Charan (18B81A1268) and D V S S Mihir (18B81A12B2)** during the academic year 2021-2022, in partial fulfilment of requirement for the award of Bachelor of Technology degree in Information Technology from Jawaharlal Nehru Technological University Hyderabad, is a bonafide record of work carried out by them under my guidance and supervision.

Certified further that to my best of the knowledge, the work in this dissertation has not been submitted to any other institution for the award of any degree or diploma.

Project Guide

Dr.B.Vikranth
Professor, IT Department
Department

Head of Department

Dr. Bipin Bihari Jayasingh
Professor,IT

Project Coordinator

Dr. J. Sengathir
Associate Professor,
Information Technology

External Examiner

DECLARATION

We hereby declare that the project report entitled "Enabling cloud storage auditing of verifiable outsourcing of key updates" is an original work done and submitted to IT Department, CVR College of Engineering, affiliated to Jawaharlal Nehru Technological University, Hyderabad in partial fulfilment for the requirement of the award of Bachelor of Technology in Information Technology and it is a record of bonafide project work carried out by us under the guidance of Dr.B.Vikranth, Professor, Department of Information Technology. We further declare that the work reported in this project has not been submitted, either in part or
in full, for the award of any other degree or diploma in this institute or any other Institute or University.

_____

Signature of Student
T.Akhil
(18B81A12B2)

_____

Signature of Student
A.Charan
(18B81A1268)

_____

Signature of Student
D V S S Mihir
(18B81A12B2)

ACKNOWLEDGEMENT

The satisfaction of completing this project would be incomplete without mentioning our gratitude towards all the people who have supported us. Constant guidance and encouragement have been instrumental in the completion of this project.

We offer our sincere gratitude to our internal guide, Dr.B.Vikranth, Professor of IT Department, for his immense support, timely co-operation, and valuable advice throughout the course of our project work

.

We would like to thank the Head of Department, Professor Dr. Bipin Bihari Jayasingh, for his meticulous care and cooperation throughout the project work.We are thankful to Dr. J. Sengathir, Project Coordinator, Associate Professor, IT Department, CVR College of Engineering for his supportive guidelines and for having provided the necessary help for carrying forward this project without any obstacles and hindrances. We also thank the Project Review Committee Members for their valuable suggestions.

ABSTRACT

Key-exposure resistance has always been an important issue for in-depth cyber defense in many security applications. Recently, how to deal with the key exposure problem in the settings of cloud storage auditing has been proposed and studied. In this project, we focus on how to make the key updates as transparent as possible for the client and propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates.

In this paradigm, key updates can be safely out sourced to some authorized party, and thus the key-update burden on the client will be kept minimal. In particular, we leverage the third party auditor (TPA) in many existing public auditing designs, let it play the role of authorized party in our case, and make it in charge of both the storage auditing and the secure key updates for key-exposure resistance. In our design, TPA only needs to hold an encrypted version of the client's secret key while doing all these burden some tasks on behalf of the client. The client only needs to download the encrypted secret key from the TPA.

TABLE OF CONTENTS

Chapter-1

INTRODUCTION

1. Introduction

Key-exposure resistance has always been an important issue for in-depth cyber defense in many security applications. Recently, how to deal with the key exposure problem in the settings of cloud storage auditing has been proposed and studied. In this project, we focus on how to make the key updates as transparent as possible for the client and propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates. In this paradigm, key updates can be safely out sourced to some authorized party, and thus the key-update burden on the client will be kept minimal. In particular, we leverage the third party auditor (TPA) in many existing public auditing designs, let it play the role of authorized party in our case, and make it in charge of both the storage auditing and the secure key updates for key-exposure resistance. In our design, TPA only needs to hold an encrypted version of the client's secret key while doing all these burden some tasks on behalf of the client. The client only needs to download the encrypted secret key from the TPA.

1.1 Motivation

Our design also equips the client with capability to further verify the validity of the encrypted secret keys provided by the TPA. All these salient features are carefully designed to make the whole auditing procedure with key exposure resistance as transparent as possible for the client. We formalize the definition and the security model of this paradigm. The security proof and the performance simulation show that our detailed design instantiations are secure and efficient.

1.2 Problem definition

The client should be able to verify the validity of the encrypted secret key after the client retrieves it from the authorized party. The goal of this paper is to design a cloud storage auditing protocol that can satisfy above requirements to achieve the outsourcing of key updates.

- **Objective of project**

In this Project, we consider achieving goal by outsourcing key updates. However, it needs to satisfy several new requirements to achieve this goal.
The real client's secret keys for cloud storage auditing should not be known by the authorized party who performs outsourcing computation for key updates.

Limitations of project

The authorized party performing outsourcing computation only knows the encrypted secret keys; key updates should be completed under the encrypted state.

1.3 Organization of Documentation

The proposed dissertation consists of seven Chapters including Introduction and Conclusions. Chapter 1 motivation, problem definition, objective and limitation of the proposed system. Chapter 2 emphasizes on detailed literature survey. Chapter 3 Describes about the analysis, software requirement specification, software and hardware requirements, algorithms. Chapter 4 described the Total Design of the Project using UML Diagrams and Chapter 5 describes the implementation details of the project. Testing and validation and the Screen Shots/ Reports is described in Chapter 6.Chapter 7 describes the conclusion and future work of the project.

1.4 Literature Survey

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration r taken into account for developing the proposed system.

a) Secure outsourcing of scientific computations

AUTHORS: M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. E. Spafford **Year**:2000

We investigate the outsourcing of numerical and scientific computations using the following framework: A customer who needs computations done but lacks the computational resources (computing power, appropriate software, or programming expertise) to do these locally, would like to use an external agent to perform these computations. This currently arises in many practical situations, including the financial services and petroleum services industries. The outsourcing is secure if it is done without revealing to the external agent either the actual data or the actual answer to the computations. Th

e general idea is for the customer to do some carefully designed local preprocessing (disguising) of the problem and/or data before sending it to the agent, and some local postprocessing of the answer returned to extract the true answer. The disguise process should be as lightweight as possible, e.g., take time proportional to the size of the input and answer. The disguise preprocessing that that the customer performs locally to "hide" the real computation can change the numerical properties of the computational performance. We present a framework for disguising scientific computations and discuss their costs, numerical properties, and levels of security. These disguise techniques can be embedded in a very high level, easy-to-use system (problem solving environment) that hides their complexity.

b) Private and cheating-free outsourcing of algebraic computations

AUTHORS: D. Benjamin and M. J. Atallah **Year:**2002

We give protocols for the secure and private outsourcing of linear algebra computations, that enable a client to securely outsource expensive algebraic computations (like the multiplication of huge matrices) to two remote servers, such that the servers learn nothing about the customer's private input or the result of the computation,and any attempted corruption of the answer by the servers is detected with high probability. The computational work done locally by the client is linear in the size of its input and does not require the client to carry out locally any expensive encryptions of such input.The computational burden on the servers is proportional to the time complexity of the current practically used algorithms for solving the algebraic problem (e.g., proportional to n^3 for multiplying two n x n matrices). If the servers were to collude against the client,then they would only find out the client' private inputs, but they would not be able to corrupt the answer without detection by the client.

c) Secure and practical outsourcing of linear programming in cloud computing

AUTHORS: C. Wang, K. Ren, and J. Wang **Year**:2002

Cloud computing enables customers with limited computational resources to outsource large-scale computational tasks to the cloud, where massive computational power can be easily utilized in a pay-per-use manner. However, security is the major concern that prevents the wide adoption of computation outsourcing in the cloud, especially when end-user's confidential data are processed and produced during the computation. Thus, secure outsourcing mechanisms are in great need to not only protect sensitive information by enabling computations with encrypted data, but also protect customers from malicious behaviors by validating the computation result. Such a mechanism of general secure computation outsourcing was recently shown to be feasible in theory, but to design mechanisms that are practically efficient remains a very challenging problem. Focusing on engineering computing and optimization tasks, this paper investigates secure outsourcing of widely applicable linear programming (LP) computations. In order to achieve practical efficiency, our mechanism design explicitly decomposes the LP computation outsourcing into public LP solvers running on the cloud and private LP parameters owned by the customer. The resulting flexibility allows us to explore appropriate security/efficiency tradeoff via higher-level abstraction of LP computations than the general circuit representation. In particular, by formulating private data owned by the customer for LP problem as a set of matrices and vectors, we are able to develop a set of efficient privacy-preserving problem transformation techniques, which allow customers to transform original LP problem into some random one while protecting sensitive input/output information. To validate the computation result, we further explore the fundamental duality theorem of LP computation and derive the necessary and sufficient conditions that correct result must satisfy. Such result verification mechanism is extremely efficient and incurs close-to-zero additional cost on both cloud server and customers. Extensive security analysis and experiment results show the immediate practicability of our mechanism design.

D) New algorithms for secure outsourcing of modular exponentiations

AUTHORS: X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou **Year**:2014

With the rapid development of cloud services, the techniques for securely outsourcing the prohibitively expensive computations to untrusted servers are getting more and more attention in the scientific community. Exponentiations modulo a large prime have been considered the most expensive operations in discrete-logarithm-based cryptographic protocols, and they may be burdensome for the resource-limited devices such as RFID tags or smartcards. Therefore, it

is important to present an efficient method to securely outsource such operations to (untrusted) cloud servers. In this paper, we propose a new secure outsourcing algorithm for (variable-exponent, variable-base) exponentiation modulo a prime in the two untrusted program model. Compared with the state-of-the-art algorithm, the proposed algorithm is superior in both efficiency and checkability. Based on this algorithm, we show how to achieve outsource-secure Cramer-Shoup encryptions and Schnorr signatures. We then propose the first efficient outsource-secure algorithm for simultaneous modular exponentiations. Finally, we provide the experimental evaluation that demonstrates the efficiency and effectiveness of the proposed outsourcing algorithms and schemes.

5) Provable data possession at untrusted stores

AUTHORS: G. Ateniese *et al* ***Year:****2007*

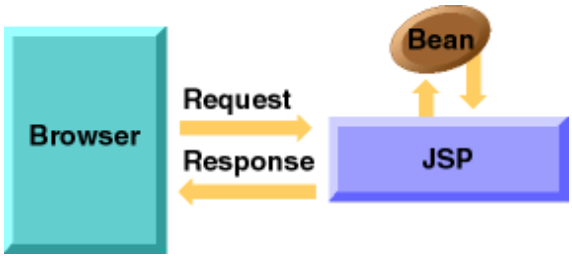
We introduce a model for *provable data possession* (PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage system.

We present two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

Access Models

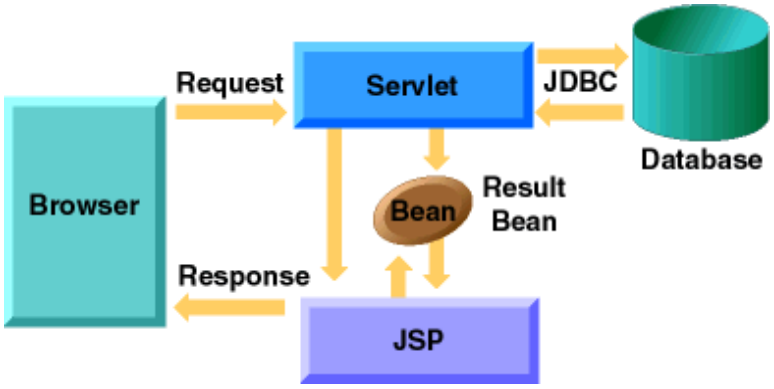
A Java Server Pages file may be accessed in at least two different ways:

- A client request comes directly into a Java Server Page.



In this scenario, suppose the page accesses reusable JavaBean components that perform particular well-defined computations like accessing a database. The result of the Bean's computations, called *result sets* are stored wihtin the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.

- A request comes through a servlet.



The servlet generates the dynamic content. To handle the response to the client, the servlet creates a Bean and stores the dynamic content (sometimes called the *result set*) in the Bean. The servlet then invokes a Java Server Page that will present the content along with the Bean containing the generated from the servlet.

There are two APIs to support this model of request processing using Java Server Pages. One API facilitates passing context between the invoking servlet and the Java Server Page. The other API lets the invoking servlet specify which Java Server Page to use.

In both of the above cases, the page could also contain any valid Java code. The Java Server Pages architecture encourages separation of content from presentation—it does not mandate it.

JDBC

In an effort to set an independent database standard API for Java, Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is

achieved through the use of "plug-in" database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after. The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

HTML

HTML, an initial of Hypertext Markup Language, is the predominant markup language for web pages. It provides a means to describe the structure of text-based information in a document — by denoting certain text as headings, paragraphs, lists, and so on — and to supplement that text with interactive forms, embedded images, and other objects. HTML is written in the form of labels (known as tags), surrounded by angle brackets. HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code which can affect the behavior of web browsers and other HTML processors.

HTML is also often used to refer to content of the MIME type text/html or even more broadly as a generic term for HTML whether in its XML-descended form (such as XHTML 1.0 and later) or its form descended directly from SGML

Hyper Text Markup Language

Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

Basic HTML Tags:

<! -- -->	specifies comments
<A>…….</A>	Creates hypertext links
<B>…….</B>	Formats text as bold
<BIG>…….</BIG>	Formats text in large font.
<BODY>…</BODY>	Contains all tags and text in the HTML document
<CENTER>...</CENTER>	Creates text
<DD>…</DD>	Definition of a term
<DL>...</DL>	Creates definition list
<FONT>…</FONT>	Formats text with a particular font
<FORM>...</FORM>	Encloses a fill-out form
<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>…</H#>	Creates headings of different levels(1 – 6)
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HR>...</HR>	Creates a horizontal rule
<HTML>…</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>…</SCRIPT>	Contains client-side or server-side script
<TABLE>…</TABLE>	Creates a table
<TD>…</TD>	Indicates table data in a table

<TR>…</TR>	Designates a table row
<TH>…</TH>	Creates a heading in a table

Attributes

The attributes of an element are name-value pairs, separated by "=", and written within the start label of an element, after the element's name. The value should be enclosed in single or double quotes, although values consisting of certain characters can be left unquoted in HTML (but not XHTML).Leaving attribute values unquoted is considered unsafe.

Most elements take any of several common attributes: id, class, style and title. Most also take language-related attributes: lang and dir.

The id attribute provides a document-wide unique identifier for an element. This can be used by stylesheets to provide presentational properties, by browsers to focus attention on the specific element or by scripts to alter the contents or presentation of an element. The class attribute provides a way of classifying similar elements for presentation purposes. For example, an HTML document (or a set of documents) may use the designation class="notation" to indicate that all elements with this class value are all subordinate to the main text of the document (or documents). Such notation classes of elements might be gathered together and presented as footnotes on a page, rather than appearing in the place where they appear in the source HTML.

An author may use the style non-attributal codes presentational properties to a particular element. It is considered better practice to use an element's son- id page and select the element with a style sheet, though sometimes this can be too cumbersome for a simple ad hoc application of styled properties. The title is used to attach sub textual explanation to an element. In most browsers this title attribute is displayed as what is often referred to as a tooltip. The generic inline span element can be used to demonstrate these various non-attributes.

The preceding displays as HTML (pointing the cursor at the abbreviation should display the title text in most browsers).

Advantages

- A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- HTML is platform independent.
- HTML tags are not case-sensitive.

JavaScript

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java. JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then update the browser's display accordingly

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

<SCRIPTS>.. </SCRIPT>.

<SCRIPT LANGUAGE = "JavaScript">

JavaScript statements

</SCRIPT>

Here are a few things we can do with JavaScript:

- Validate the contents of a form and make calculations.
- Add scrolling or changing messages to the Browser's status line.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

JavaScript Vs Java

JavaScript and Java are entirely different languages. A few of the most glaring differences are:

- Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.

- While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

Advantages

- JavaScript can be used for Sever-side and Client-side scripting.
- It is more flexible than VBScript.
- JavaScript is the default scripting languages at Client-side since all the browsers supports it.

1.5. Existing System

While all existing protocols focus on the faults of the cloud, they have overlooked the possible weak sense of security and/or low security settings at the client.

Unfortunately, previous auditing protocols did not consider this critical issue, and any exposure of the client's secret auditing key would make most of the existing auditing protocols unable to work correctly.

We focus on how to reduce the damage of the client's key exposure in cloud storage auditing.

1.6. Disadvantages of Existing system

Existing system don't like auditing protocol with verifiable outsourcing of key updates.
Third party has the access to see client's secret key without encryption.
No verification system available for client's for to check validity of the encrypted secret keys when downloading them from the TPA.

1.7. Proposed System

We propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates. In this new paradigm, key-update operations are not performed by the client, but by an authorized party. The authorized party holds an encrypted secret key of the client for cloud

storage auditing and updates it under the encrypted state in each time period. The client downloads the encrypted secret key from the authorized party and decrypts it only when he would like to upload new files to cloud. In addition, the client can verify the validity of the encrypted secret key.

We design the first cloud storage auditing protocol with verifiable outsourcing of key updates. In our design, the third partyauditor (TPA) plays the role of the authorized party who is in charge of key updates.

ADVANTAGES:

The TPA does not know the real secret key of the client for cloud storage auditing, but only holds an encrypted version. In the detailed protocol, we use the blinding technique with homomorphic property to form the encryption algorithm to encrypt the secret keys held by the TPA. It makes our protocol secure and the decryption operation efficient.

The security model of the cloud storage auditing protocol with verifiable outsourcing of key updates.

Chapter 2

SYSTEM ANALYSIS

2.1 Introduction

A Software Requirements Specification (SRS) is a description of a particular software product, program or set of programs that performs a set of functions in a target environment (IEEE Std. 830-1993).

2.2 Software Requirement Specification

2.2.1 User requirement

This project is developed using graphics in HTML,jsp. The options available are displayed in a menu format, like in an online editor. Clicking on any particular menu item through mouse or through keyboard a dropdown menu is displayed, listing all the options available under that menu item and the user can select the needed actions according to their wish.

2.2.2 Software requirement

Operating System : Windows XP/7
Web Server : Tomcat 7.0
Front End : HTML, Java, Jsp
Scripts : JavaScript.
Database : Mysql
Database Connectivity : JDBC.

2.2.3 Hardware requirement

- Hard disk : 80 GB
- RAM : 1 GB
- Processor : i3

Feasibility Study

Economic Feasibility

Economic feasibility attempts 2 weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system.

A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could include increased customer satisfaction, improvement in product quality better decision making timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, better employee morale.

Operational Feasibility

Proposed project is beneficial only if it can be turned into information systems that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? Here are questions that will help test the operational feasibility of a project:

Is there sufficient support for the project from management from users? If the current system is well liked and used to the extent that persons will not be able to see reasons for change, there may be resistance.

Are the current business methods acceptable to the user? If they are not, Users may welcome a change that will bring about a more operational and useful systems.

Have the user been involved in the planning and development of the project?

Early involvement reduces the chances of resistance to the system and in general and increases the likelihood of successful project.

Since the proposed system was to help reduce the hardships encountered. In the existing manual system, the new system was considered to be operational feasible.

Technical Feasibility

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, .at this point in time, not too many detailed design of the system, making it difficult to access issues like performance, costs on (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis.Understand the different technologies involved in the proposed system before commencing the project we have to be very clear about what are the technologies that are to be required for the development of the new system. Find out whether the organization currently possesses the required technologies.

Non- Functional Requirements

The major non-functional Requirements of the system are as follows

- Usability

  The system is designed with completely automated process hence there is no or less user intervention.

- Reliability

  The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

- Performance

  This system is developing in the high level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

- Supportability

  The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having JVM, built into the system.

- Implementation

  The system is implemented with Java environment. The java software development kit and net beans used as software and windows xp professional is used as the platform.

2.3 UML Description

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

- User Model View
  - This view represents the system from the users perspective.
  - The analysis representation describes a usage scenario from the end-users perspective.

- Structural model view
  - In this model the data and functionality are arrived from inside the system.
  - This model view models the static structures.

- Behavioral Model View

  It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

- Implementation Model View

  In this the structural and behavioral as parts of the system are represented as they are to be built.

- Environmental Model View

  In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the system.
- UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

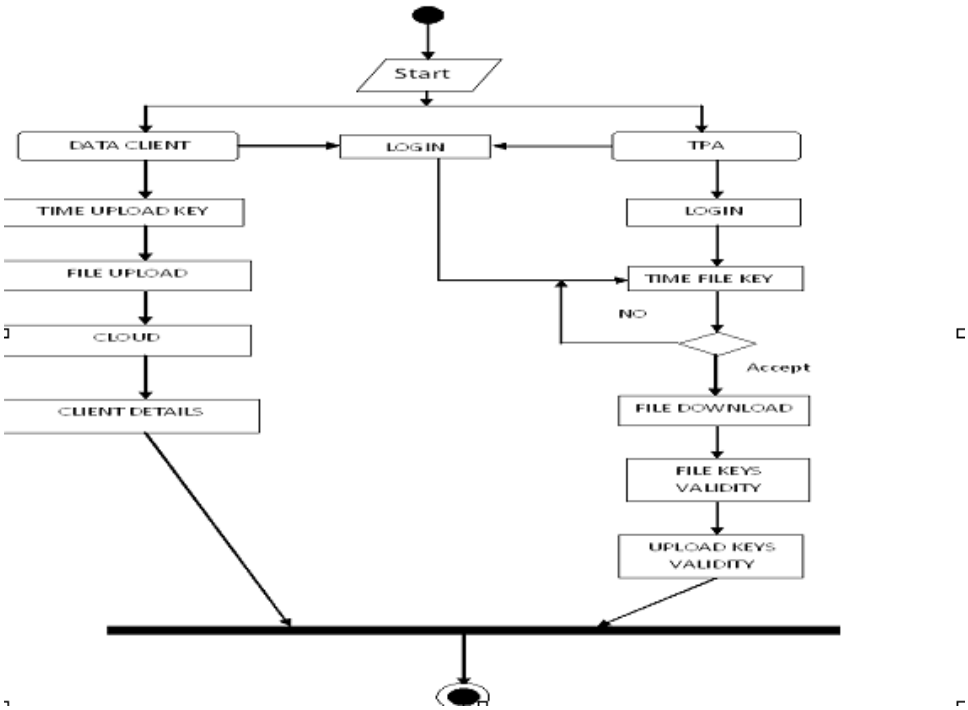
2.4 UML Diagrams

2.4.1 USE CASE DIAGRAM

Use Case: Use case describes the behavior of a system. It is used to structure things in a model. It contains multiple scenarios, each of which describes a sequence of actions that is clear enough for outsiders to understand.

Actor: An actor represents a coherent set of roles that users of a system play when interacting with the use cases of the system. An actor participates in use cases to accomplish an overall purpose. An actor can represent the role of a human, a device, or any other systems.

2.5 Activity Diagram



2.6 SEQUENCE DIAGRAM:

This diagram is simple and visually logical, so it is easy to see the sequence of the flow of control. It also clearly shows concurrent processes and activations in a design.

Object: Object can be viewed as an entity at a particular point in time with a specific value and as a holder of identity that has different values over time. Associations among objects are not shown. When you place an object tag in the design area, a lifeline is automatically drawn and attached to that object tag.

Actor: An actor represents a coherent set of roles that users of a system play when interacting with the use cases of the system. An actor participates in use cases to accomplish an overall purpose. An actor can represent the role of a human, a device, or any other systems.

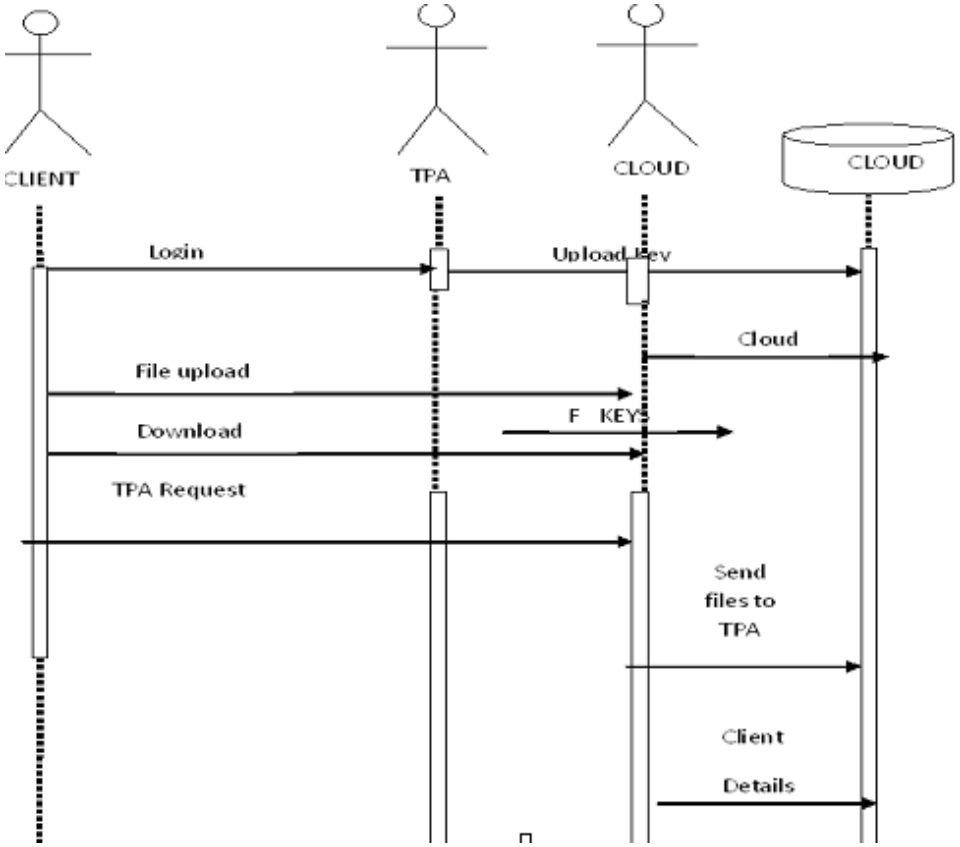
Message: A message is a sending of a signal from one sender object to other receiver object(s). It can also be the call of an operation on receiver object by caller object. The arrow can be labeled with the name of the message (operation or signal) and its argument values

Duration Message: A message that indicates an action will cause transition from one state to another state.

Self Message: A message that indicates an action will perform at a particular state and stay there.

Create Message: A message that indicates an action that will perform between two states.

Sequence Diagram



Chapter 3

DESIGN

3.1 Design Principle

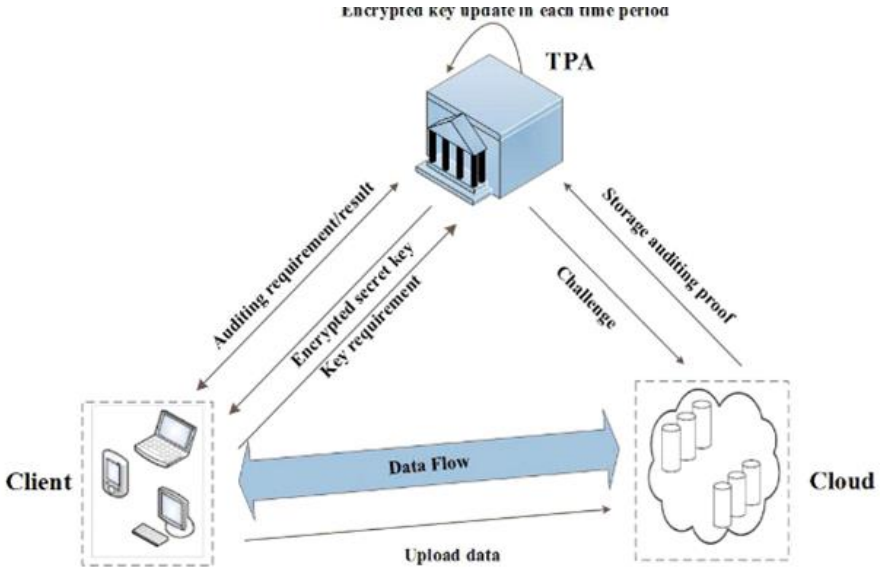
Design is a meaningful engineering representation of something that is to be built. Software design is a process through which the requirements are translated into a representation of the software. Design is the place where quality is fostered in software engineering. Design is the perfect way to accurately translate a customer's requirement in to a finished software product. Design creates a representation or model, provides detail about software data structure, architecture, interfaces and components that are necessary to implement a system. This chapter discusses about the design part of the project. Here in this document the various UML diagrams that are used for the implementation of the project are discussed.

The Unified Modeling Language (UML) is a visual modeling language used to specify, visualize, construct and document a software intensive system. The embedded real-time software systems encountered in applications such as telecommunications, school systems, aerospace, and defense typically tends to be large and extremely complex. It is crucial in such systems that the software is designed with a sound architecture. A good architecture not only simplifies construction of the initial system, but also, readily accommodates changes forced by a steady stream of new requirements.

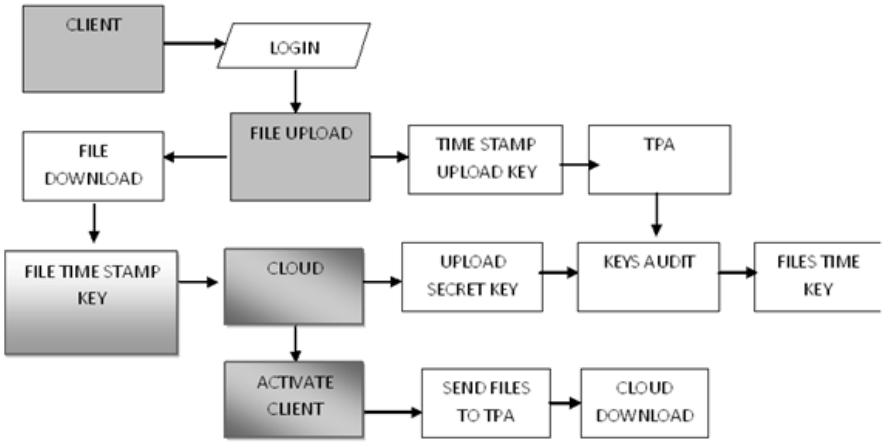
The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

The primary goals in the design of the UML are: Provide users with a ready-to-use, expressive visual modeling language so they can develop and exchange meaningful models. Provide extensibility and specialization mechanisms to extend the core concepts. Be independent of particular programming languages and development processes. Provide a formal basis for understanding the modeling language. Encourage the growth of the OO tools market. Support higher-level development concepts such as collaborations, frameworks, patterns and components. Integrate best practices.

3.2 ARCHITECTURE DIAGRAM



3.3 DATA FLOW DIAGRAM



3.4 CLASS DIAGRAM:

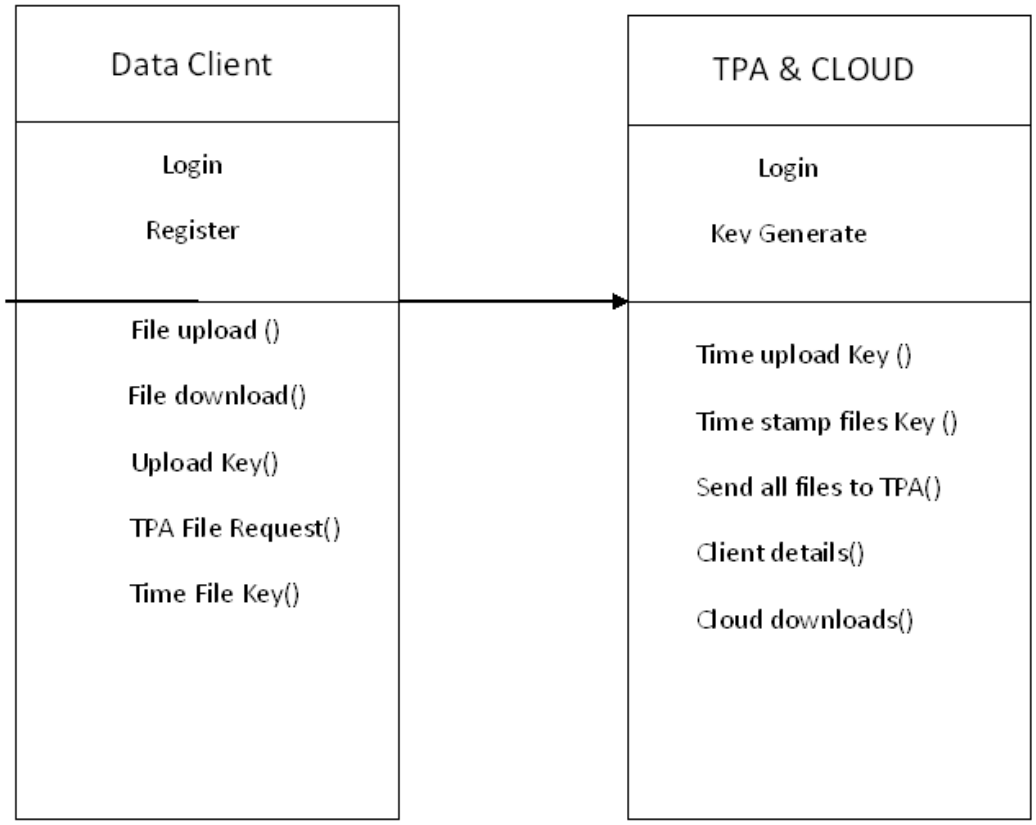
Class: A Class is a description for a set of objects that shares the same attributes, and has similar operations, relationships, behaviors and semantics.

Generalization: Generalization is a relationship between a general element and a more specific kind of that element. It means that the more specific element can be used whenever the general element appears. This relation is also known as specialization
or inheritance link.

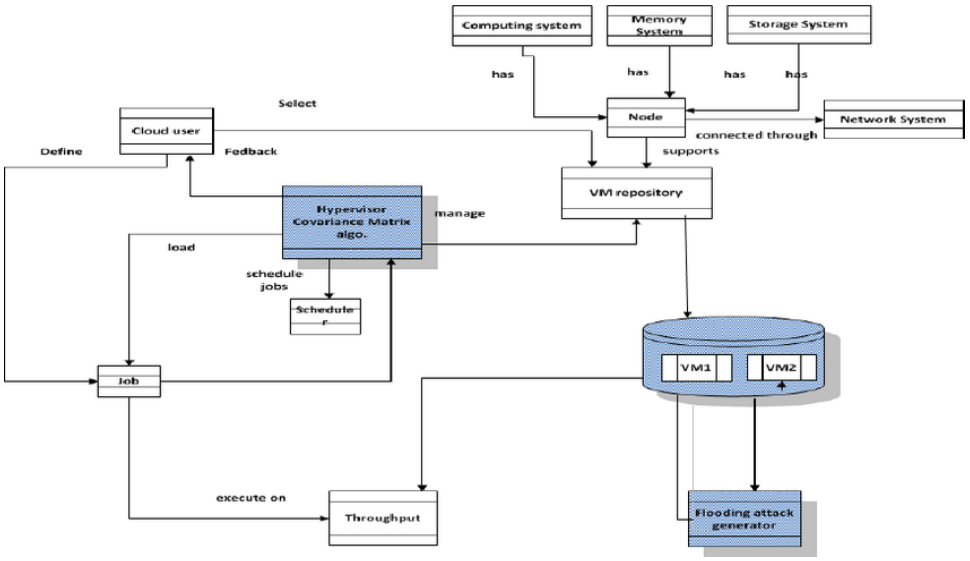
Realization: Realization is the relationship between a specialization and its implementation. It is an indication of the inheritance of behavior without the inheritance of structure.

Association: Association is represented by drawing a line between classes. Associations represent structural relationships between classes and can be named to facilitate model understanding. If two classes are associated, you can navigate from an object of one class to an object of the class.

Aggregation: Aggregation is a special kind of association in which one class represents as the larger class that consists of a smaller class. It has the meaning of "has-a" relationship.



CLASS DIAGRAM OF CLOUD ENVIRONMENT



Chapter 4

IMPLEMENTATION & RESULTS

4.1 Introduction

The most crucial phase of any project is the implementation. This includes all those activities that take place to convert from the old system to the new system. It involves setting up of the system for use by the concerned end user. A successful implementation involves a high level of interaction between the analyst, programmers and the end user. The most common method of implementation is the phased approach, which involves installation of the system concurrently with the existing system. This has its advantage in that the normal activity carried out, as part of the existing system is anyway hampered. The end users are provided with sufficient documentation and adequate training in the form of demonstration/presentation in order to familiarize with the system.

4.2 Module design and organization

MODULE DESCRIPTION:

Client Module

This module includes the Client registration and client login details.

Every Client need to register while accessing to the cloud.

Every Client will be activated by the Cloud.

After Cloud activated, every Client need to provide time stamp upload key to upload a new files into cloud.

Time stamp upload key will be provided by third party auditor.

Client can view file details and download the file using time stamp file key provided by TPA.

Third Party Auditor

It acts as admin.

TPA Provide time Upload secret key in Encrypted state for every client to upload new file into cloud. It will be send as in directly while Client downloading the upload key.

The upload secret key, while user downloading key it will updated according to time.

After cloud given auditing proof then only TPA can audit all files.

And also provide the File Stamp key for all files to the client request for corresponding files key.

Cloud Module

Activate data client.

Cloud sends storage auditing proof for all files to TPA.

Cloud can view the client downloaded files from cloud.

Generate time period key

Each time client accessing and downloading the file from cloud, TPA will provide each time file update key to client registered mail Id. So same file key will not be there for same file.

- Time stamp upload key will be provided by TPA. Client can download the upload key each time client uploading new file into cloud and they need not to give request key from TPA.
- At the time of client downloading the time stamp upload key, the request will send in directly to TPA and update according to time by TPA and send encrypted upload secret key to client.
- And finally, client can decrypt download the upload secret key.
- After getting decrypt upload secret key, now Client can upload a new file into cloud.

Important Java Code Snippets

Dbconn.java

```
package DatabaseConnectivity;
import java.sql.Connection;
import java.sql.DriverManager;
public class Dbconn {
   public static Connection getConnection() {
      Connection con = null;
      try {
         Class.forName("com.mysql.jdbc.Driver");
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/Enabling_Cloud",     "root",
"root");
      } catch (Exception ex) {
         ex.printStackTrace();
      }
      return con;
   }
}
```

Register.java

```
package Db_register;
import DatabaseConnectivity.Dbconn;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.Statement;
import java.util.Random;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class register extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse response)
```

```
throws IOException{
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
       Connection con = null;
    Statement st = null;
    String name = request.getParameter("name");
    String email = request.getParameter("email");
    String pass = request.getParameter("pass");
    String repass = request.getParameter("repass");
    String dob = request.getParameter("dob");
    String gender = request.getParameter("gender");
    String contact = request.getParameter("con");
    String loc = request.getParameter("location");
    try {
    Random s = new Random();
    int a = s.nextInt(10000 - 5000) +25000 ;
    System.out.print(a);
        con = Dbconn.getConnection();
        st = con.createStatement();
        int i = st.executeUpdate("insert into register (name, email, pass, repass, dob, gender, contact, loc, ukey) values ('" + name + "','" + email + "','" + pass + "','" + repass + "','" + dob + "','" + gender + "','" + contact + "','"+loc+"', '"+a+"')");
        if (i != 0) {
            response.sendRedirect("index.html?msg=success");
        } else {
            response.sendRedirect("register.jsp?msgg=failed");
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    }}
```

TrippleDes.java

```
package EncryptionDecryption;
```

```
import java.security.spec.KeySpec;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESedeKeySpec;
import org.apache.commons.codec.binary.Base64;
public class TrippleDes {
    private static final String UNICODE_FORMAT = "UTF8";
    public static final String DESEDE_ENCRYPTION_SCHEME = "DESede";
    private KeySpec ks;
    private SecretKeyFactory skf;
    private Cipher cipher;
    byte[] arrayBytes;
    private String myEncryptionKey;
    private String myEncryptionScheme;
    SecretKey key;
    public TrippleDes() throws Exception {
        myEncryptionKey = "ThisIsSpartaThisIsSparta";
        myEncryptionScheme = DESEDE_ENCRYPTION_SCHEME;
        arrayBytes = myEncryptionKey.getBytes(UNICODE_FORMAT);
        ks = new DESedeKeySpec(arrayBytes);
        skf = SecretKeyFactory.getInstance(myEncryptionScheme);
        cipher = Cipher.getInstance(myEncryptionScheme);
        key = skf.generateSecret(ks);
    }
    public String encrypt(String unencryptedString) {
        String encryptedString = null;
        try {
            cipher.init(Cipher.ENCRYPT_MODE, key);
            byte[] plainText = unencryptedString.getBytes(UNICODE_FORMAT);
            byte[] encryptedText = cipher.doFinal(plainText);
            encryptedString = new String(Base64.encodeBase64(encryptedText));
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
        return encryptedString;
    }
    public String decrypt(String encryptedString) {
        String decryptedText=null;
        try {
            cipher.init(Cipher.DECRYPT_MODE, key);
            byte[] encryptedText = Base64.decodeBase64(encryptedString);
            byte[] plainText = cipher.doFinal(encryptedText);
            decryptedText= new String(plainText);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return decryptedText;
    }
}
```

Ftpcon.java

```
package FileUpload;
import java.io.File;
import java.io.FileInputStream;
import org.apache.commons.net.ftp.FTPClient;
public class Ftpcon {
    FTPClient client = new FTPClient();
    FileInputStream fis = null;
    boolean status;
    public boolean upload(File file,String fname) {
        try {
            client.connect("ftp.drivehq.com");
            client.login("drive05", "drive15");
            client.enterLocalPassiveMode();
            fis = new FileInputStream(file);
            status = client.storeFile(" /king/" + fname, fis);
            client.logout();
            fis.close();
        } catch (Exception e) {
```

```
            System.out.println(e);
        }
        if (status) {
            System.out.println("success");
            return true;
        } else {
            System.out.println("failed");
            return false;
        }
    }
}
```

Mail.java

```
package Mail;
import java.util.Properties;
import javax.mail.Message;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
public class MailSender {
    public static boolean sendMail(String receiver, String title, String msg){
        boolean bool = true;
        final String password = "cloudcloud";
        final String from   = "cloudcomputing40@gmail.com";
        String to   = receiver;

        Properties props = new Properties();
        props.put("mail.smtp.host", "smtp.gmail.com");
        props.put("mail.smtp.socketFactory.port", "465");
        props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "465");
        Session session = Session.getDefaultInstance(props,
        new javax.mail.Authenticator() {
```

```
        protected PasswordAuthentication getPasswordAuthentication() {
        return new PasswordAuthentication(from,password);
        }
        });
           try{
              System.out.println("within try mail sending");
                    Message message = new MimeMessage(session);
                    message.setFrom(new InternetAddress(from));
                    message.setRecipients(Message.RecipientType.TO,
                    InternetAddress.parse(to));
                          message.setSubject(title);
                          message.setText(msg);
                    Transport.send(message);
           }catch(Exception ex){
              bool = false;
              ex.printStackTrace();
           }
        return bool;
     }
}
```

client_login.java

```
package LoginAction;
import DatabaseConnectivity.Dbconn;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
public class client_login extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response) throws IOException
        {
    response.setContentType("text/html;charset=UTF-8");
    PrintWriter out = response.getWriter();
    try {
        HttpSession user = request.getSession(true);
        String uname=request.getParameter("name");
      String pass=request.getParameter("pass");
      Connection con=Dbconn.getConnection();
      Statement st=con.createStatement();
      ResultSet rs=st.executeQuery("select * from register where email="'+uname+"' AND pass="'+pass+"'");
          if(rs.next()){
             String s=rs.getString("status");
             String name=rs.getString("name");
            if(s.equalsIgnoreCase("Yes")) {
                user.setAttribute("email", uname);
                user.setAttribute("uname", name);

          response.sendRedirect("user_page.jsp");
            }
             else {
                out.println("you not a activated");
             }
          }
            else{
   out.println("Incorrect Email and password");
}
}
catch(Exception e){
```

```
            out.println(e);
        }
      }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    public String getServletInfo() {
        return "Short description";
    }
}
```

cloud_login.java

```
package LoginAction;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class cloud_login extends HttpServlet {
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String uname=request.getParameter("name");
           String pass=request.getParameter("pass");
           if(uname.equalsIgnoreCase("cloud")&&pass.equalsIgnoreCase("cloud")){
              // out.println("success... ");
```

```
            response.sendRedirect("cloud_page.jsp");
          }
          else{
            out.println("incorrect username or password ");
          }
        }
        catch(Exception e){
          out.println(e);
        }
        finally {
          out.close();
        }
    }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    public String getServletInfo() {
        return "Short description";
    }
}
```

tpa_login.java

```
package LoginAction;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class tpa_login extends HttpServlet {
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
```

```
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String uname=request.getParameter("name");
           String pass=request.getParameter("pass");

           if(uname.equalsIgnoreCase("TPA")&&pass.equalsIgnoreCase("TPA")){
             // out.println("success... ");
              response.sendRedirect("admin_page.jsp");
           }
           else{
              out.println("incorrect username or password ");
           }
        }
        catch(Exception e){
            out.println(e);
        }
        finally {
            out.close();
        }
    }
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
    public String getServletInfo() {
        return "Short description";
    }
}
```

Chapter 5

TESTING & VALIDATION

5.1 Introduction

The system once finished has to go through a series of testing in order to ensure that it works the way it ought to. The various types of testing measures to be taken are:

Test to see if the requirements are taken care of.
Test to see if all the inputs are handled effectively.
Test the system by traversing all the paths and discover my surprises.
Check if errors and the exceptions have been handled properly.
See if validations of input data are taken care of.
Testing Objectives
Types of Tests:
There are six types of test a software product must satisfy:
Unit Test
Functional Test
Performance Test
Stress Test and
Structural Test
System Test

Functional Test, Performance Test and Stress Test are known as Black box testing. Structural Test is referred as White box or Glass Box testing.

1.Unit Testing:

Unit testing focuses verification effort on the smallest unit of software design. Unit Testing is considered as an equivalent to coding step. After the source level code has been developed, reviewed and verified for correct syntax, unit test case design begins. In most of the applications, a driver is nothing more than a main program that accepts test case data, passes such data to the module to be tested and prints the relevant results.

2.Functional Testing:

Functional Testing involves testing the system under typical operating condition, typical input values and for typical expected results. The functional boundaries specifies boundary within which the system can function. three types of functional tests are done

Checking the documented functions.

Checking with maximum values.

Checking with valid input.

3.Performance Testing:

Performance Testing is conducted to identify the bottlenecks in the system and to fine-tune the overall performance of the system.

4.Stress Testing:

Stress Testing involves overloading the system in various ways and observing the behavior. The system is tested with high network traffic and more number of clients. Stress test provides valuable insight about the strengths and weakness of a system.

5.Structural Testing:

Structural Testing are concerned with examining the internal processing logic of a program and traversing particular execution paths.

6.System Testing:

System Testing involves two kinds of activities

1.Integration Testing and

2. Acceptance Testing

1.Integration Testing:

Integrating all the functionalities since some functions work perfectly when run alone tests the system.

Integration Testing is of two types

Top down Integration Testing and

Bottom-Up Integration testing.

Top down Integration Testing:

In Top down Integration the top of the hierarchy is tested then one or two immediately subordinate routines are tested.

Bottom-Up Integration Testing:

First the modules at the very bottom, which have no subordinates, are tested then these are combined with higher-level modules for testing.

2. Acceptance Testing:

Acceptance Testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements.

Testing Strategies

Quality Assurance

The aim of this step is to maintain or to ensure the quality of the system developed. The quality assurance goals in the system life cycle involves

1. **Quality factors specification: -** This was done to determine the factors that lead to high quality of a system.

a. **Correctness**- The extent to which a program meets System specification.

b. **Reliability –** The degree to which a program meets system specification.

c. **Efficiency** - The amount of computer resources required by the entire program to perform a function.

d. **Usability** – The effort required learning and operating the system.

e. **Maintainability** – The ease with which the program errors are located and corrected.

f. **Testability -** The effort required to test a program to ensure its correct performance.

g. **Portability** – The ease of transporting a program from one hardware configuration to another.

h. **Accuracy** - The required precision in input editing, computation and output.

i. **Error Tolerance** – Error detection and correction versus error avoidance.

j. **Expandability** - Ease of adding or expanding existing databases.

k. **Access Controls and Audit** – Control of access to the system and the extent to which that access can be audited.

l. **Communication** – How useful the input and output of the system are.

1.**Software Requirements Specification**: - This was done to generate the required documents that provide the technical specification for the design and development of the software.

2. **Software Design Specification**: - This was done in order to provide the functions and features described in the previous stage.

3.**Software Testing and Implementation**: - This was done to provide necessary software adjustment for the system to continue to comply with the original specifications.

Quality Assurance is the review of software and related documentation for correctness, accuracy, maintainability, reliability, and expendability. This also includes assurances that the system meets the specifications and requirements for its intended use performance.

5.2 Design of test cases and scenarios

Test Cases

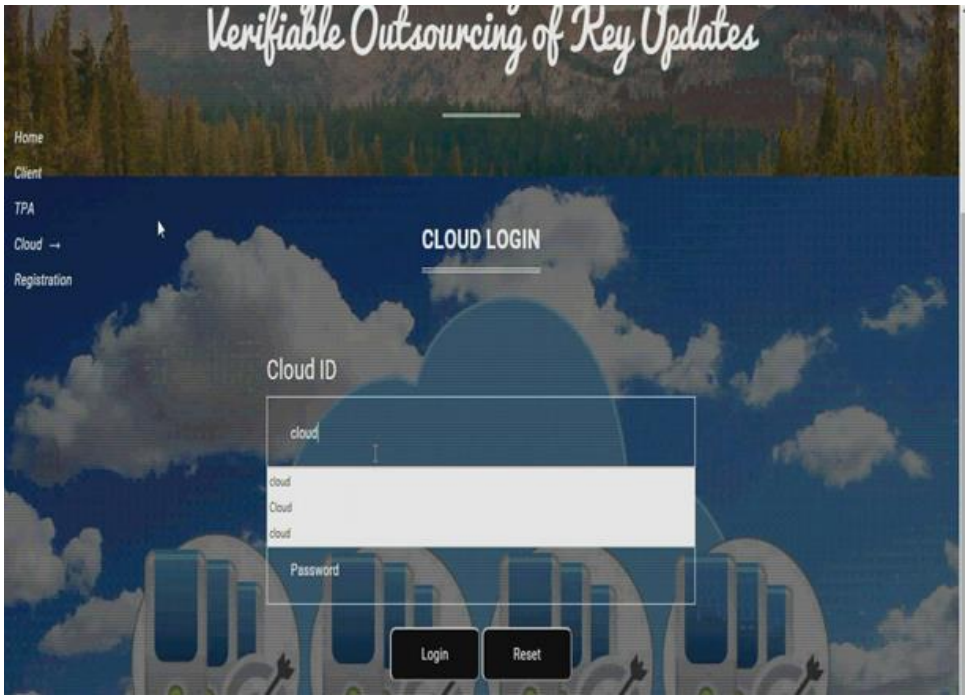
Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Test case number	Test case	Input	Expected output	Obtained output
1	User Registration Login	Give registered username and password.	User Login page is open. Authorized user can upload and download the file	User Login page is open. authorized user can download the file.
2	TPA Login	TPA username and password	TPA Login page is open.View the file.stored into the data base. Sends the file key to user.	TPA Login page is open. View the file. Stored into the data base. Sends the file key to user.
3	CSP Login	Give CSP user name and password	CSP Login Page is send files to TPA..	CSP Login Page is send files to TPA..

5.3 Output Screens

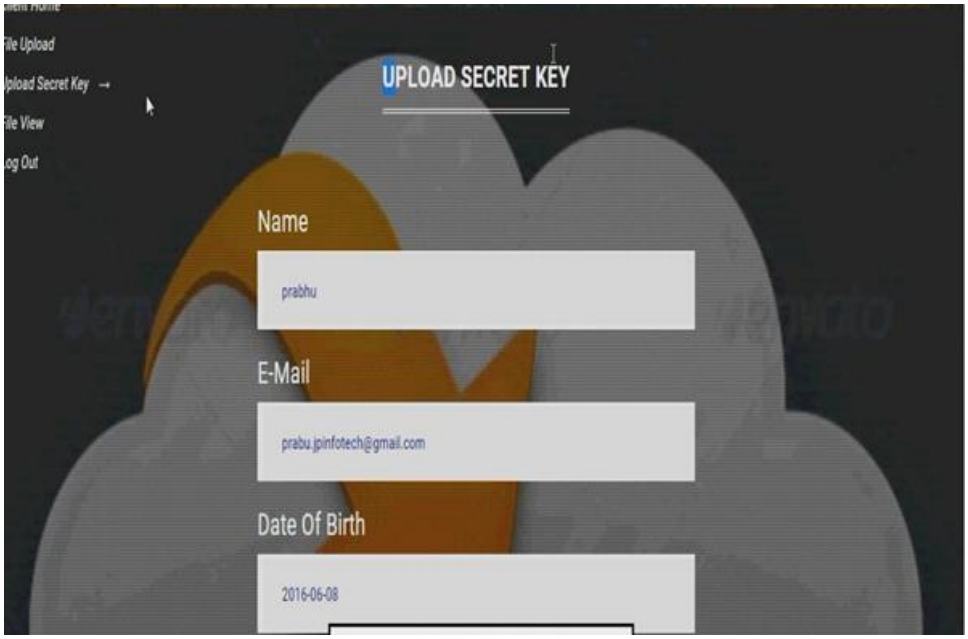
5.3.1 Output Screen Result 1

5.3.2 Output Screen Result 2

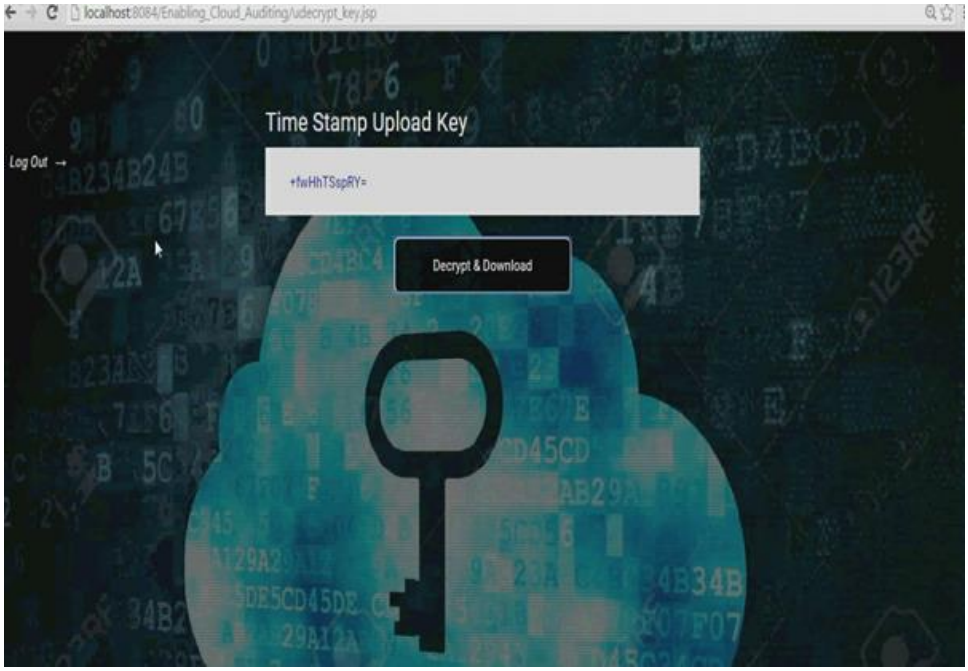


5.3.3 Output Screen Result 3

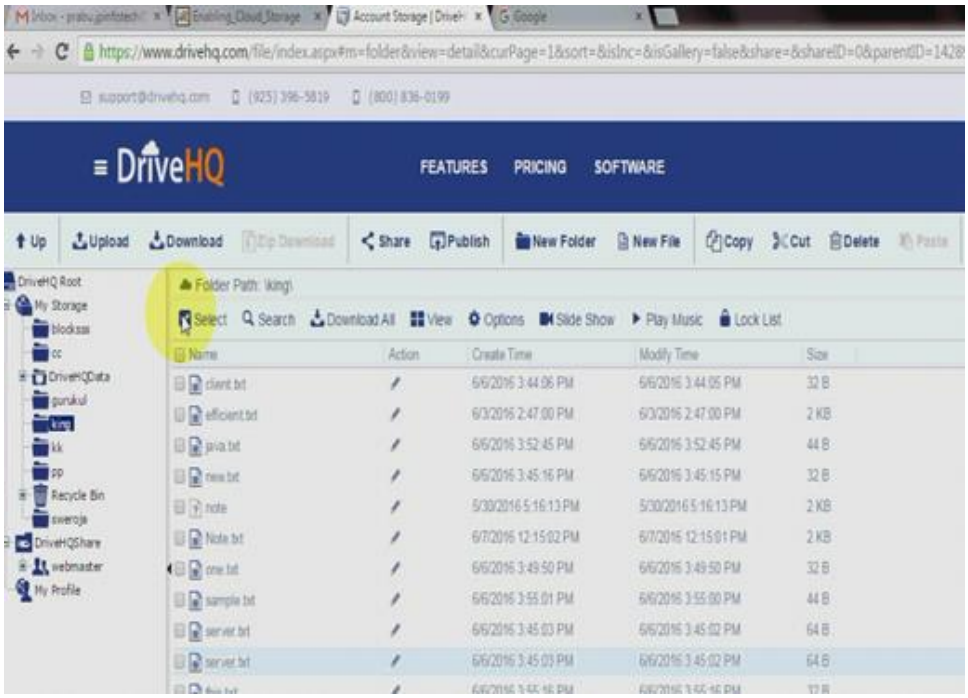
5.3.4 Output Screen Result 4



5.3.5 Output Screen Result 5



5.3.6 Output Screen Result 6



5.3.7 Output Screen Result 7

5.3.8 Output Screen Result 8

CONCLUSION

Conclusion

In this project, we study on how to outsource key updates for cloud storage auditing with key-exposure resilience. We propose the first cloud storage auditing protocol with verifiable outsourcing of key updates. In this protocol, key updates are outsourced to the TPA and are transparent for the client. Cloud storage auditing is viewed as an important service to verify integrity of data in public cloud. Current auditing protocols are based on assumption that auditing protocols are absolutely secure which is not always true. The integrity of the data formerly stored in cloud can still be substantiated even if the client's current secret key for cloud storage auditing is bare in these kinds of protocols. It is enacted in the definition and the security model of auditing protocol with key-exposure resilience, and has given the practical solution.

The security proof and the asymptotic presentation assessment depicted that the protocol is secure and efficient. We use binary tree structure and pre order traversal to update secret keys of the client .The security proof and performance analysis show that the proposed protocol is secure and efficient.

Future enhancements

In addition, the TPA only sees the encrypted version of the client's secret key, while the client can further verify the validity of the encrypted secret keys when downloading them from the TPA. We give the formal security proof and the performance simulation of the proposed scheme. Instead of DriveHQ , Amazon web services or google cloud can be used. Instead of client signing in multiple times single Sign in can be implemented in a more secure manner.

REFERENCES

Reference Papers

[1] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," *Adv. Comput.*, vol. 54, pp. 215–272, 2002.

[2] D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing of algebraic computations," in *Proc. 6th Annu. Conf. Privacy, Secur. Trust*, 2008, pp. 240–245.

[3] C. Wang, K. Ren, and J. Wang, "Secure and practical outsourcing of linear programming in cloud computing," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 820–828.

[4] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in *Proc. 17th Eur. Symp. Res. Comput. Secur.*, 2012, pp. 541–556.

[5] G. Ateniese *et al.*, "Provable data possession at untrusted stores," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 598–609.

[6] A. Juels and B. S. Kaliski, Jr., "PORs: Proofs of retrievability for large files," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 584–597.

[7] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Advances in Cryptology*. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.

[8] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. 4th Int. Conf. Secur. Privacy Commun. Netw.*, 2008, Art. ID 9.

[9] F. Sebe, J. Domingo-Ferrer, A. Martinez-balleste, Y. Deswarte, and J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 8, pp. 1034–1038, Aug. 2008.

[10] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiplereplica provable data possession," in *Proc. 28th IEEE Int. Conf. Distrib. Comput. Syst.*, Jun. 2008, pp. 411–420.

[11] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Efficient provable data possession for hybrid clouds," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, 2010, pp. 756–758.

[12] C. Wang, K. Ren, W. Lou, and J. Li, “Toward publicly auditable secure cloud data storage services,” *IEEE Netw.*, vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010.

[13] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, “Enabling public auditability and data dynamics for storage security in cloud computing,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 5, pp. 847–859, May 2011.

[14] K. Yang and X. Jia, “Data storage auditing service in cloud computing: Challenges, methods and opportunities,” *World Wide Web*, vol. 15, no. 4, pp. 409–428, 2012.

[15] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, “Dynamic audit services for outsourced storages in clouds,” *IEEE Trans. Services Comput.*, vol. 6, no. 2, pp. 227–238, Apr./Jun. 2013.

[16] K. Yang and X. Jia, “An efficient and secure dynamic auditing protocol for data storage in cloud computing,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 9, pp. 1717–1726, Sep. 2013.

[17] B. Lynn. (2015). *The Pairing-Based Cryptographic Library*. [Online]. Available: http://crypto.Stanford.edu/pbc/

APPENDIX A - ABBREVIATIONS

- DB – Database
- DOM – Document Object Model
- ER – Entity Relationship
- HTML – Hyper Text Markup Language

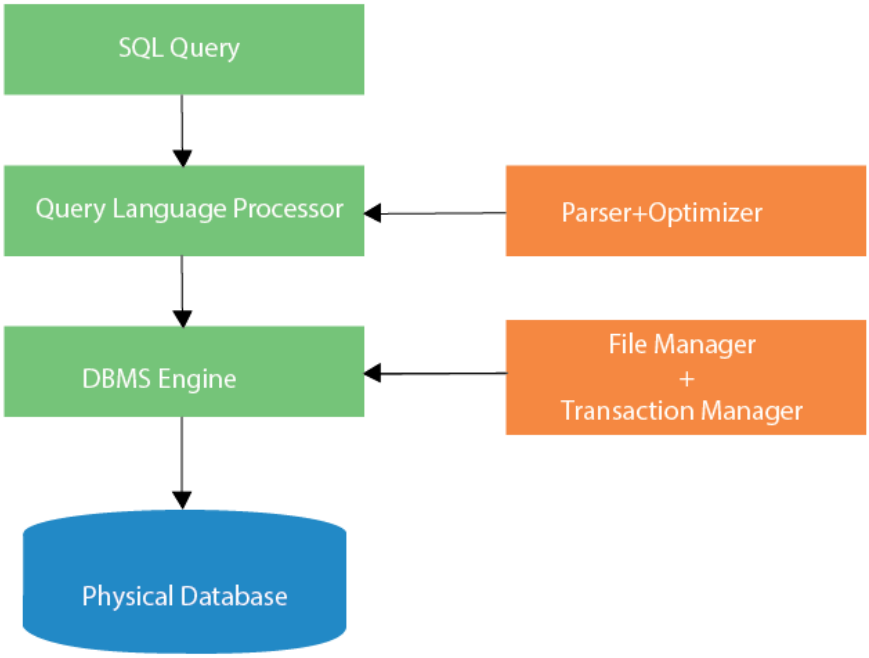
- HTTP – Hypertext Transfer Protocol
- JS – JavaScript
- JSON – JavaScript Object Notation
- JWT – JSON Web Token
- ODM – Object Data Modeling
- SOA – Service Oriented Architecture
- UI – User Interface
- UML – Unified Modeling Language
- URL - Uniform Resource Locator
- XML – Extensible Markup Language

APPENDIX B – SOFTWARE INSTALLATION PROCEDURE

SQL INSTALLATION

SQL is a standard language for accessing and manipulating databases.

- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert records in a database
- SQL can update records in a database
- SQL can delete records from a database
- SQL can create new databases
- SQL can create new tables in a database
- SQL can create stored procedures in a database
- SQL can create views in a database
- SQL can set permissions on tables, procedures, and views



Step 1 – Download the Evaluation Edition from http://www.microsoft.com/download/en/details.aspx?id=29066

Once the software is downloaded, the following files will be available based on your download (32 or 64 bit) option.

ENU\x86\SQLFULL_x86_ENU_Core.box

ENU\x86\SQLFULL_x86_ENU_Install.exe

ENU\x86\SQLFULL_x86_ENU_Lang.box

OR

ENU\x86\SQLFULL_x64_ENU_Core.box

ENU\x86\SQLFULL_x64_ENU_Install.exe

ENU\x86\SQLFULL_x64_ENU_Lang.box

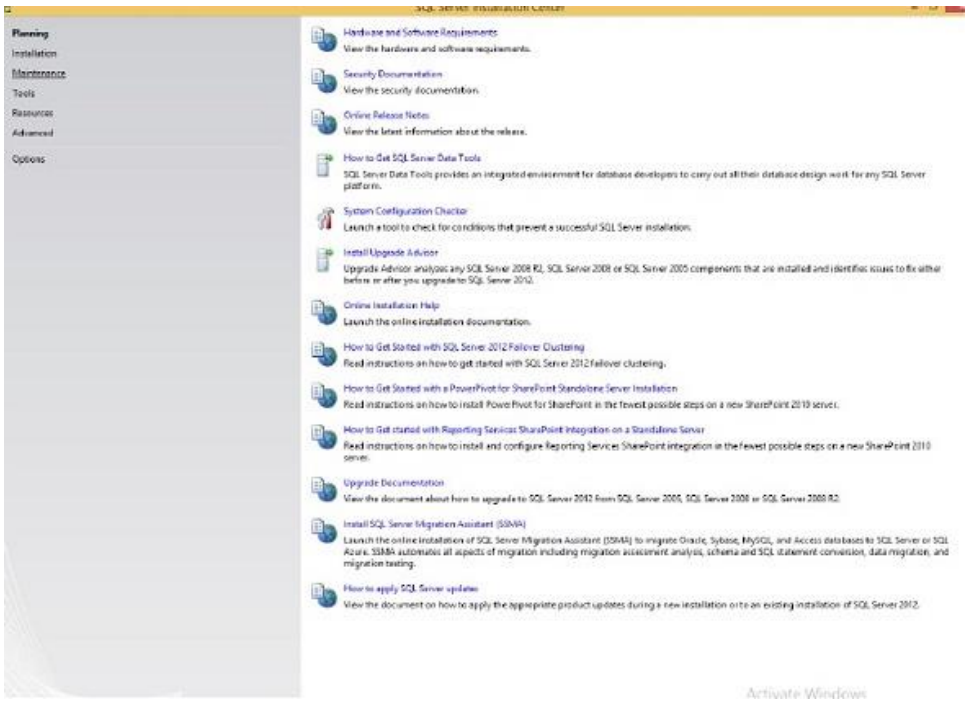
Note – X86 (32 bit) and X64 (64 bit)

Step 2 – Double-click the "SQLFULL_x86_ENU_Install.exe" or "SQLFULL_x64_ENU_Install.exe", it will extract the required files for installation in the"SQLFULL_x86_ENU" or "SQLFULL_x86_ENU" folder respectively.

Step 3 – Click the "SQLFULL_x86_ENU" or "SQLFULL_x64_ENU_Install.exe" folder and double-click "SETUP" application.

For understanding, here we have used SQLFULL_x64_ENU_Install.exe software.

Step 4 – Once we click on 'setup' application, the following screen will open.

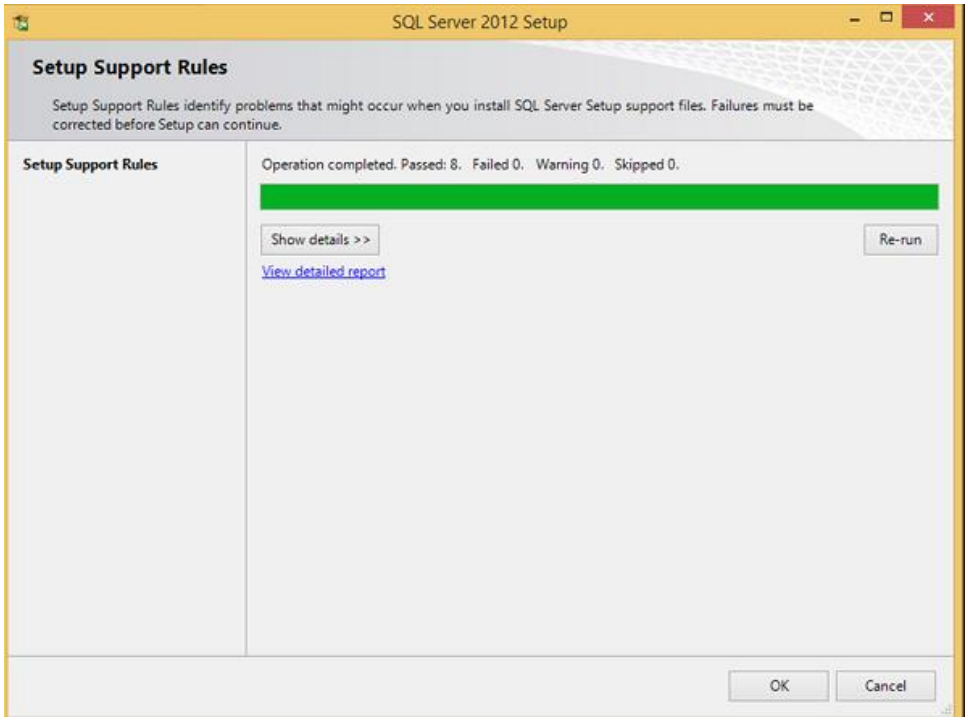


Setup

Step 5 – Click Installation which is on the left side of the above screen.

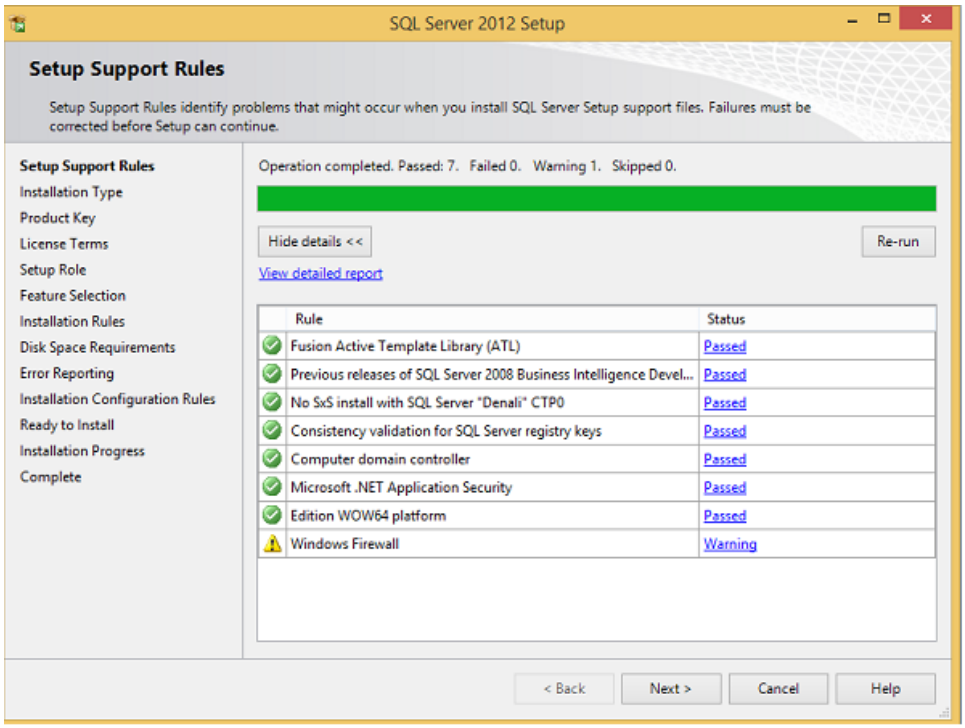
Installation

Step 6 – Click the first option of the right side seen on the above screen. The following screen will open.



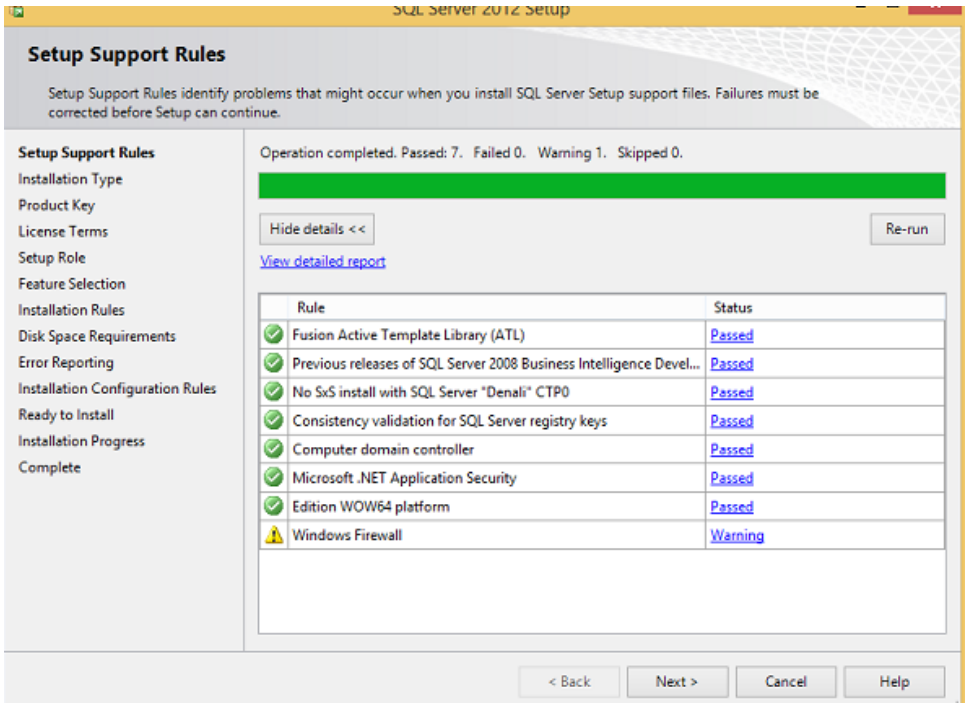
Support Rules

Step 7 – Click OK and the following screen pops up.



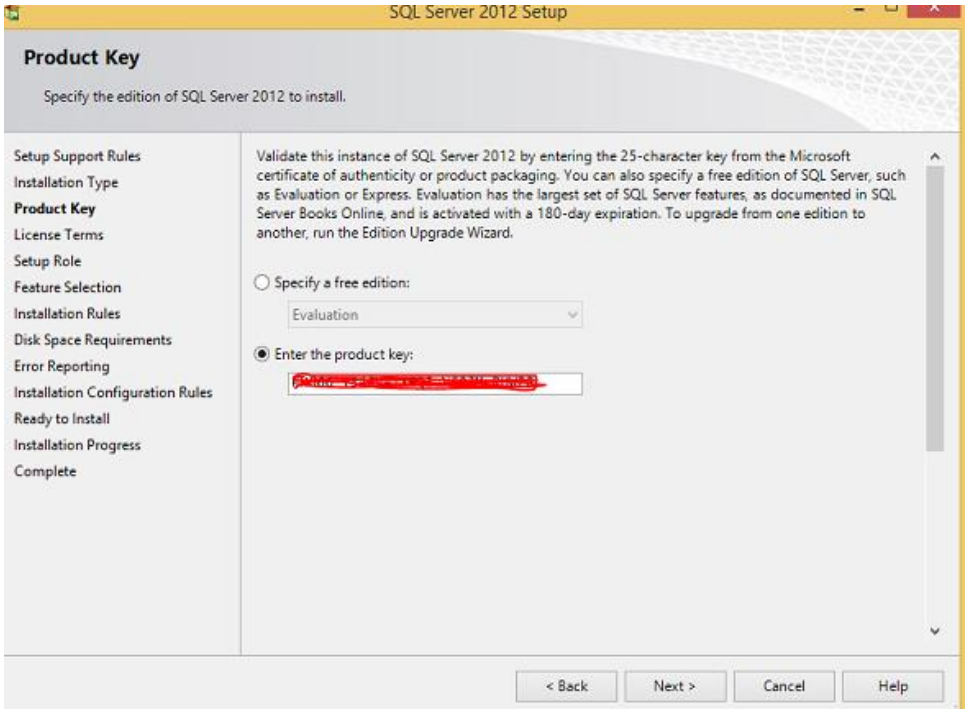
Setup Rules

Step 8 – Click Next to get the following screen.



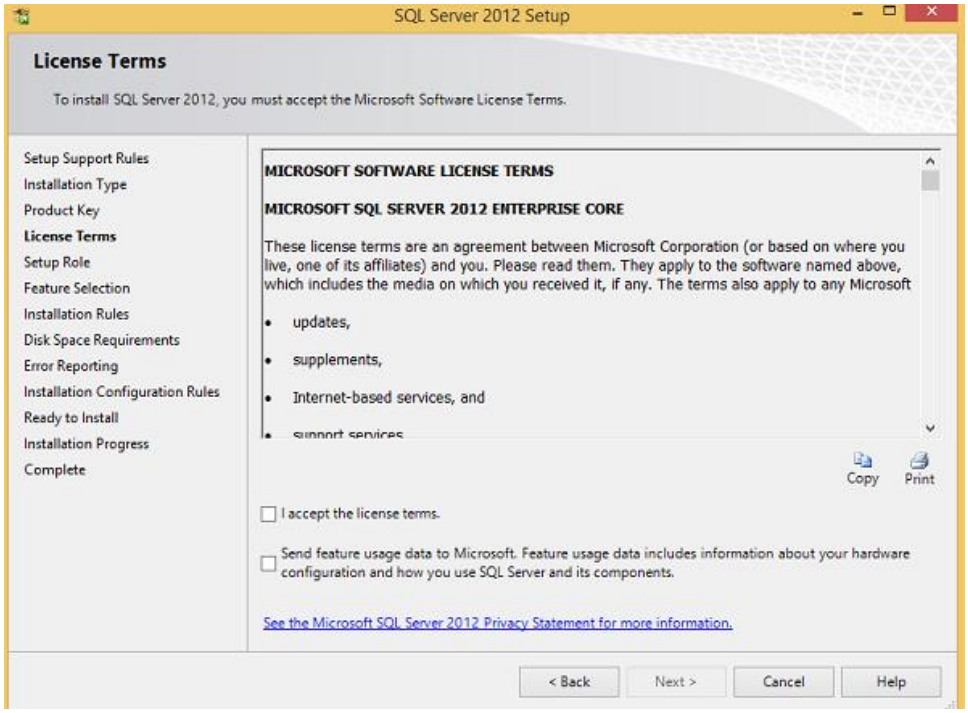
Product Key

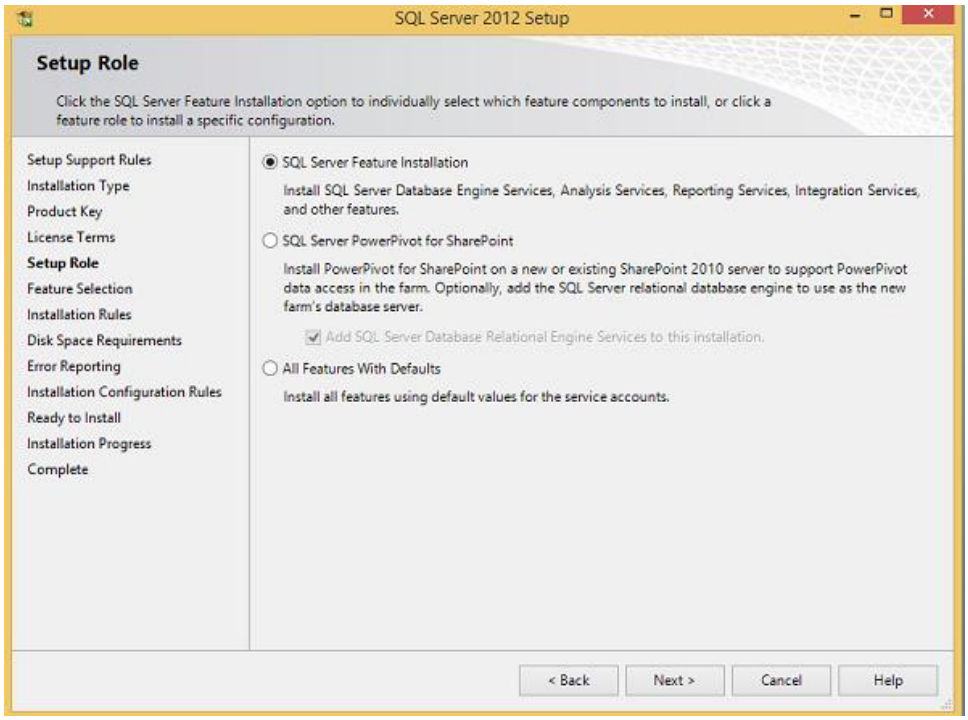
Step 9 – Make sure to check the product key selection and click Next.



License Terms

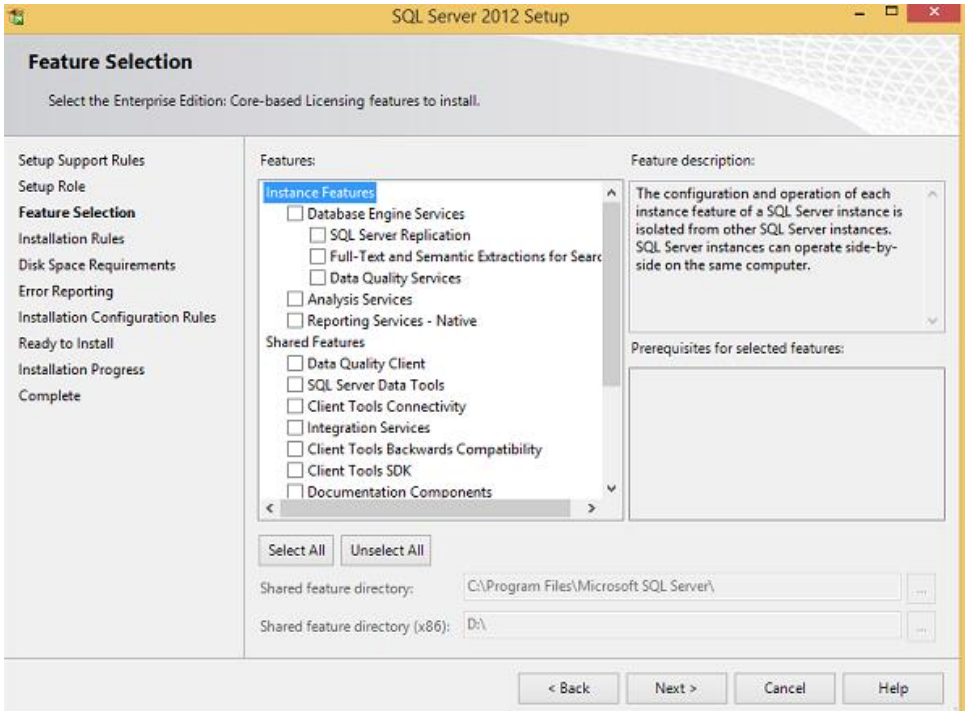
Step 10 – Select the checkbox to accept the license option and click Next.





Setup role

Step 11 – Select SQL Server feature installation option and click Next.

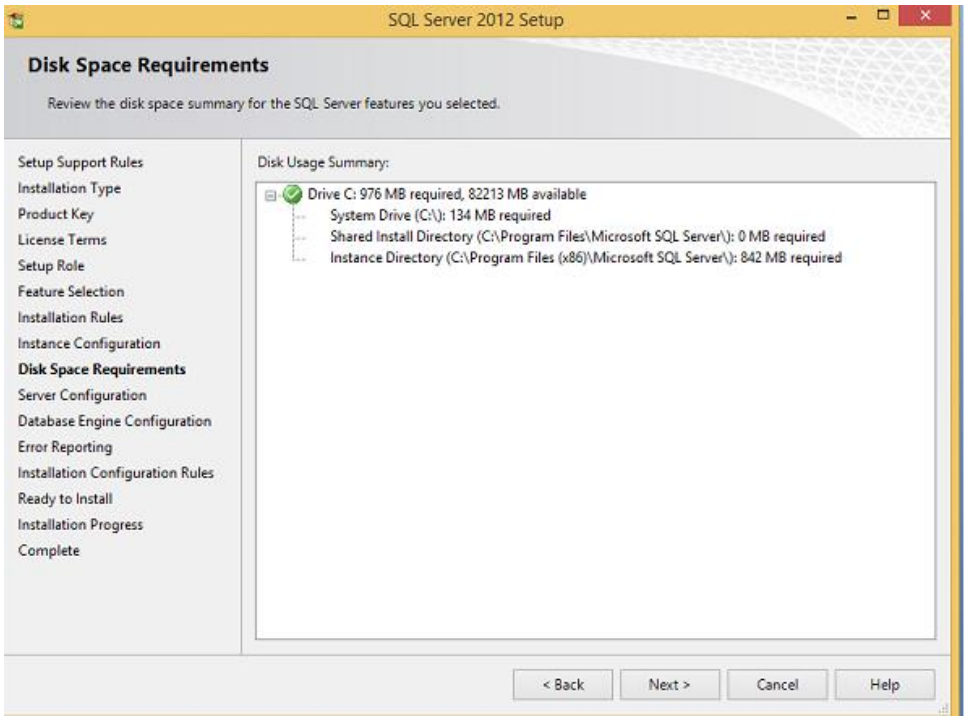


Feature Selection

Step 12 – Select Database engine services checkbox and click Next.

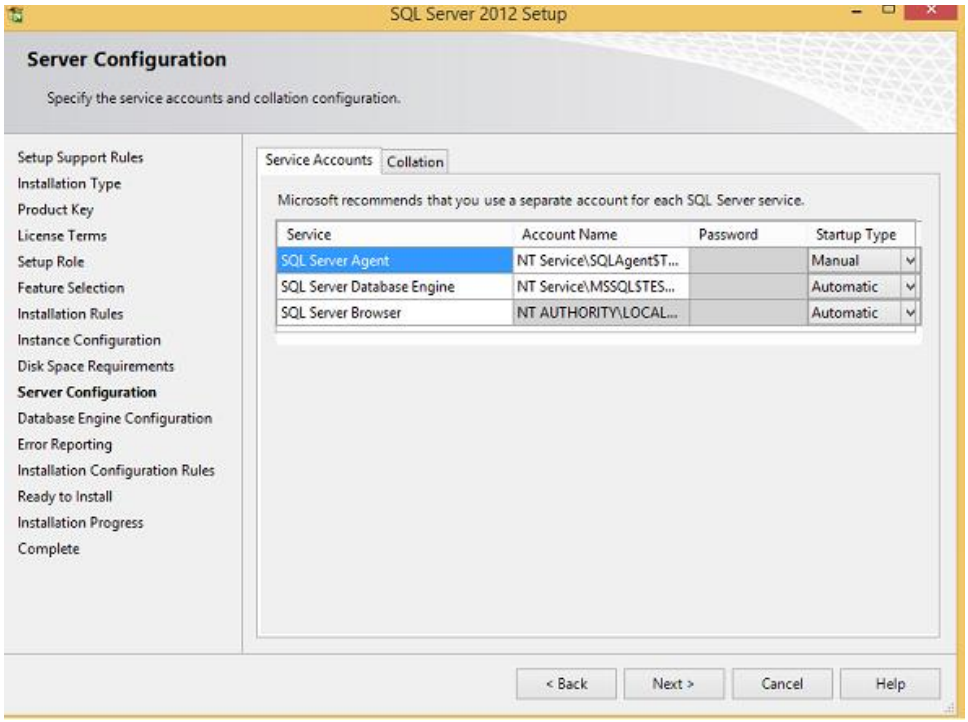
Instance Configuration

Step 13 – Enter the named instance (here I used TestInstance) and click Next.



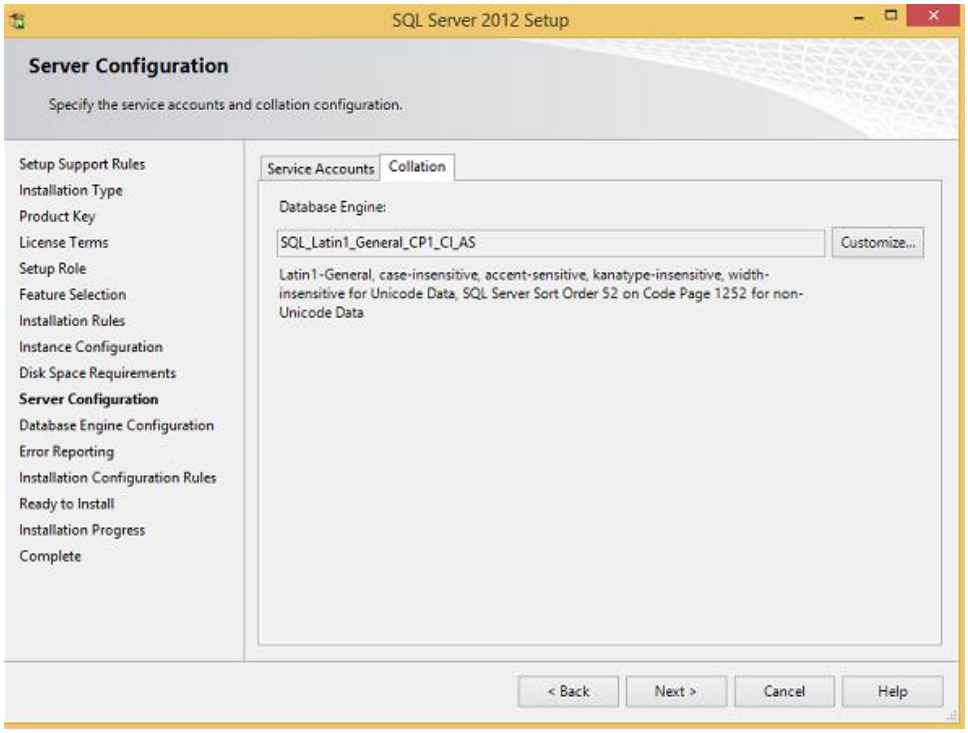
Disk Space

Step 14 – Click Next on the above screen and the following screen appears.



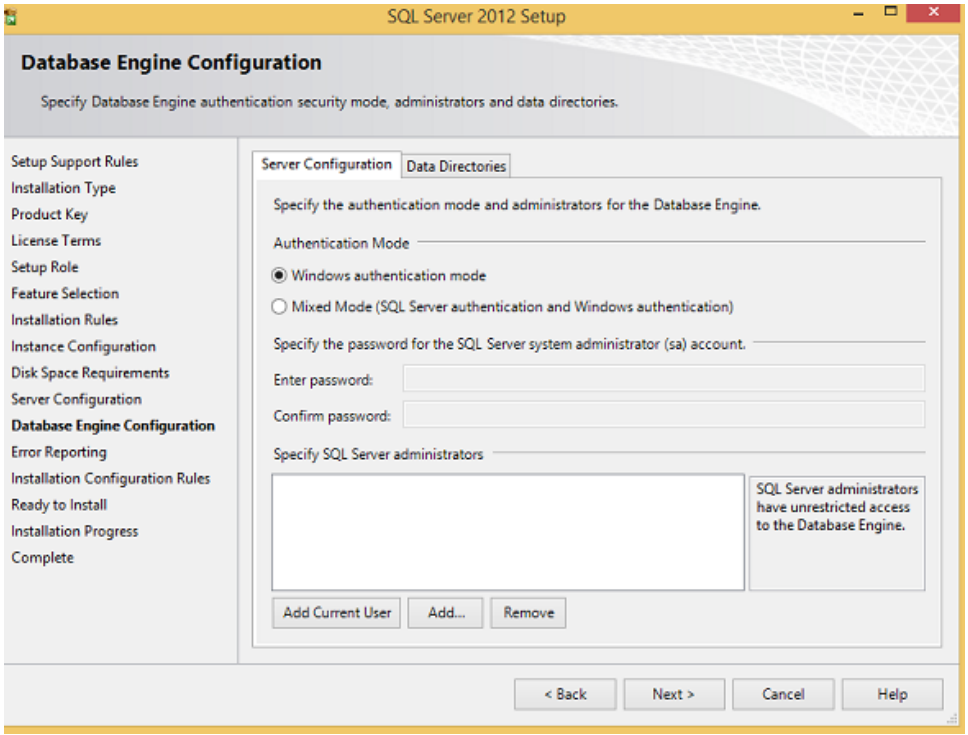
Server Configuration

Step 15 – Select service account names and start-up types for the above listed services and click Collation.



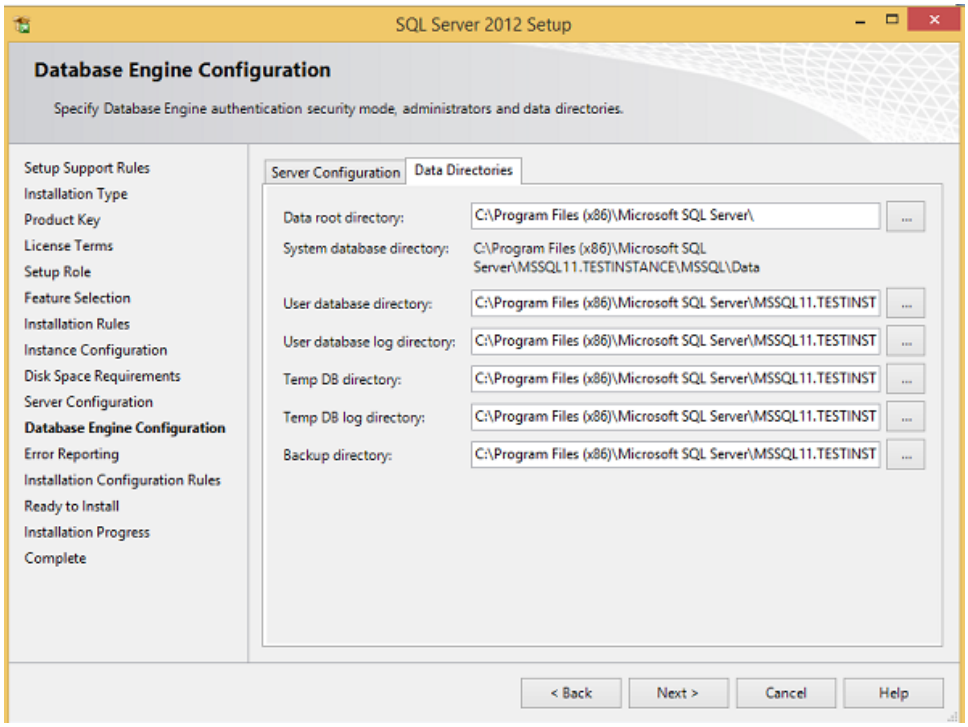
Configuration

Step 16 – Make sure the correct collation selection is checked and click Next.



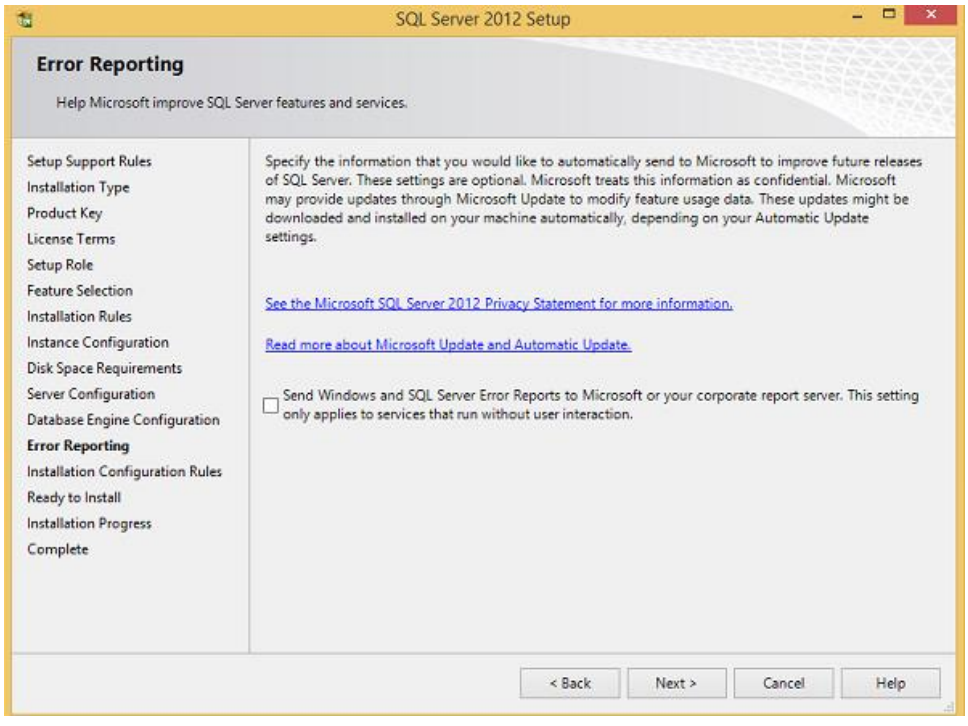
Database Engine

Step 17 – Make sure authentication mode selection and administrators are checked and click Data Directories



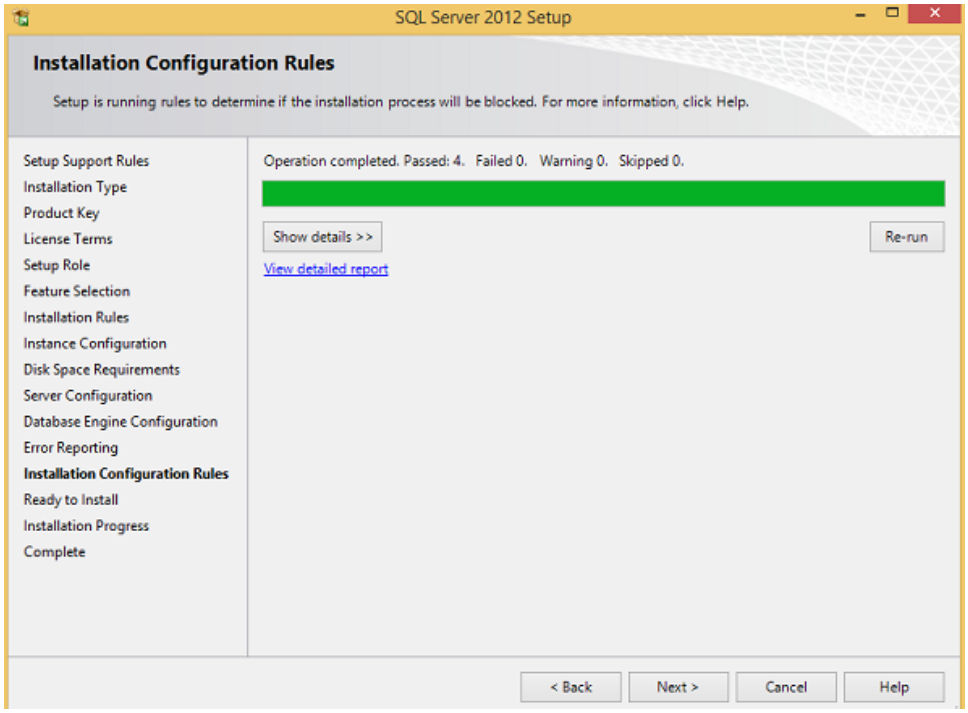
Database Configuration

Step 18 – Make sure to select the above directory locations and click Next. The following screen appears.



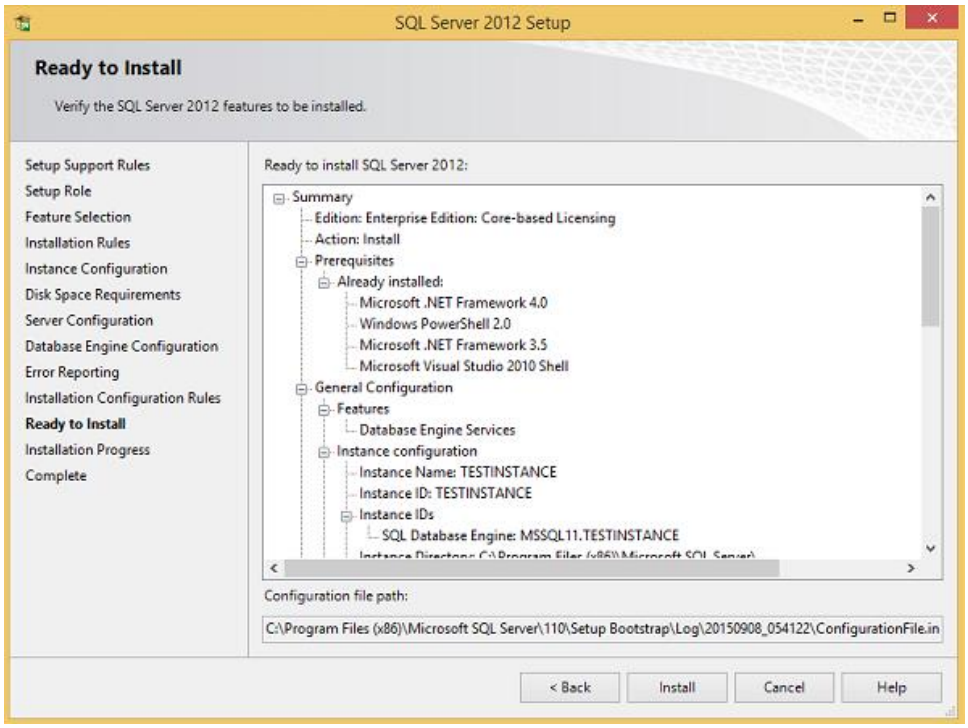
Error Reporting

Step 19 – Click Next on the above screen.



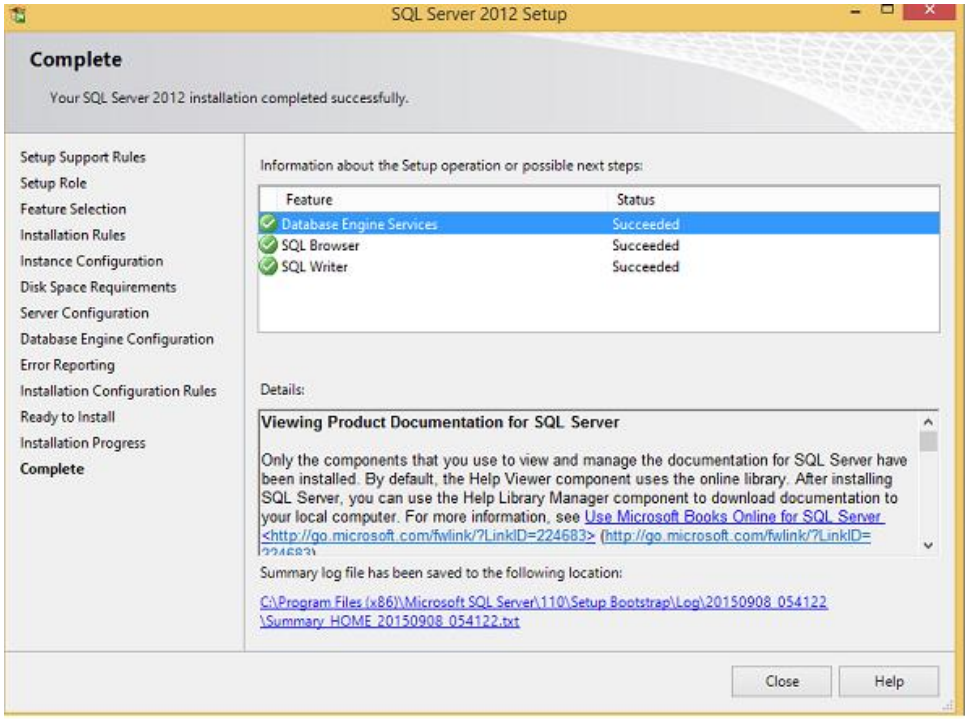
Installation Configuration

Step 20 – Click Next on the above screen to the get the following screen.



Ready To Install

Step 21 – Make sure to check the above selection correctly and click Install.



APACHE TOMCAT INSTALLATION

Apache Tomcat implements Java Servlet, JavaServer Pages (JSP), and the WebSockets Application Programming Interface (API). Essentially, it's a pure Java HTTP web server that enables Java code, and thus gives your website more cross-platform freedom than some of its alternatives.

A Java-centric web server can, therefore, be a valuable tool if you're developing a wide variety of projects. Apache Tomcat can help you expand your range so you can take on more clients and grow your career.

Plus, Tomcat is intuitive to set up as it comes with its own Windows Installer. However, there are several other routes you can take to get started, so you ultimately have control when it comes to how you run it. The element of user choice is also reflected in the suite of customization features it comes with.

Since it can run independently of Apache, it's a stable platform, too. Even if the rest of the server fails, Tomcat should still plug along.

That's certainly not all when it comes to Tomcat's advantages. As its code is open-source, there's a thriving online community that maintains it. Tapping into this network of developers will provide you with valuable resources and information.

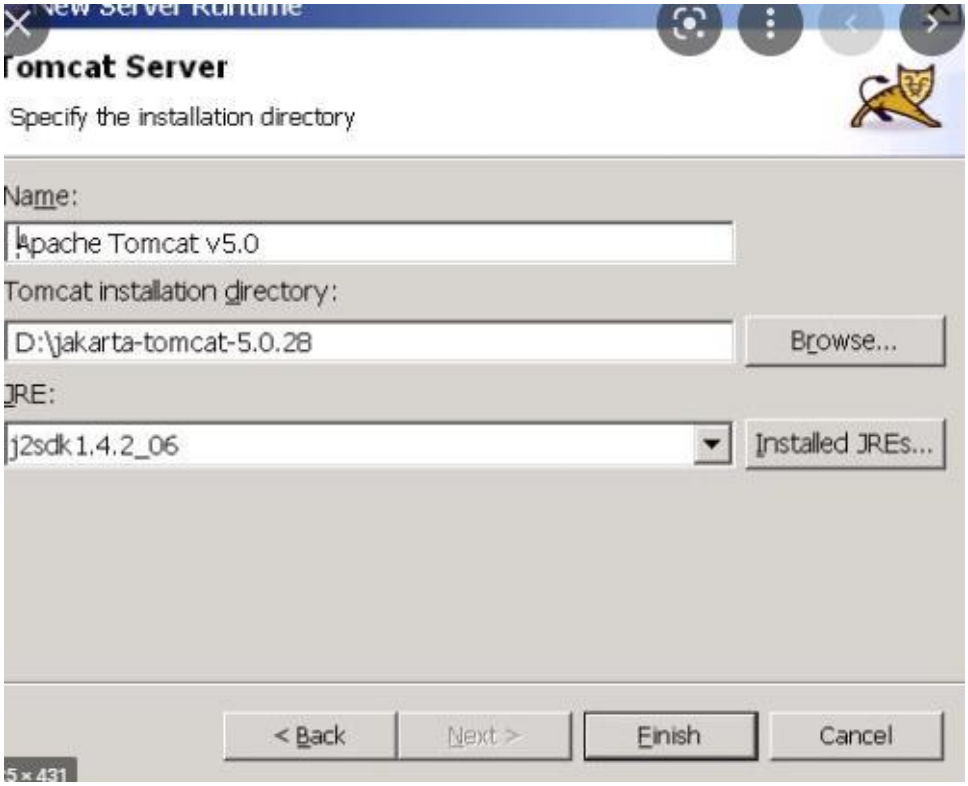
Alongside that, there are plenty of online tutorials to help you make a smooth transition to utilizing this platform. Indeed, the Apache Tomcat website alone offers a wealth of resources, on a range of topics from startup settings to security guides.

Lastly, Tomcat provides a server environment that's fast and has solid development time. Its reload and redeployment features let you update individual Java classes without having to restart the whole system, ultimately making your process more efficient.

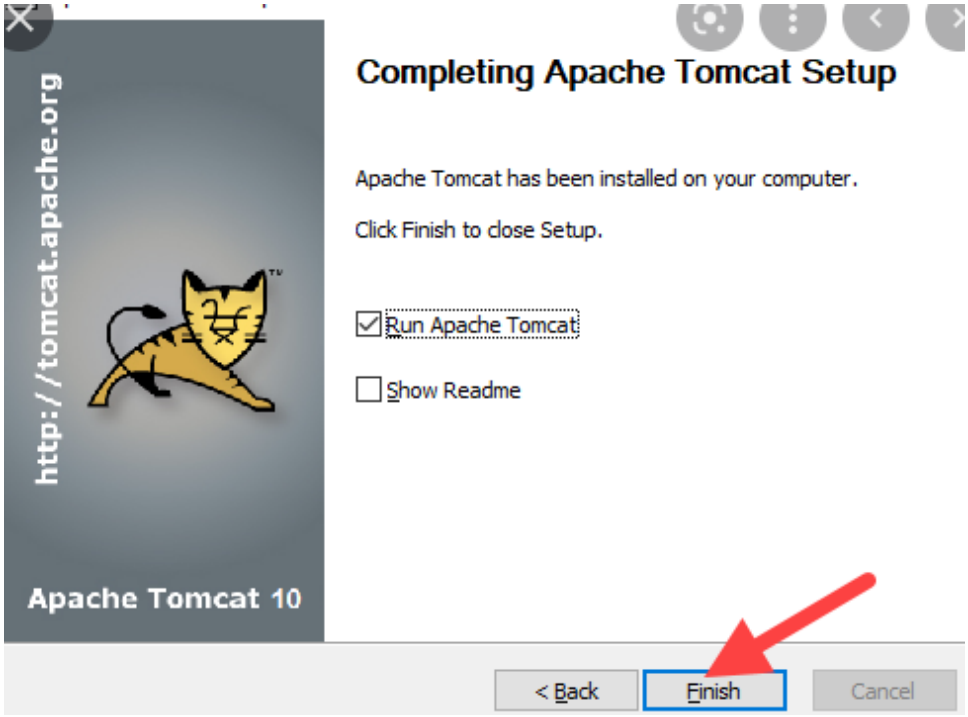
Getting started with Apache Tomcat is simple. However, before you begin, you need to install the Java Development Kit (JDK) if you don't already have it.

Once you have done so, you're ready to jump in. To start, head to the Apache Tomcat homepage and find the *Download* button.

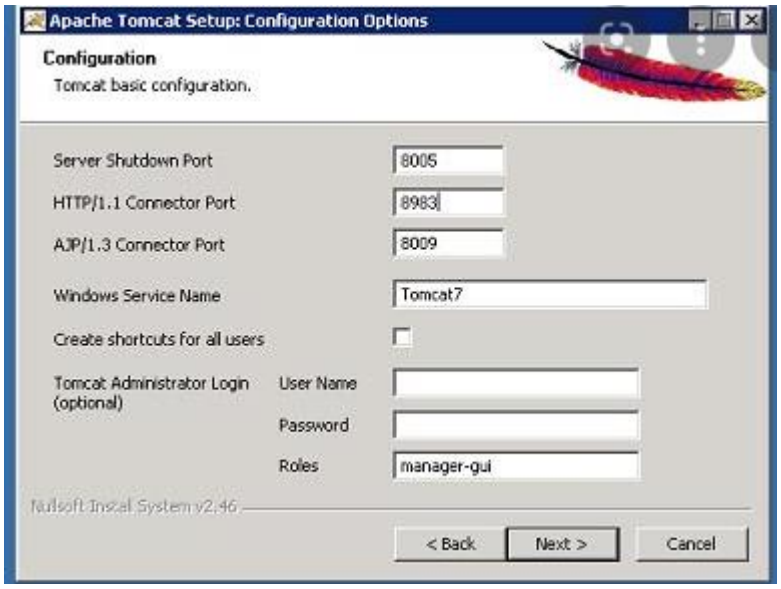
Go to the Tomcat Web page.



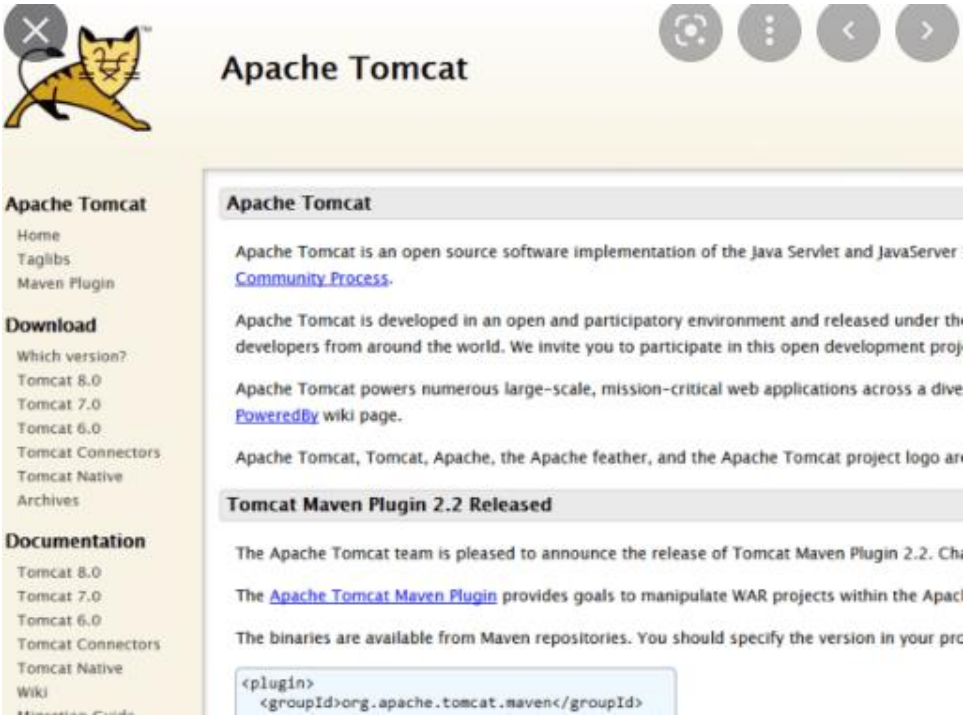
Click on Binaries under the Download label on the left side of the page.



Scroll down until you see Tomcat 4.1.x. (x will be some number greater than 10).



Click on the link ending with exe (e.g. 4.1.27 exe).

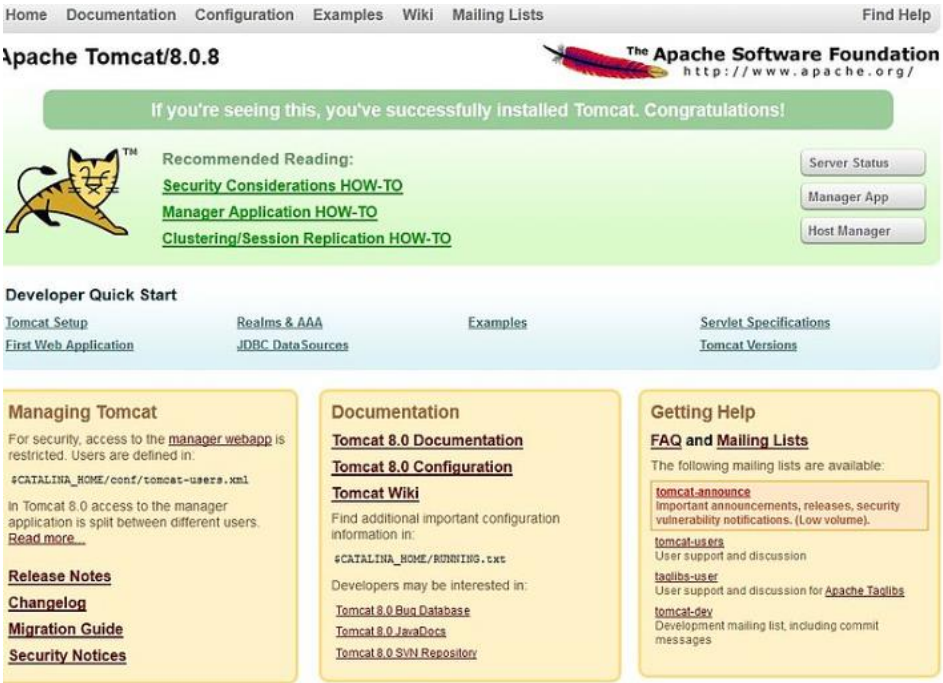


Download and run the exe file.

I suggest you install Tomcat at c:\tomcat4

Use the default settings and provide a password that you will remember.

now assume that your tomcat are installed at c:\tomcat4



JAVA INSTALLATION

Overview of Java

Java is a programming language originally developed by James Gosling at Sun Microsystems (now part of Oracle Corporation) and released in

1995 as a core component of Sun Microsystems' Java platform.Java applications are typically compiled to bytecode (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture.It is intended to let application developers "write once, run anywhere." Java is currently one of the most popular programming languages in use, particularly for client-server web applications.Java technology is both a programming language and a platform Independent.

Features of Java

Simple

Architecture neutral

Object oriented

Portable

Distributed

High performance

Interpreted

Multithreaded

Robust

Dynamic

Secure

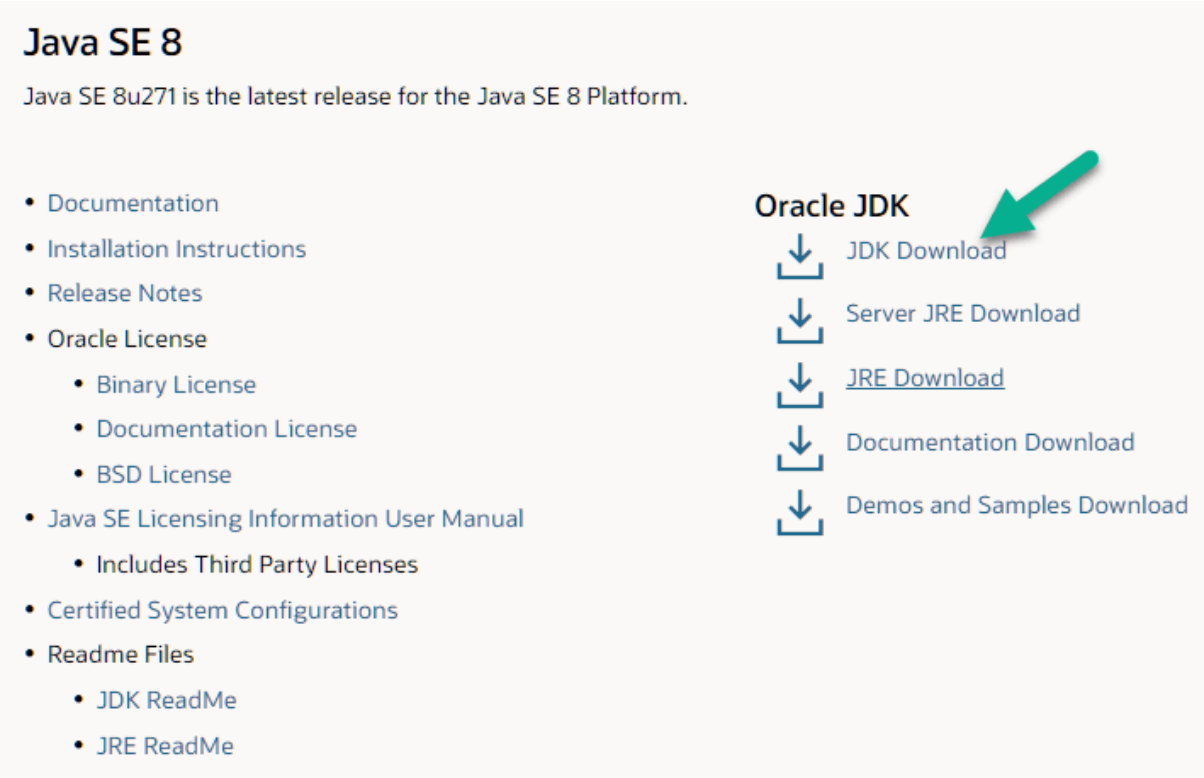
Java Virtual Machine (JVM)

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

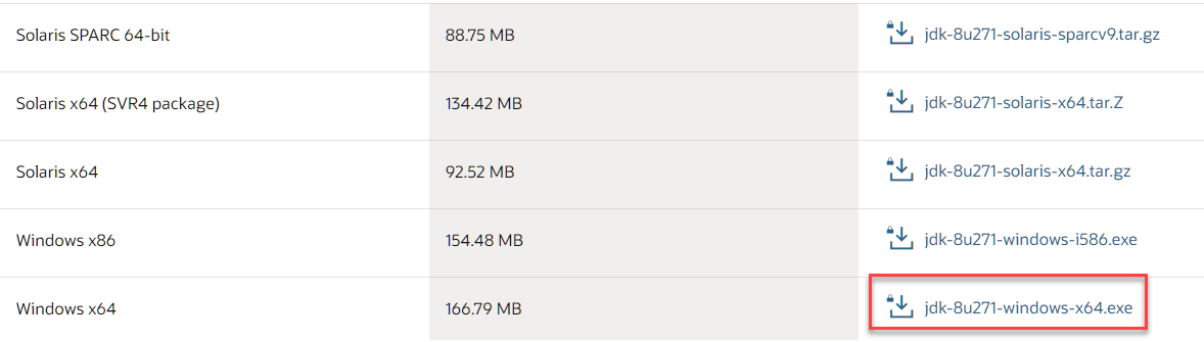
Following are the steps on how to install Java in Windows 10 for JDK 8 free download for 32 bit or JDK8 download for Windows 64 bit and installation

Step 1) Go to link. Click on JDK Download for Java download JDK 8.

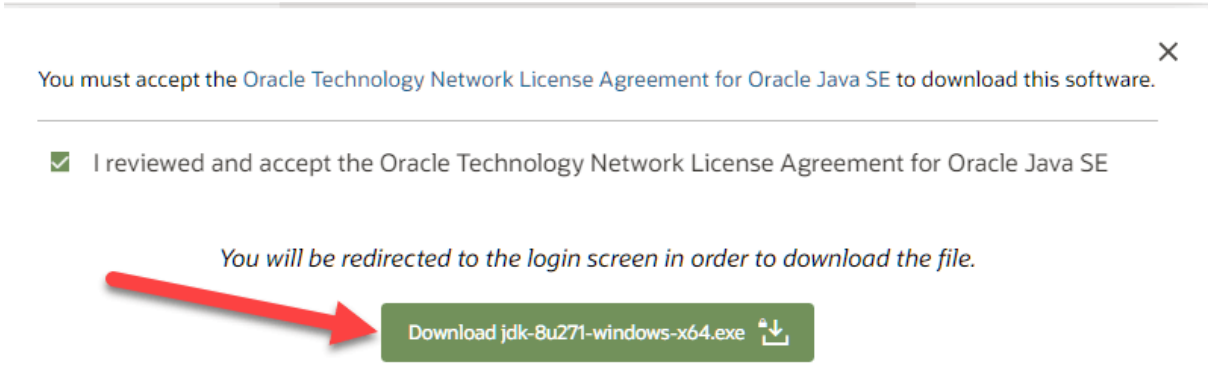


Step 2) Next,

1. Accept License Agreement
2. Download Java 8 JDK for your version 32 bit or JDK download 64 bit.

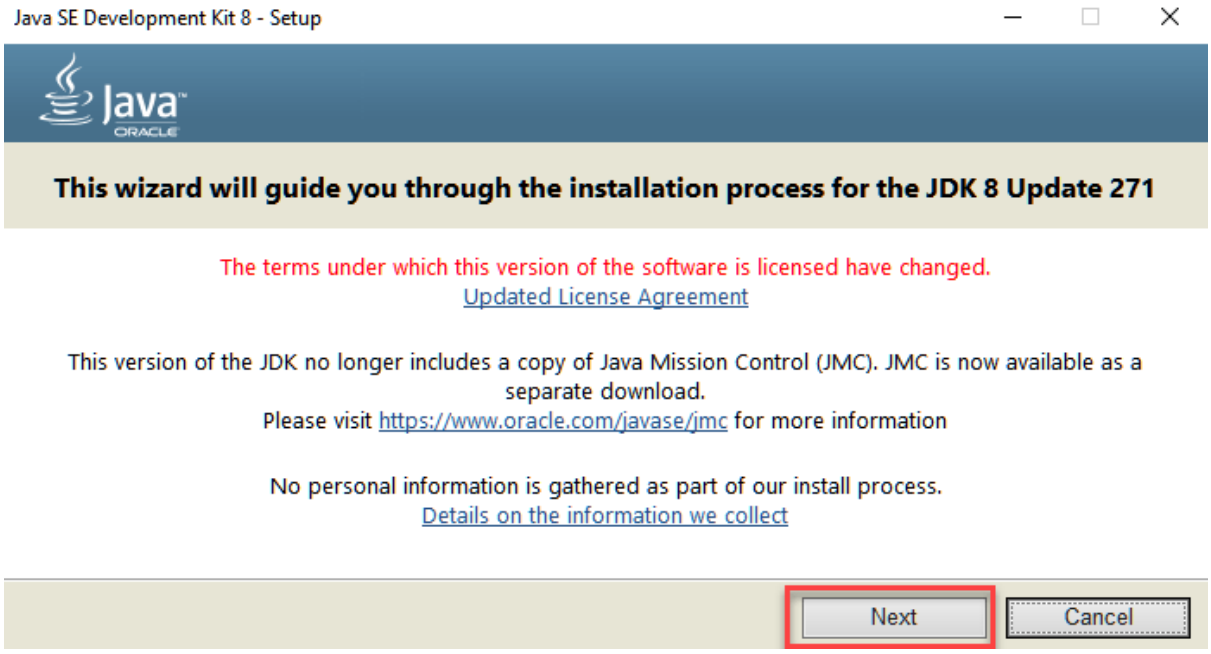


Step 3) When you click on the Installation link the popup will be open. Click on I reviewed and accept the Oracle Technology Network License Agreement for Oracle Java SE development kit and you will be redirected to the login page. If you don't have an oracle account you can easily sign up by adding basics details of yours.

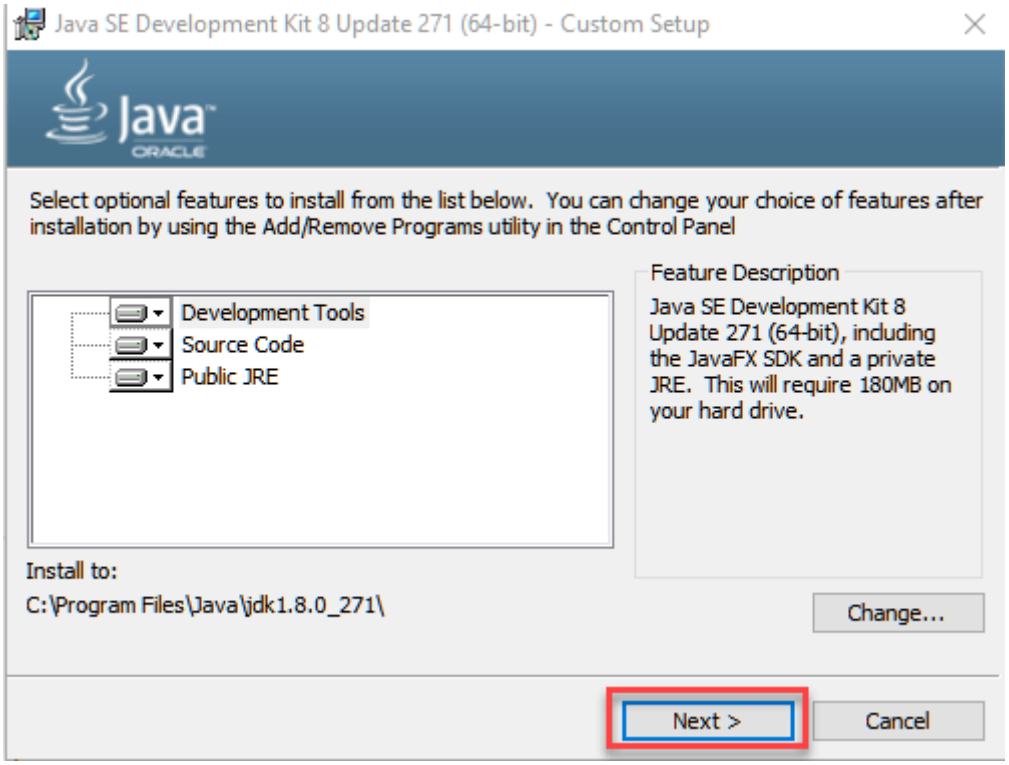


NOTE: You will be required to create an Oracle Account to start Java 8 download of the file.

Step 4) Once the Java JDK 8 download is complete, run the exe for install JDK. Click Next



Step 5) Select the PATH to install Java in Windows… You can leave it Default. Click next.



NOTE: Follow the onscreen instructions in succeeding steps to install Java 8 on Windows 10.

Step 6) Once you install Java in windows, click Close

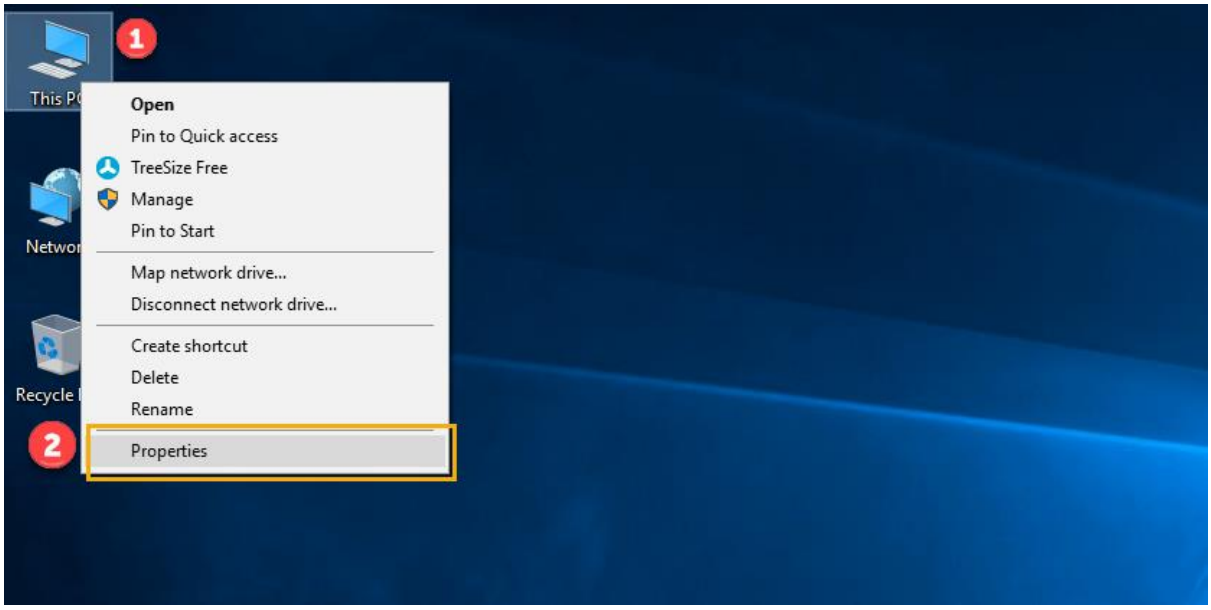
How to set Environment Variables in Java: Path and Classpath

The PATH variable gives the location of executables like javac, java etc. It is possible to run a program without specifying the PATH but you will need to give full path of executable like **C:\Program Files\Java\jdk1.8.0_271\bin\javac A.java** instead of simple **javac A.java**

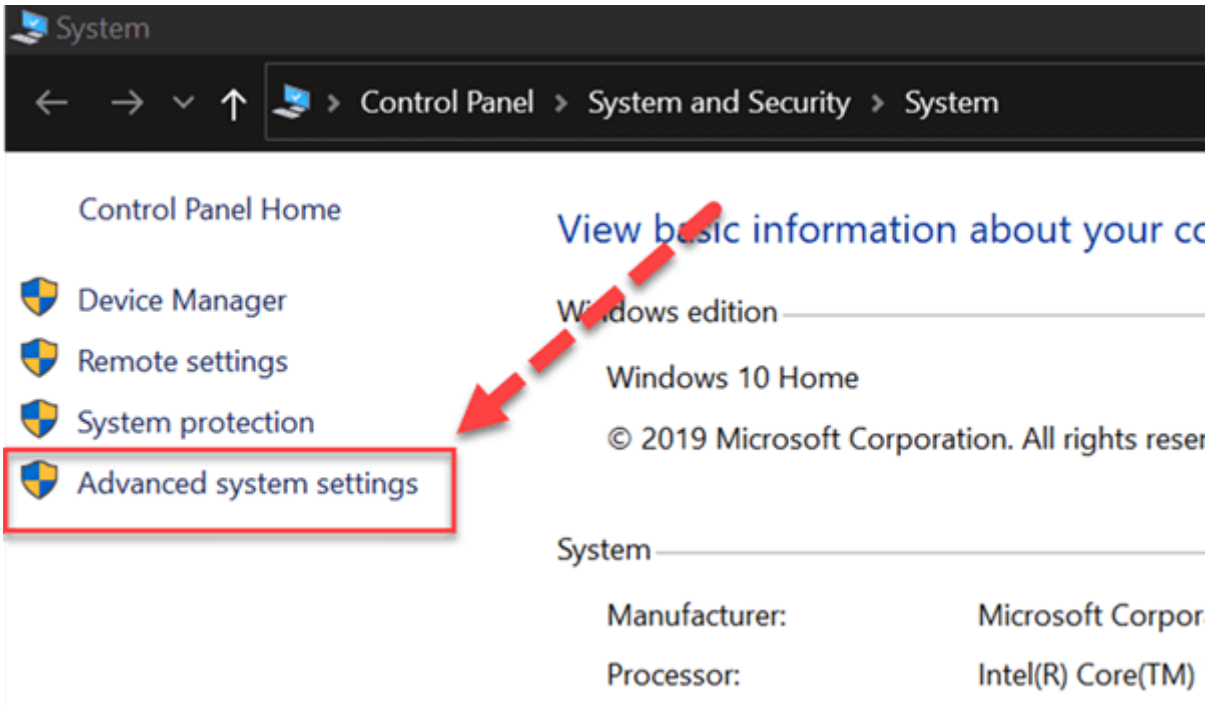
The CLASSPATH variable gives location of the Library Files.

Let's look into the steps to set the PATH and CLASSPATH

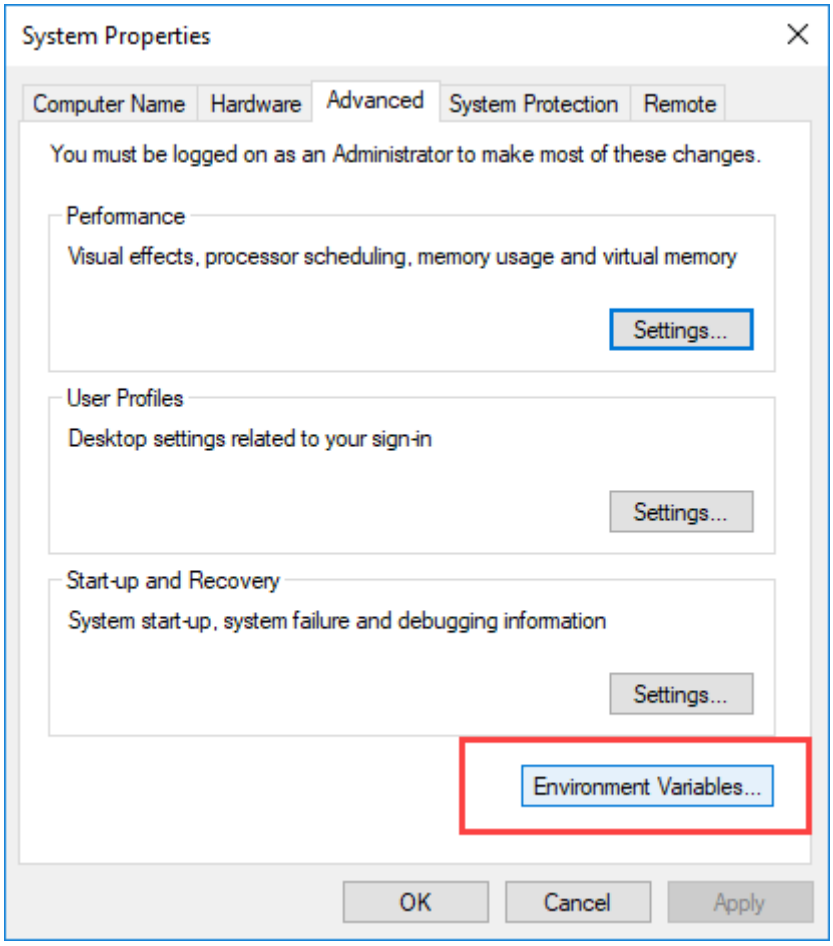
Step 1) Right Click on the My Computer and Select the properties



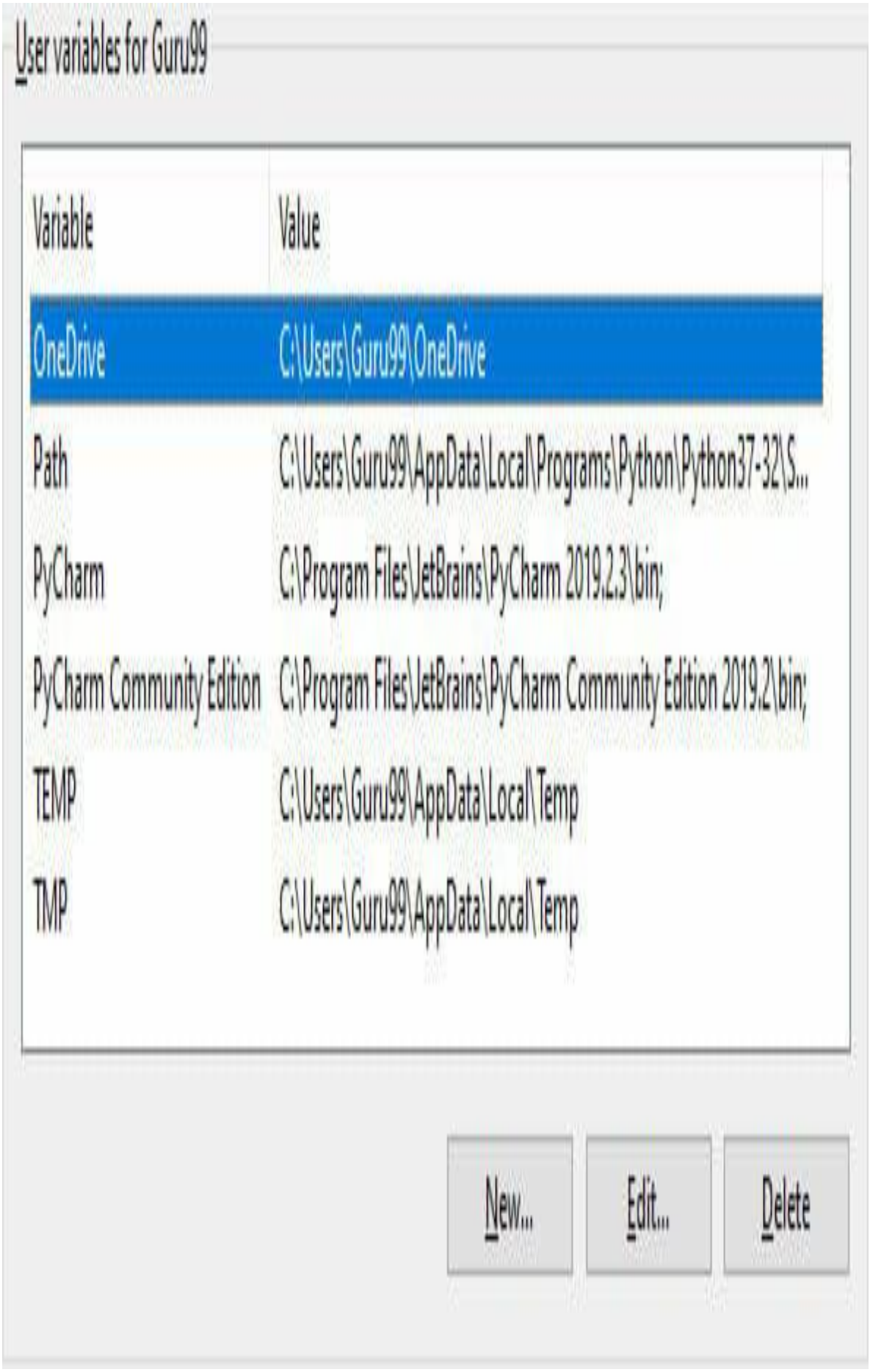
Step 2) Click on advanced system settings



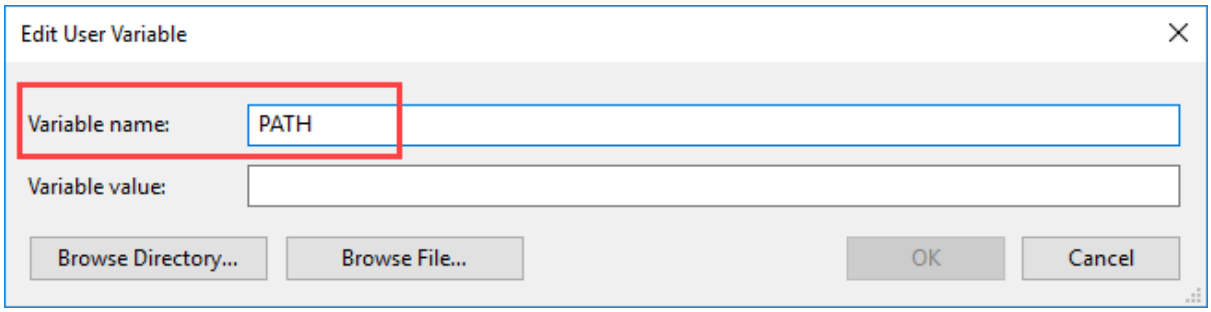
Step 3) Click on Environment Variables to set Java runtime environment



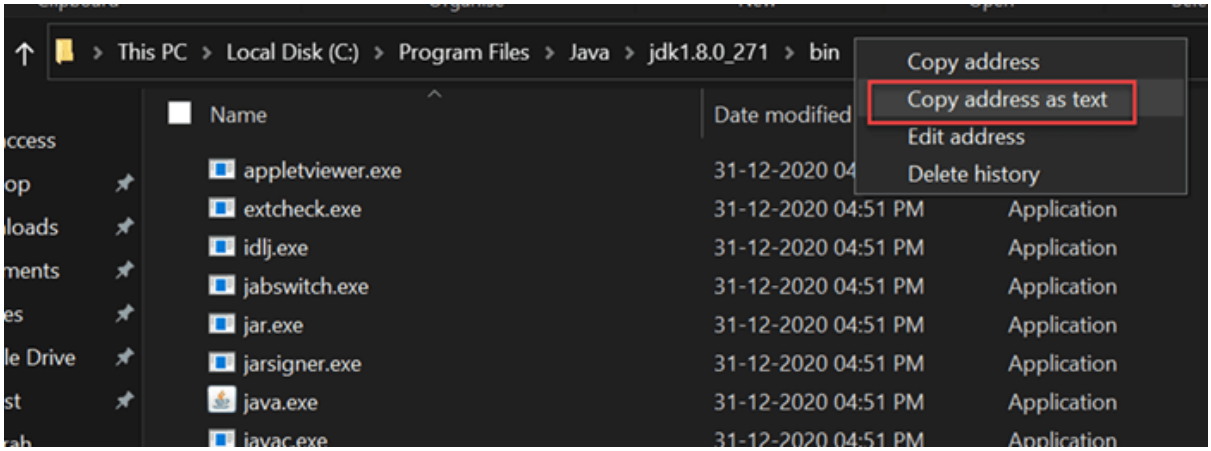
Step 4) Click on new Button of User variables


User variables for Guru99
Variable
Value
OneDrive
C:\Users\Guru99\OneDrive
Path
C:\Users\Guru99\AppData\Local\Programs\Python\Python37-32\S...
PyCharm
C:\Program Files\JetBrains\PyCharm 2019.2.3\bin;
PyCharm Community Edition
C:\Program Files\JetBrains\PyCharm Community Edition 2019.2\bin;
TEMP
C:\Users\Guru99\AppData\Local\Temp
TMP
C:\Users\Guru99\AppData\Local\Temp
New...
Edit...
Delete

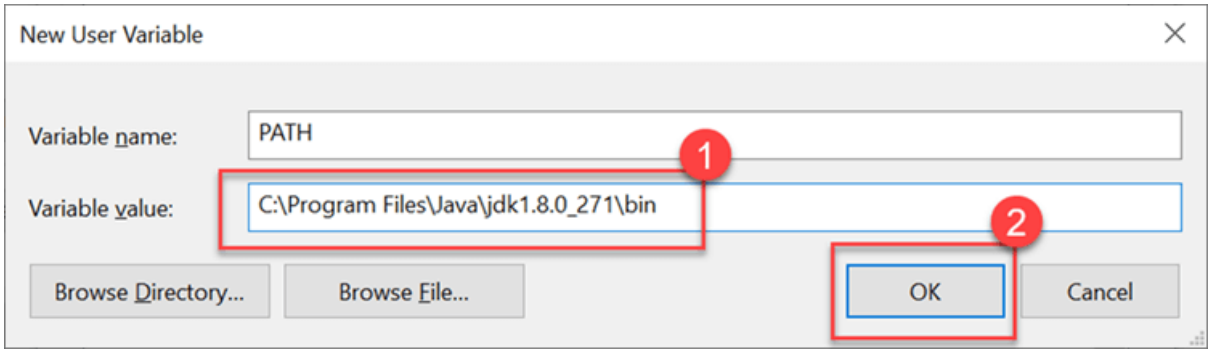
Step 5) Type PATH in the Variable name.



Step 6) Copy the path of bin folder which is installed in JDK folder.



Step 7) Paste Path of bin folder in Variable value. Click on OK Button.

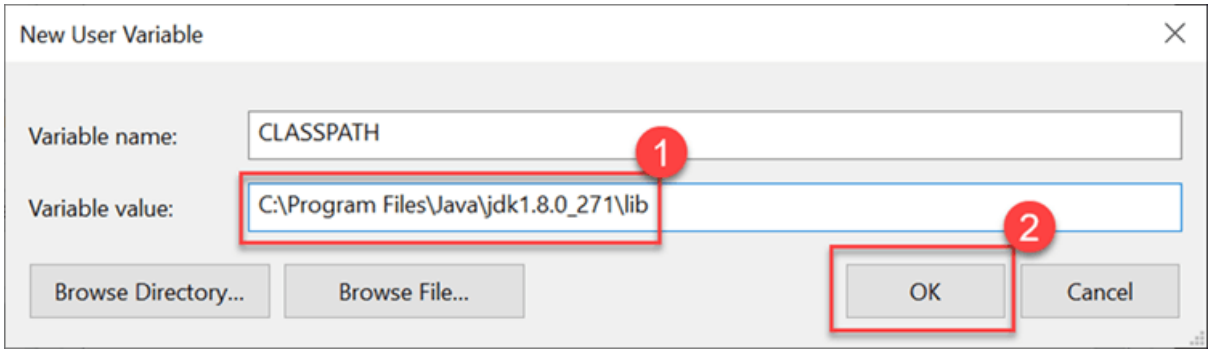


Note: In case you already have a PATH variable created in your PC, edit the PATH variable to

PATH = <JDK installation directory>\bin;%PATH%;

Here, %PATH% appends the existing path variable to our new value

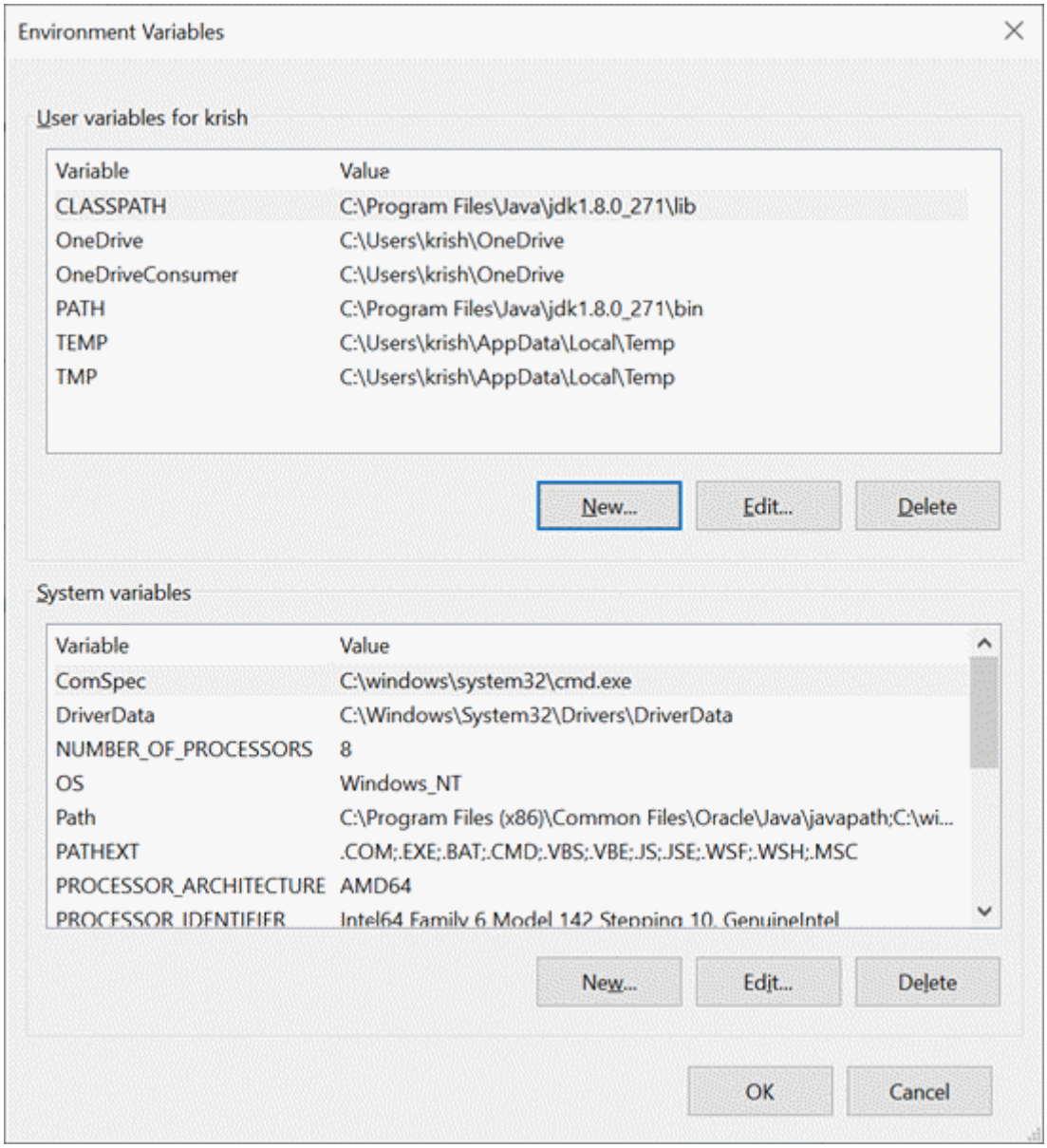
Step 8) You can follow a similar process to set CLASSPATH.



Note: In case you java installation does not work after installation, change classpath to

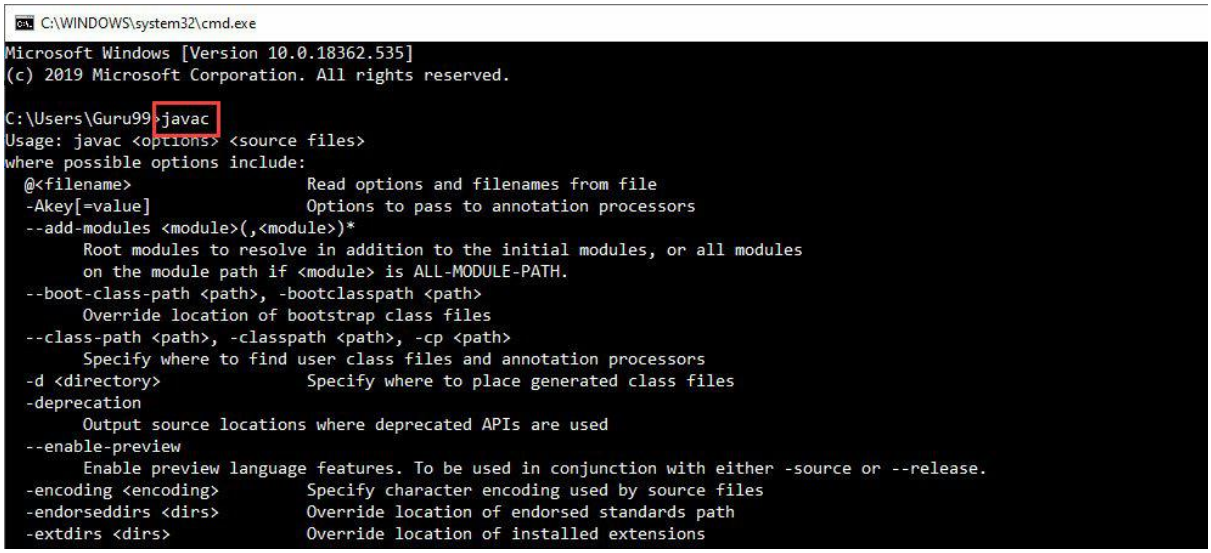
CLASSPATH = <JDK installation directory>\lib\tools.jar;

Step 9) Click on OK button



Step 10) Go to command prompt and type javac commands.

If you see a screen like below, Java is installed.



APPENDIX C – SOFTWARE USAGE MANUAL

As the SLA agreement is not transparent to the users, there comes the need to have auditing to check for SLA violation. There are two types of auditing depending upon which is being audited: Internal Audit and External Audit. Internal

Audit audits the processes that takes place in providing the service. External Audit audits the quality of service such as CPU performance, availability and SLA parameters.

Audit can be both static and dynamic. In static auditing , auditing is done periodically to verify the integrity of data. samples are taken from the data and it is verified for integrity of data. In dynamic auditing, auditing is done on dynamic data. The dynamic data operations are modification, insertion and deletion. Batch auditing is required when there is multiple owner and multiple cloud servers.The TPA process works in three steps: Key Generation , Server integrity proof, integrity verification.

Steps to create an account in DriveHQ:

You can easily sign up a new DriveHQ account on DriveHQ.com website, or from any of our client software/mobile apps. From our website, click "Sign Up", then enter your username, password, email address and optionally more user info, agree to the "Membership Agreement".