**A Scalable Two-layer Blockchain System for Distributed Multi-distributed storage.**

**Abstract:**

In this application, we are carrying out cryptography strategies which includes applying a technique called a calculation to plain text to transform it into something that will give off an impression of being babble to any individual who doesn't have the way to unscramble it. Here the client ought to enroll their subtleties and can login, henceforth they can ready to see the rundown of document put away in this application and the substance of the record can't see the client since it would be encoded when the record is transferred in this application. The client can transfer their intrigued record and can download with the entrance of administrator. While transferring the document is changed over completely to non intelligible code and the key too. The administrator is keep up with the subtleties of downloader and the uploader subtleties. Then the client can download the document of who intrigued with the assistance of decoded key.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **LIST OF FIGURES**  **LIST OF SYMBOLS**  **LIST OF ABBREVIATIONS**  **LIST OF TABLES** | i  v  vii  xi  xii |
| 1. | **CHAPTER 1 : INTRODUCTION**   * 1. INTRODUCTION   2. EXISTING SYSTEM   1.2.1 LITERATURE SURVEY  1.3 PROPOSED SYSTEM |  |
| 2. | **CHAPTER 2 :PROJECT DESCRIPTION**  2.1 GENERAL  2.2 MODULES  2.3 MODULES EXPLANATION  2.4 TECHNIQUE OR ALGORITHM |  |
| 3. | **CHAPTER 3 : REQUIREMENTS**  3.1 General  3.2 Hardware REQUIREMENTS  3.3 Software REQUIREMENTS |  |
| 4. | CHAPTER 4 : SYSTEM DESIGN  4.1 general  4.1.1 USECASE DIAGRAM:  4.1.2 CLASS DIAGRAM:  4.1.3 OBJECT DIAGRAM  4.1.4 STATE DIAGRAM:  4.1.5 ACTIVITY DIAGRAM  4.1.6 SEQUENCE DIAGRAM  4.1.7 COLLABORATION DIAGRAM  4.1.8 COMPONENT DIAGRAM  4.1.9 DATAFLOW DIAGRAM  4.1.10 E-R DIAGRAM  4.1.11 SYSTEM ARCHITECTURE |  |
| 5. | **CHAPTER 5 :SOFTWARE SPECIFICATION**  5.1 general  FRONT END  5.2 features of java  5.2.1 the java framework  5.2.2 objective of java  5.2.3 JAVA SERVER PAGES  5.2.4 EVOLUTION OF WEB APPLICATIONS  5.2.5 BENEFITS OF JSP  5.3 SERVLETS  5.4 JAVA SERVLETS  5.5 CONCLUSION |  |
| 6. | **CHAPTER 6 : IMPLEMENTATION**  6.1 GENERAL  6.2 IMPLEMENTATION |  |

|  |  |  |
| --- | --- | --- |
| 7. | **CHAPTER 7 : SNAPSHOTS**  7.1 GENERAL  7.2 VARIOUS SNAPSHOTS |  |
| 8. | CHAPTER 8 : SOFTWARE TESTING  8.1. FEASIBILITY STUDY  8.1.1. ECONOMICAL FEASIBILITY  8.1.2. TECHNICAL FEASIBILITY  8.1.3. OPERATIONAL FEASIBILITY  8.2. SYSTEM TESTING  8.2.1. VARIOUS LEVELS OF TESTING  8.2.1.1. WHITE BOX TESTING  8.2.1.2. BLACK BOX TESTING  8.2.1.3. UNIT TESTING  8.2.1.4. FUNCTIONAL TESTING  8.2.1.5. PERFORMANCE TESTING  8.2.1.6. INTEGRATION TESTING  8.2.1.7. VALIDATION TESTING  8.2.1.8. SYSTEM TESTING  8.2.1.9. OUTPUT TESTING  8.2.1.10. USER ACCEPTANCE TESTING |  |
| 9. | **CHAPTER 9 :**  **APPLICATIONS AND FUTURE ENHANCEMENT**  9.1 FUTURE ENHANCEMENTS |  |
| **10** | **CHAPTER 10 :**  CONCLUSION  REFERENCES |  |
|  | **GANTT CHART** |  |
|  |  |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME OF THE FIGURE** | **PAGE NO.** |
| 2.3.2 | Module Diagram |  |
| 4.2 | Activity Diagram for Anonymous database Management |  |
| 4.3 | Use case Diagram for Anonymous database Management |  |
| 4.4 | Data flow diagram for Anonymous database Management |  |
| 4.5 | Sequence diagram for Anonymous database Management |  |
| 4.6 | Collaboration diagram for Anonymous database Management. |  |
| 4.7 | Class diagram for Anonymous database Management |  |
| 4.8 | Architecture Diagram for Anonymous database Management |  |
| 4.9 | Object Diagram |  |
| 4.10 | State Diagram |  |
| 4.11 | Component Diagram |  |
| 4.12 | E-R Diagram |  |
| 7.1 | Medical home page |  |
| 7.2 | Patient Registration Page |  |
| 7.3 | Doctor Registration page |  |
| 7.4 | Patient get Doctor Appointment Page |  |
| 7.5 | Doctor Appointment Page |  |

**LIST OF SYSMBOLS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **NOTATION**  **NAME** | **NOTATION** | **DESCRIPTION** |
| 1. | Class | *Class Name*  *-attribute*  *-attribute*  *+operation*  *+operation*  *+operation*  *+ public*  *-private*  *# protected* | Represents a collection of similar entities grouped together. |
| 2. | Association | name  Class B  Class A  Class A  Class B | Associations represents static relationships between classes. Roles representsthe way the two classes see each other. |
| 3. | Actor | Class A  Class A  Class B  Class B | It aggregates several classes into a single classes. |
| 4. | Aggregation | Interaction between the system and external environment |

|  |  |  |  |
| --- | --- | --- | --- |
| 5. | Relation  (uses) | uses | Used for additional process communication. |
| 6. | Relation  (extends) | extends | Extends relationship is used when one use case is similar to another use case but does a bit more. |
| 7. | Communication |  | Communication between various use cases. |
| 8. | State | State | State of the processs. |
| 9. | Initial State |  | Initial state of the object |
| 10. | Final state |  | F inal state of the object |
| 11. | Control flow |  | Represents various control flow between the states. |
| 12. | Decision box |  | Represents decision making process from a constraint |
| 13. | Usecase |  | Interact ion between the system and external environment. |

|  |  |  |  |
| --- | --- | --- | --- |
| 14. | Component |  | Represents physical modules which is a collection of components. |
| 15. | Node |  | Represents physical modules which are a collection of components. |
| 16. | Data Process/State |  | A circle in DFD represents a state or process which has been triggered due to some event or acion. |
| 17. | External entity |  | Represents external entities such as keyboard,sensors,etc. |
| 18. | Transition |  | Represents communication that occurs between processes. |
| 19. | Object Lifeline |  | Represents the vertical dimensions that the object communications. |
| 20. | Message | Message | Represents the message exchanged. |

**LIST OF ABBREVATION**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **ABBREVATION** | **EXPANSION** |
| 1**.** | DB | DataBase |
| 2. | SMC | Secure MultipartyComputation |
| 3. | MDA | Medical Admin |
| 4. | DBC | Data Base Confidentiality |
| 10 | JVM | Java Virtual Machine |
| 11. | JSP | Java Server Page |

**Introduction:**

Here the client should select their nuances and can login, consequently they can prepared to see the overview of archive set aside in this application and the substance of the record can't see the client since it would be encoded when the record is moved in this application. The client can move their charmed record and can download with the entry of overseer. While moving the record is changed over totally to non clear code and the key as well. The executive is stay aware of the nuances of downloader and the uploader nuances. Then, at that point, the client can download the report of who interested with the help of decoded key.

**1.2.1 LITERATURE SURVEY:**

# **TITLE 1:** Coded Caching for Time-Varying Files Popularities and Asynchronous Delivery

**AUTHOR:** [Mohamed Amir](https://ieeexplore.ieee.org/author/38275847400), [Ebrahim Bedeer](https://ieeexplore.ieee.org/author/37713883200), [Mohamed Hossam Ahmed](https://ieeexplore.ieee.org/author/37280827400), [Tamer Khattab](https://ieeexplore.ieee.org/author/37282020000).

**YEAR:**  2021

**DESCRIPTION:**

Data communication has seen exponential growth recently, and it currently dominates wireless communication. As a result, proactive caching was developed to minimize peak traffic rates by storing content, in advance, at different nodes in the network. We consider proactive caching for a broadcast wireless network with one central hub such as a satellite (ST) and K associated mobile units (MUs) such as mobile mini-ground stations or end users. The ST has a library of files, and the MUs demands are assumed to be limited to this library, while the popularity of the library files changes over time. We assume that the MUs demands arrive at different times, and hence, asynchronous file delivery is necessary. We propose a new scheme that minimizes the files delivery sum rate and show that we can use the file delivery messages to proactively and constantly update the MU finite caches. We show that this mechanism reduces the downloaded traffic of the network. The proposed scheme uses index coding to jointly encode the delivery of different demanded files with the cache updates to other MUs to follow the changes in the files popularities. An offline optimization of the delivery sum rate of the scheme is proposed, where it requires knowledge of the files popularities across the whole transmission period.

# **TITLE 2:** A Secure Communication System in Self-Organizing Networks via Lightweight Group Key Generation

**AUTHOR:**

[Sirui Peng](https://ieeexplore.ieee.org/author/37087407198), [Biao Han](https://ieeexplore.ieee.org/author/37065857300), [Celimuge Wu](https://ieeexplore.ieee.org/author/37673017100), [Baosheng Wang](https://ieeexplore.ieee.org/author/37292888500)

**YEAR:**  2020

**DESCRIPTION:**

There is no central node in self-organizing networks, and therefore the traditional key distribution methods based on public infrastructure do not work. The standardized pre-shared keys have predictable security risks. The physical-layer secret key generation has become a technology worth considering due to its lightweight, security, and decentralization. However, most of the previous work has focused on two devices, and remains a challenge to expand the pairwise key into the group key. Since the channel reciprocity only exists between two devices, some information would be exchanged on the unencrypted channel, causing information leakage. This paper designs a secure communication system in self-organizing networks. It adopts an adaptive quantizer to generate the pairwise keys and proposes DORCE, Difference Of quantization Results at one deviCE. The authenticated users share the group key via the difference between pairwise keys. The algorithm is implemented in a mesh topology, which is suitable for self-organizing networks because users' joining and leaving will not have a great impact on the network topology. The algorithm's Key Achievable Rate is up to 4 bits. Experimental results demonstrate that DORCE can generate the group keys in seconds.

# **TITLE 3:** Adaptive and Optimum Secret Key Establishment for Secure Vehicular Communications

**AUTHOR:**

[Sirui Peng](https://ieeexplore.ieee.org/author/37087407198), [Biao Han](https://ieeexplore.ieee.org/author/37065857300), [Celimuge Wu](https://ieeexplore.ieee.org/author/37673017100), [Baosheng Wang](https://ieeexplore.ieee.org/author/37292888500)

**YEAR:**  2020

**DESCRIPTION:**

In intelligent transportation systems (ITS), communications between vehicles, i.e. vehicle-to-vehicle (V2V) communications are of greatest importance to facilitate autonomous driving. The current state-of-the-art for secure data exchange in V2V communications relies on public-key cryptography (PKC) consuming significant computational and energy resources for the encryption/decryption process and large bandwidth for the key distribution. To overcome these limitations, physical-layer security (PLS) has emerged as a lightweight solution by exploiting the physical characteristics of the V2V communication channel to generate symmetric cryptographic keys. Currently, key-generation algorithms are designed via empirical parameter settings, without resulting in optimum key-generation performance. In this paper, we devise a key-generation algorithm for PLS in V2V communications by introducing a novel channel response quantisation method that results in optimum performance via analytical parameter settings. Contrary to the current state-of-the-art, the channel responses incorporate all V2V channel attributes that contribute to temporal variability, such as three dimensional (3D) scattering and scatterers' mobility. An extra functionality, namely, Perturbe-Observe (PO), is further incorporated that enables the algorithm to adapt to the inherent non-reciprocity of the V2V channel responses at the legitimate entities. Optimum performance is evidenced via maximisation of the key bit generation rate (BGR) and key entropy (H) and minimisation of the key bit mismatch rate (BMR).

# **TITLE 4:** Identifying library functions in executable file using patterns

**AUTHOR:**

[M. Van Emmerik](https://ieeexplore.ieee.org/author/37284153100), [Celimuge Wu](https://ieeexplore.ieee.org/author/37673017100).

**YEAR:**  2002

**DESCRIPTION:**

Re-engineering from legacy executable (binary) files is greatly facilitated by identifying and naming statically linked library functions. This paper presents an efficient method for generating files of patterns; each pattern is a transformation of the first several bytes of a library function's executable code. Given a suitable pattern file, a candidate function can be identified in linear time. One pattern file is generated for each combination of compiler vendor, version and memory model (where applicable). The process of identifying these parameters in a given executable file also identifies the main function of the program, i.e. the start of the code written by the user. The pattern files are produced automatically from a compiler's library file in a few seconds, with no user intervention required. Due to various limitations, not all library functions can be identified correctly; a small number will be either incorrectly identified or not identified.

**Proposed System**:

**Description**:

By using this application the client can prepared to move their nuances and download from it with the help of unscrambled key after the upheld by the overseer.

**Techniques**:

RSA algorithm, SQL operation

**Advantages**:

Carrying out RSA algorithm is extremely simple. RSA calculation is free from any potential harm for communicating private information. Breaking RSA calculation is extremely challenging as it includes complex arithmetic.

**CHAPTER 2**

**2.1 GENERAL**

Thus, there search database used by the doctor is anonymous. Suppose that certain data concerning visitors are related to the use of a drug over a period of four years and certain side effects have been observed and recorded by the doctors in the doctor’s database. It is clear that these data (even if anonymized) need to be kept confidential and accessible only to the few specialist of the institution working on this project, until further evidence is found about the drug. If these anonymous data were to be disclosed, privacy of the visitors would not be at risk; however the company manufacturing the drug may be adversely affected.

**2.2 MODULES:**

**MODULE NAMES:**

* LOGIN
* UPDATE REPORT
* FORWORD REPORT
* ADD FILE
* ADMIN DATA VIEW
* CATEGORY VIEW
* ACCEPT RESPONSE

**2.3 MODULE DESCRIPTION:**

**1. LOGIN:**

This is the first module in our project, here symbolizes a unit of work performed within a database management system (or similar system) against a database, and treated in a coherent and reliable way independent of other transactions. A transaction generally represents any change in database user will transfer the amount to provider.

**2. UPDATE REPORT:**

In this module is used to help to the user to update the report with the land longitude and the user will update the report along with their opinion and the report will be stored the database.

**3. FORWORD REPORT:**

In this module is used to help to the NGO to send the user report to admin with the land longitude and the will update the report along with their opinion and the will be stored the database.

**4. ADMIN DATA VIEW:**

In this module the admin will also view the data file. And analysis the Admin will be responsible for your file stored in database.

**5. CATEGORY VIEW:**

**I**n this module the admin will also view the data file fully analysed data in category wise view. Admin will be responsible for your file stored in database.

**6. ACCEPT THE REQUEST:**

The client need to demand to their intrigued record information to the administrator. Once The administrator has acknowledged the solicitation of client they can recover the information from data base.

**MODULE DIAGRAM:**

**LOGIN:**

MAINPAGE

USER

LOGIN

**UPDATE DATA:**

DATA BASE

USER

UPDATE

**FORWORD REPORT:**

ADMIN

DATABASE

FORWORD

FILE VIEW

**CATEGORY VIEW:**

VIEW BY CATEGORY

ADMIN

ANALYS

**ADD FILE:**

DATABASE

FILE UPLOAD

GENERATE KEY

**2.4 ALGORITHM USED:**

RSA encrypts messages through the following algorithm   
  
**1. Key Generation**

I. Choose two distinct prime numbers p and q.

II. Find n such that n = pq.

n will be used as the modulus for both the public and private keys.

III. Find the totient of n, ϕ(n)ϕ(n)=(p-1)(q-1).IV. Choose an e such that 1 < e < ϕ(n), and such that e and ϕ(n) share no divisors other than 1 (e and ϕ(n) are relatively prime).e is kept as the public key exponent.V. Determine d (using modular arithmetic) which satisfies the congruence relation  
  
de ≡ 1 (mod ϕ(n)).In other words, pick d such that de - 1 can be evenly divided by (p-1)(q-1), the totient, or ϕ(n)*.*This is often computed using the Extended Euclidean Algorithm, since e and ϕ(n) are relatively prime and d is to be the modular multiplicative inverse of e.d is kept as the private key exponent.The public key has modulus n and the public (or encryption) exponent e. The private key has modulus n and the private (or decryption) exponent d, which is kept secret.

**2. Encryption**

I. Person A transmits his/her public key (modulus n and exponent e) to Person B, keeping his/her private key secret.II. When Person B wishes to send the message "M" to Person A, he first converts M to an integer such that 0 < m < n by using agreed upon reversible protocol known as a padding scheme.III. Person B computes, with Person A's public key information, the ciphertext c corresponding to  
  
c ≡ me (mod n).  
IV. Person B now sends message "M" in ciphertext, or c, to Person A.

**3. Decryption**

I. Person A recovers m from c by using his/her private key exponent, d, by the computation  
  
m ≡ cd (mod n).II. Given m, Person A can recover the original message "M" by reversing the padding scheme.This procedure works since  
  
c ≡ me (mod n)*,*cd ≡(me)d (mod n),  
cd ≡ mde (mod n).  
  
By the symmetry property of mods we have that  
  
mde ≡ mde (mod n)*.*  
Since de = 1 + kϕ(n), we can write  
  
mde ≡ m1 + kϕ(n) (mod n),mde ≡ m(mk)ϕ(n) (mod n),mde ≡ m (mod n).  
  
From Euler's Theorem and the Chinese Remainder Theorem, we can show that this is true for all m and the original message  
  
cd ≡ m (mod n), is obtained.

The idea of making one of your own encryption algorithms public on the internet seems very strange at first.  However, this is acutally one of the most important steps in RSA encryption.  
  
If Person C intercepts your message to Person B, they already know the encryption key (exponent e, modulus n).  However, what he/she doesn't have is the decryption exponent d.  Since you encrypted your message with Person B's encryption key, only Person B has the decryption key (exponent d, modulus n) to decrypt it.  Person C is only missing one piece of information, exponent d, which turns out to be the hardest piece of information to find.  
  
Person C also knows that  
  
de ≡ 1 (mod ϕ(n)), or  
de ≡ 1 (mod (p - 1)(q - 1)).  
  
Since he/she knows that n = pq, the simplest way to find n would be to somehow factor n into the exact primes used by Person B in the algorithm.  From there, he/she could simply calculate the congruence to find d.  
  
With larger (which are more secure) primes, this turns out to be nearly impossible to do.  
  
If p = 7717 and q = 7919, n would be 61110923.  If we let e = 5, then all Person C knows is  
  
e = 5,  
n = 61110923.  
  
Clearly, it would take very long to factor n, but imagine what would happen if  
  
p = 982451653,  
q = 961748941.  
  
Then n would be 944871836856449473.  
  
Now factoring n is basically impossible to do by hand.  However, even this value of n is smaller than most values of n used in RSA Encryption.  It took 290 computers over the internet and a supercomputer 4 months to find that

n = 1094173864157052742180970732204035761200373294544920599091384213147634  
9984288934784717997257891267332497625752899781833797076537244027146743  
531593354333897  
  
Had prime factors   
  
p = 102639592829741105772054196573991675900716567808038066803341933521790711307779,  
q = 106603488380168454820927220360012878679207958575989291522270608237193062808643.  
  
In this case n had only 155 digits. Many values of n have over 200 digits, making the RSA algorithm nearly unbreakable.

**CHAPTER 3:**

**REQUIREMENTS ENGINEERING**

**3.1 GENERAL:**

These are the requirements for doing the project. Without using these tools and software’s we can’t do the project. So we have two requirements to do the project. They are

1. Hardware Requirements.

2. Software Requirements.

**3.2 HARDWARE REQUIREMENTS:**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the system does and not how it should be implemented.

PROCESSOR : PENTIUM IV 2.6 GHz, Intel Core 2 Duo.

RAM : 4GB DD RAM

MONITOR : 15” COLOR

HARD DISK : 40 GB

**3.3 SOFTWARE REQUIREMENTS:**

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team’s and tracking the team’s progress throughout the development activity.

Front End : J2EE (JSP, SERVLETS) JAVASCRIPT

Back End : MY SQL 5.5

Operating System : Windows 07

IDE : Eclipse

**CHAPTER 4**

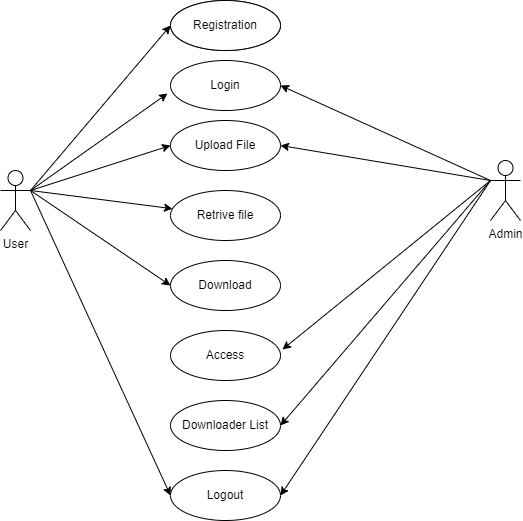
**DESIGN ENGINEERING**

**4.1 GENERAL**

**GENERAL**

Design Engineering deals with the various UML [Unified Modeling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

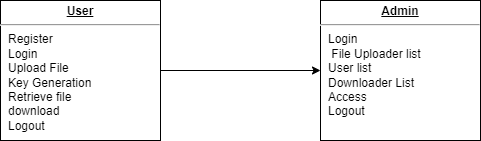
**USE CASE DIAGRAM:**

****

**EXPLANATION:**

The use case diagram is the main building block of [object oriented](http://en.wikipedia.org/wiki/Object_oriented) modeling. It is used both for general [conceptual modeling](http://en.wikipedia.org/wiki/Conceptual_model) of the systematic of the application, and for detailed modeling translating the models into [programming code](http://en.wikipedia.org/wiki/Programming_code). For this in our component diagram first propose a data in this proposed method we are using Hash-Solomon Code Algorithm to encrypt the data.

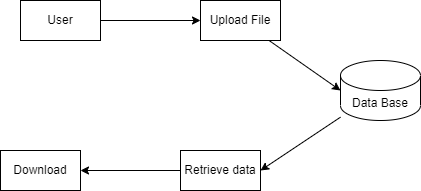
**CLASS DIAGRAM:**

****

**EXPLANTION:**

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. The classes in a class diagram represent both the main objects and or interactions in the application and the objects.

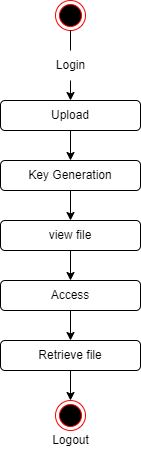
**OBJECT DIAGRAM:**

****

**EXPLANATION:**

An **object diagram** in the [Unified Modeling Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML), is a [diagram](https://en.wikipedia.org/wiki/Diagram) that shows a complete or partial view of the structure of a modeled [system](https://en.wikipedia.org/wiki/System) at a specific time. Object is an instance of a class in a particular moment in runtime that can have its own state and data values. Likewise a static [UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language) object diagram is an instance of a [class diagram](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/); it shows a snapshot of the detailed state of a system at a point in time, thus an object diagram encompasses objects and their relationships which may be considered a special case of a class diagram or a [communication diagram](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-communication-diagram/).

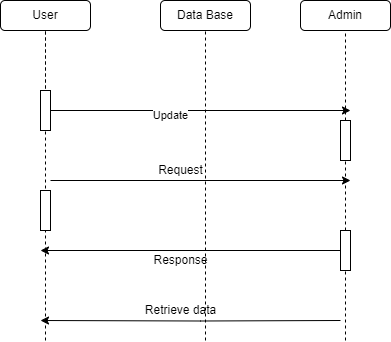
**STATE DIAGRAM:**

****

**EXPLANATION:**

State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics. In our state diagram first propose a . For this in our component diagram first propose a data. In this proposed method we are using Hash-Solomon Code Algorithm to encrypt the data.

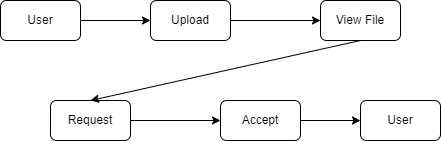
**SEQUENCE DIAGRAM:**

****

**EXPLANATION:**

In our sequence diagram specifying processes operate with one another and in order. In our sequence diagram first propose a For this in our component diagram first propose a data.In this proposed method we are using Hash-Solomon Code Algorithm to encrypt the data.

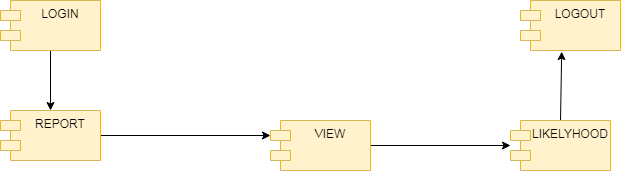
**COLLABORATION DIAGRAM:**

****

**EXPLANATION:**

Collaboration diagram, also called a communication diagram or interaction diagram is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved. Objects are shown as rectangles with naming labels inside. These labels are preceded by colons and may be underlined. The relationships between the objects are shown as lines connecting the rectangles.

**COMPONENT DIAGRAM:**



**EXPLANATION:**

Component diagrams show the dependencies and interactions between software components.  A component is a container of logical elements and represents things that participate in the execution of a system. Components also use the services of other components through one of its interfaces. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

**DATAFLOW DIAGRAM:**

LOGOUT

Login

Retrieve data

Request

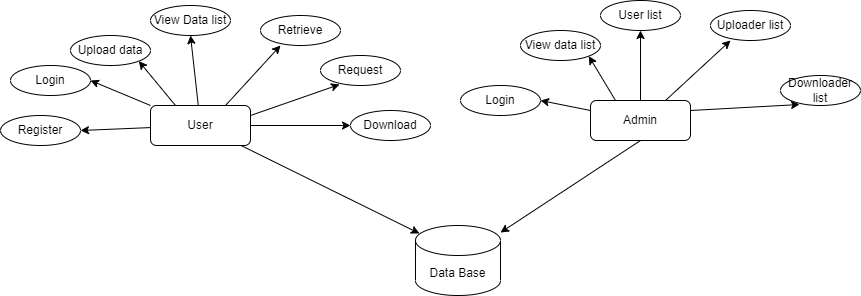
View File List

Upload File

**EXPLANATION:**

A data**-**flowdiagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow; there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

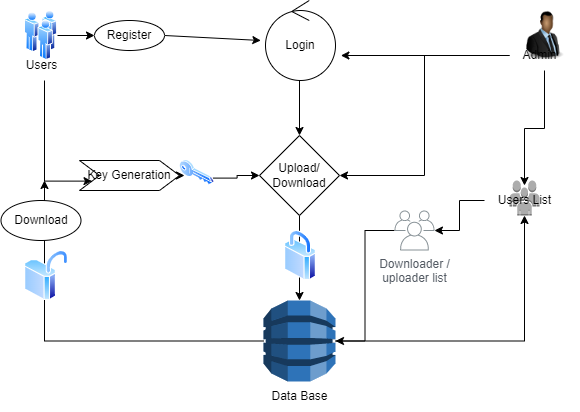
**E-R DIAGRAM:**

****

**EXPLANATION:**

An entity-relationship diagram (ERD) is a data modeling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure. For each data flow, at least one of the endpoints (source and / or destination) must exist in a process. The refined representation of a process can be done in another data-flow diagram, which subdivides this process into sub-processes.

SYSTEM ARCHITECTURE:



**EXPLANATION:**

The systems architect establishes the basic structure of the system, we propose a Hash code Solomon algorithm and we can put a small part of data in local machine and fog server in order to protect the privacy. Moreover, based on computational intelligence, this algorithm can compute the distribution proportion stored in cloud, fog, and local machine, respectively. Through the theoretical safety analysis and experimental evaluation, the feasibility of our scheme has been validated, which is really a powerful supplement to existing cloud storage scheme.

**CHAPTER 5**

**DEVELOPMENT TOOLS**

**5.1 GENERAL:**

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

**5.2 FEATURES OF JAVA**

**5.2.1 THE JAVA FRAMEWORK**

**Java** is a [programming language](http://en.wikipedia.org/wiki/Programming_language) originally developed by [James Gosling](http://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) and released in 1995 as a core component of Sun Microsystems' [Java platform](http://en.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its [syntax](http://en.wikipedia.org/wiki/Syntax_(programming_languages)) from [C](http://en.wikipedia.org/wiki/C_(programming_language)) and [C++](http://en.wikipedia.org/wiki/C%2B%2B) but has a simpler [object model](http://en.wikipedia.org/wiki/Object_model) and fewer [low-level](http://en.wikipedia.org/wiki/Low-level_programming_language) facilities. Java applications are typically [compiled](http://en.wikipedia.org/wiki/Compiler) to [byte code](http://en.wikipedia.org/wiki/Java_bytecode) that can run on any [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) (JVM) regardless of [computer architecture](http://en.wikipedia.org/wiki/Computer_architecture). Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere".

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

**5.2.2 OBJECTIVES OF JAVA**

To see places of Java in Action in our daily life, explore java.com.

## **WHY SOFTWARE DEVELOPERS CHOOSE JAVA:**

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

* Write software on one platform and run it on virtually any other platform
* Create programs to run within a Web browser and Web services
* Develop server-side applications for online forums, stores, polls, HTML forms processing, and more
* Combine applications or services using the Java language to create highly customized applications or services
* Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat

## **Some Ways Software Developers Learn Java:**

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

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

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

2. Encapsulation: It is the mechanism of combining the information and providing the abstraction.

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

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

**5.2.3 Java Server Pages - An Overview**

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

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

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

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

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

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

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

**5.2.4 Evolution of Web Applications:**

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

1. Scalability - a successful site will have more users and as the number of users is increasing fastly, the web applications have to scale correspondingly.

2. Integration of data and business logic - the web is just another way to conduct business, and so it should be able to use the same middle-tier and data-access code.

3. Manageability - web sites just keep getting bigger and we need some viable mechanism to manage the ever-increasing content and its interaction with business systems.

4. Personalization - adding a personal touch to the [web page](http://www.roseindia.net/jsp/javaserverpagestutorial.shtml) becomes an essential factor to keep our customer coming back again. Knowing their preferences, allowing them to configure the information they view, remembering their past transactions or frequent search keywords are all important in providing feedback and interaction from what is otherwise a fairly one-sided conversation.

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

1. Serve HTML and XML, and stream data to the web client

2. Separate presentation, logic and data

3. Interface to databases, other Java applications, CORBA, directory and mail services

4. Make use of application server middleware to provide transactional support.

5. Track client sessions .

**5.2.5 Benefits of JSP**

One of the main reasons why the JavaServer Pages technology has evolved into what it is today and it is still evolving is the overwhelming technical need to simplify application design by separating dynamic content from static template display data. Another benefit of utilizing JSP is that it allows to more cleanly separate the roles of web application/HTML designer from a software developer. The JSP technology is blessed with a number of exciting benefits, which are chronicled as follows:

1. The JSP technology is platform independent, in its dynamic web pages, its web servers, and its underlying server components. That is, JSP pages perform perfectly without any hassle on any platform, run on any web server, and web-enabled application server. The JSP pages can be accessed from any web server.

2. The JSP technology emphasizes the use of reusable components. These components can be combined or manipulated towards developing more purposeful components and page design. This definitely reduces development time apart from the At development time, JSPs are very different from Servlets, however, they are precompiled into Servlets at run time and executed by a JSP engine which is installed on a Web-enabled application server such as BEA WebLogic and IBM WebSphere.

**5.3 Servlets:**

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

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

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

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

**5.4 Java Servlets**

Java Servlet is a generic server extension that means a java class can be loaded dynamically to expand the functionality of a server. Servlets are used with web servers and run inside a Java Virtual Machine (JVM) on the server so these are safe and portable. Unlike applets they do not require support for java in the web browser. Unlike CGI, servlets don't use multiple processes to handle separate request. Servets can be handled by separate threads within the same process. Servlets are also portable and platform independent.

A web server is the combination of computer and the program installed on it. Web server interacts with the client through a web browser. It delivers the [web pages](http://www.roseindia.net/servlets/IntroductionToWebServer.shtml) to the client and to an application by using the web browser and  he HTTP protocols respectively.

The define the web server as the package of  large number of programs installed on a computer connected to Internet or intranet for downloading the requested files using [File Transfer](http://www.roseindia.net/servlets/IntroductionToWebServer.shtml) Protocol, serving e-mail and building and publishing web pages. A web server works on a client server model.

**5.5 Conclusion**

JSP and Servlets are gaining rapid acceptance as means to provide dynamic content on the Internet. With full access to the Java platform, running from the server in a secure manner, the application possibilities are almost limitless. When JSPs are used with Enterprise JavaBeans technology, e-commerce and database resources can be further enhanced to meet an enterprise's needs for web applications providing secure transactions in an open platform. J2EE technology as a whole makes it easy to develop, deploy and use web server applications instead of mingling with other technologies such as CGI and ASP. There are many tools for facilitating quick web software development and to easily convert existing server-side technologies to JSP and Servlets.

**CHAPTER 6**

**IMPLEMENTATION**

**6.1 GENERAL**

This chapter describes the implementation of searched based application. It deals with the source code for main viewpoint for Anonymous Database Management.

**Admin seen user opinion in category vise**

**CHAPTER 8**

**SOFTWARE TESTING:**

**8.1. FEASIBILITY STUDY**

Feasibility studies aim to objectively and rationally uncover the strengths and weaknesses of the existing business or proposed venture, opportunities and threats as presented by the environment, the resources required to carry through, and ultimately the prospects for success.

In its simplest term, the two criteria to judge feasibility are cost required and value to be attained. As such, a well-designed feasibility study should provide a historical background of the business or project, description of the product or service, accounting statements, details of the operations and management, marketing research and policies, financial data, legal requirements and tax obligations. Generally, feasibility studies precede technical development and project implementation.

They are 3 types of Feasibility

• Economical feasibility

• Technical feasibility

• Operational feasibility

**8.1.1. ECONOMICAL FEASIBILITY**

The assessment is based on an outline design of system requirements in terms of Input, Processes, Output, Fields, Programs, and Procedures. This can be quantified in terms of volumes of data, trends, frequency of updating, etc. in order to estimate whether the new system will perform adequately or not.

**8.1.2. TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources.

**8.1.3. OPERATIONAL FEASIBILIT**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

**8.2. SYSTEM TESTING**

The software, which has been developed, has to be tested to prove its validity. Testing is considered to be the least creative phase of the whole cycle of system design. In the real sense it is the phase, which helps to bring out the creativity of the other phases makes it shine.

**8.2.1. VARIOUS LEVELS OF TESTING**

1. White Box Testing

2. Black Box Testing

3. Unit Testing

4. Functional Testing

5. Performance Testing

6. Integration Testing

7. Validation Testing

8. System Testing

9. Output Testing

10. User Acceptance Testing

**8.2.1.1. WHITE BOX TESTING**

White-box testing, sometimes called glass-box, is a test case design method that uses the control structure of the procedural design to derive test cases. Using White Box testing methods, we can derive test cases that

• Guarantee that all independent paths within a module have been exercised at least once

• Exercise all logical decisions on their true and false sides.

• Execute all loops at their boundaries and within their operational bounds.

• Exercise internal data structures to assure their validity.

**8.2.1.2. BLACK BOX TESTING**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

**8.2.1.3. UNIT TESTING**

Unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process.

Unit testing is software verification and validation method in which the individual units of source code are tested fit for use. A unit is the smallest testable part of an application. In this testing, each class is tested to be working satisfactorily.

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

**8.2.1.4. FUNCTIONAL TESTING**

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional Testing usually describes what the system does. Functional testing differs from system testing in that functional testing "verifies a program by checking it against ... design document(s) or specification(s)", while system testing "validate a program by checking it against the published user or system requirements" (Kane, Falk, Nguyen 1999, p. 52). Functional testing typically involves five steps. The identification of functions that the software is expected to perform

1. The creation of input data based on the function's specifications

2. The determination of output based on the function's specifications

3. The execution of the test case

4. The comparison of actual and expected outputs.

**8.2.1.5. PERFORMANCE TESTING**

In general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

Performance testing is a subset of performance engineering, an emerging computer science practice which strives to build performance into the implementation, design and architecture of a system.

**8.2.1.6. INTEGRATION TESTING**

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when put together. The problem of course, is “putting them together”- interfacing. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready. All the errors found in the system are corrected for the next phase.

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e. assemblages (or groups of units), are exercised through their interfaces using black box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test whether all the components within assemblages interact correctlyfor example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing.

**8.2.1.7. VALIDATION TESTING**

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it full fills its intended purpose. These are critical components of a quality management system such as ISO 9000. The words "verification" and "validation" are sometimes preceded with "Independent" (or IV&V), indicating that the verification and validation is to be performed by a disinterested third party.

It is sometimes said that validation can be expressed by the query "Are you building the right thing?" and verification by "Are you building it right?". In practice, the usage of these terms varies. Sometimes they are even used interchangeably.

**8.2.1.8. SYSTEM TESTING**

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called *assemblages*) or between any of the *assemblages* and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behaviour and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification.

**8.2.1.9. OUTPUT TESTING**

After performing the validation testing, next step is output testing of the proposed system since no system could be useful if it does not produce the required output generated or considered in to two ways. One is on screen and another is printed format. The output comes as the specified requirements by the user. Hence output testing does not result in any correction in the system.

**8.2.1.10. USER ACCEPTANCE TESTING**

User acceptance of a system is the factor for the success of any system. The system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

• Input screen design.

• Output screen design.

• Online message to guide user.

• Format of the ad-hoc reports and other outputs.

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using the test data. While testing the system by using test data errors are again uncovered and correct.

**CHAPTER 9**

**9.1 FUTURE ENHANCEMENT:**

* In future implementation will add some algorithm to implement the project very secure.
* And add some module or states to improve more options to implementation.

**ADVANTAGES:**

* Safe and Secure.
* Analyze the file list then control

**10: CONCLUSION:**

The manager is stay aware of the nuances of downloader and the uploaded nuances. The client can download the record of who captivated with the help of decoded key. The client can move their interested record and can download with the entry of chairman. While moving the report is changed over totally to non-comprehensible code and the key as well.

**REFERENCE:**

**1.** Z. J. Czech, G. Havas and Β. S. Majewski, "Perfect Hashing", *Theoretical Computer Science*, no. 182, pp. 1-143, 1997.

[CrossRef](https://doi.org/10.1016/S0304-3975(96)00146-6)[Google Scholar](https://scholar.google.com/scholar?as_q=Perfect+Hashing&as_occt=title&hl=en&as_sdt=0%2C31)

**2.** Z. J. Czech, G. Havas and B. S. Majewski, "An optimal algorithm for generating minimal perfect hash functions", *Information Processing Letters*, vol. 43, no. 5, pp. 257-264, October 1992.

[CrossRef](https://doi.org/10.1016/0020-0190(92)90220-P)[Google Scholar](https://scholar.google.com/scholar?as_q=An+optimal+algorithm+for+generating+minimal+perfect+hash+functions&as_occt=title&hl=en&as_sdt=0%2C31)

**3.** R. Gupta, S. Bhaskar and S. Smolka, "On Randomization in Sequential and Distributed Algorithms", *ACM comput. Surveys*, vol. 26, no. 1, pp. 7-86, March 1994.

[CrossRef](https://doi.org/10.1145/174666.174667)[Google Scholar](https://scholar.google.com/scholar?as_q=On+Randomization+in+Sequential+and+Distributed+Algorithms&as_occt=title&hl=en&as_sdt=0%2C31)

**4.** J. Ebert, "A versatile data structure for edge-oriented graph algorithms", *Communications of the ACM*, vol. 30, no. 6, pp. 513-519, June 1987.

**5.** C. Cifuentes and K. J. Gough, "Decompilation of binary programs", *Software – Practice and Experience*, vol. 25, no. 7, pp. 811-829, 1995.

[CrossRef](https://doi.org/10.1002/spe.4380250706)[Google Scholar](https://scholar.google.com/scholar?as_q=Decompilation+of+binary+programs&as_occt=title&hl=en&as_sdt=0%2C31)

**6.** I. Guilfanov, *FLIRT (Fast Library Identification and Recognition Technology)*, [online] Available: http://www.datarescue.com/idaflirt.htm.

[Google Scholar](https://scholar.google.com/scholar?as_q=FLIRT+%28Fast+Library+Identification+and+Recognition+Technology%29&as_occt=title&hl=en&as_sdt=0%2C31)

**7.** J. Larus, *EEL: An Executable Editing Library*, [online] Available: http://www.cs.wisc.edu/~larus/eel.html.

[Google Scholar](https://scholar.google.com/scholar?as_q=EEL%3A+An+Executable+Editing+Library&as_occt=title&hl=en&as_sdt=0%2C31)

**8.** *Tracepoint An execution profiler for Windows/NT binary programs*, [online] Available: http://www.tracepoint.com.