

**XTREMELY RANDOMIZED TREES WITH
PRIVACY PRESERVATION FOR DISTRIBUTED
STRUCTURED HEALTH DATA**

A Major Project Report on

**“EXTREMELY RANDOMIZED TREES WITH PRIVACY PRESERVATION
FOR DISTRIBUTED STRUCTURED HEALTH DATA”**

Submitted to the

**GURU NANAK INSTITUTIONS TECHNICAL
CAMPUS(AUTONOMUS), Ibrahimpatnam, Hyderabad**

In partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

MOHAMMAD AMID ANSARI (19WJ1A05Y3)

SABAVATH SRIKANTH (20WJ5A0525)

UDAVATH KISHAN KUMAR (20WJ5A0528)

Under the Esteemed Guidance of

Mrs.CH.SUSHMA

Assistant Professors, CSE Dept.



Department of Computer Science Engineering

GURU NANAK INSTITUTIONS TECHNICAL CAMPUS(AUTONOMOUS)

School of Engineering & Technology Ibrahimpatnam, R.R District 501506

2022-2023



GURU NANAK INSTITUTIONS TECHNICAL CAMPUS



AUTONOMOUS
under Section 2 (f) & 12 (b) of
University Grants Commission Act

Date: 11-01-2023

Department of Computer Science Engineering

CERTIFICATE

This is to certify that this mini project report entitled “**EXTREMELY RANDOMIZED TREES WITH PRIVACY PRESERVATION FOR DISTRIBUTED STRUCTURED HEALTH DATA**” by **MOHAMMAD AMID ANSARI (19WJ1A05Y3), SABAVATH SRIKANTH (20WJ5A0525), UDAVATH KISHAN KUMAR (20WJ5A0528)** was submitted in partial fulfillment of the requirements for the degree of **Bachelor of Technology in Computer Science and Engineering** of the **GURU NANAK INSTITUTIONS TECHNICAL CAMPUS(AUTONOMOUS), Ibrahimpatnam, Hyderabad- 501506** during the academic year **2022-2023**, is a Bonafide record of work carried out under our guidance and supervision.

INTERNAL GUIDE

Mrs.CH.SUSHMA

PROJECT CO-ORDINATOR

Mrs. V.SWATHI

HOD CSE

Mr. V .DEVASEKHAR

EXTERNAL EXAMINER



RAM Innovative Infotech
M : +919581012012
E : raminnovativeinfotech@gmail.com
Flat NO.#309, Amrutha Ville,
Copp: yashoda Hospital, Somajiguda,
Hyderabad-82, Telangana, India
www.raminnovativeinfotech.webs.com

PROJECT COMPLETION CERTIFICATE

This is to certify that the following students of final year B.Tech, Department of **Computer Science and Engineering** Guru Nanak Institutions Technical Campus (GNITC) have completed their training and project at GNITC successfully.

STUDENT NAME:

ROLLNO:

- | | |
|---------------------------------------|--------------------------|
| 1. <u>MOHAMMAD AMID ANSARI</u> | <u>19WJ1A05Y3</u> |
| 2. <u>SABAVATH SRIKANTH</u> | <u>20WJ5A0525</u> |
| 3. <u>UDAVATH KISHAN KUMAR</u> | <u>20WJ5A0528</u> |

The training was conducted on **JAVA** Technology for the completion of the project titled **“EXTREMELY RANDOMIZED TREES WITH PRIVACY PRESERVATION FOR DISTRIBUTED STRUCTURED HEALTH DATA”** in **RAM INNOVATIVE INFOTECH** The project has been completed in all aspects.



ACKNOWLEDGEMENT

We wish to express our sincere thanks to **MR.SARDAR TAVINDER SINGH KOHLI ,CHAIRMAN, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **MR.SARDAR GAGANDEEP SINGH KOHLI ,Vice Chairman, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **DR.HARVINDER SINGH SAINI , Managing Director , GNITC** for providing us with a conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **DR.KODUGANTI VENKAT RAO , Director , GNITC** for providing us with a conducive environment for carrying through our academic schedules and Project with ease.

We wish to express our sincere thanks to **DR. RISHI SAYAL, Associate Director, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We would like to say our sincere thanks to **Mr. V .DEVASEKHAR Professor & Head, Department of CSE, GNITC** for providing seamless support and right suggestions are given in the development of the project.

We would like to say our sincere thanks to **Mrs. V.SWATHI Assistant Professor, Department of CSE, Project Coordinator**, for providing seamless support and right suggestions are given in the development of the project.

We have been truly blessed to have a wonderful internal guide **Mrs.CH.SUSHMA Assistant Professor, Department of CSE,GNITC** for guiding us to explore the ramification of our work and we express our sincere gratitude towards him for leading me through the completion of Project.

Finally, we would like to thank our family members for their moral support and encouragement to achieve their goals.

MOHAMMAD AMID ANSARI	(19WJ1A05Y3)
SABAVATH SRIKANTH	(20WJ5A0525)
UDAVATH KISHAN KUMAR	(20WJ5A0528)

LIST OF CONTENTS

CONTENTS	PAGE NO
ABSTRACT.....	IV
LIST OF DIAGRAMS	V
LIST AND DESCRIPTION OF SYMBOLS.....	VI
LIST OF ABBREVIATIONS.....	X
CHAPTER 1: INTRODUCTION.....	1
1.1 General.....	1
1.2 Scope of the Project.....	3
1.3 Objective.....	3
1.4 Existing System.....	4
1.4.1 Existing System Disadvantages.....	4
1.4.2 Literature Survey.....	5
1.5 Proposed System.....	11
1.5.1 Proposed System Advantages.....	11
CHAPTER 2: PROJECT DESCRIPTION.	12
2.1 General	12
2.2 Methodologies.	12
2.2.1 Modules Name	12
2.2.2 Modules Explanation and Diagram.....	13
2.2.3 Given Input Expected Output.....	17
2.3 Technique Used or Algorithm Used.....	18
CHAPTER 3: REQUIREMENTS ENGINEERING.....	19
3.1 General.....	19
3.2 Hard Requirements.....	19
3.3 Software Requirements.....	20
3.4 Functional Requirements.....	21
3.5 Non-Functional Requirements.....	21

CHAPTER 4:DESIGN ENGINEERING.....	22
4.1 General.	22
4.1.1 Use Case Diagram	23
4.1.2 Class Diagram	24
4.1.3 Object Diagram	25
4.1.4 State Diagram.....	26
4.1.5 Sequence Diagram.....	27
4.1.6 Collaboration Diagram.....	28
4.1.7 Activity Diagram.....	29
4.1.8 Component Diagram.....	30
4.1.9 E-R Diagram.....	31
4.1.10 Data Flow Diagram.....	32
4.1.11 Deployment Diagram.....	35
4.2 System Architecture.....	36
CHAPTER 5 : DEVELOPMENT TOOLS.....	37
5.1 General... ..	37
5.2 Features of Java	37
5.2.1 The Java Framework... ..	37
5.2.2 Objectives of Java	38
5.2.3 Java Swing Overview... ..	39
5.2.4 Evaluation of Collection Frameworks... ..	40
5.3 Conclusion	43
CHAPTER 6:IMPLEMENTATION.....	44
6.1 General... ..	44
6.2 Implementation	44
CHAPTER 7:SNAPSHOTS.....	49
7.1 General... ..	49
7.2 Various Snapshots.....	49

CHAPTER 8: SOFTWARE TESTING.....	53
8.1 General.	53
8.2 Developing Methodologies.....	53
8.3 Types of Testing.....	54
8.3.1 Unit Testing.....	55
8.3.2 Functional Test.....	55
8.3.3 System Test.....	55
8.3.4 Performance Test.....	55
8.3.5 Integration Test.....	56
8.3.6 Acceptance Test.	56
8.3.7 Build the Test Plan	56
CHAPTER 9: APPLICATION.....	57
9.1 General.....	57
9.2 Future Enhancement.....	57
CHAPTER 10: CONCLUSION & REFERENCES ..	58
10.1 Conclusion.....	58
10.2 References.	59

ABSTRACT

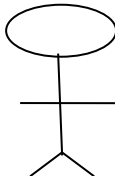
Artificial intelligence and machine learning have recently attracted considerable attention in the healthcare domain. The data used by machine learning algorithms in healthcare applications is often distributed over multiple sources, for instance, hospitals or patients' personal devices. One main difficulty lies in analyzing such data without compromising patients' privacy and personal data, which is a primary concern in healthcare applications. Therefore, in these applications, we are interested in running machine learning algorithms over distributed data without disclosing sensitive information about the data subjects. In this paper, we propose a distributed extremely randomized trees algorithm for learning from distributed data with privacy preservation. We present the implementation of our technique (which we refer to as k-PPD-ERT) on a cloud platform and demonstrate its performance based on medical data, including Heart Disease, Breast Cancer, and mental health datasets (Depression and Psykose datasets) associated with the Norwegian Introducing Mental health through Adaptive Technology (INTROMAT) project.

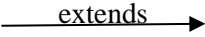

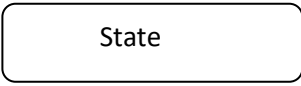
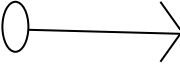
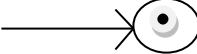
TABLE OF CONTENTS

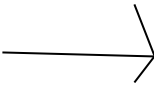
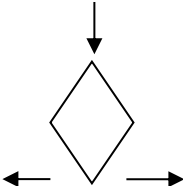
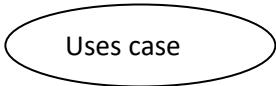
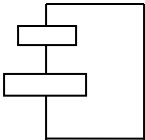
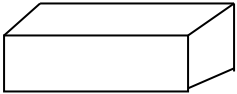
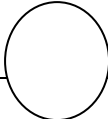
LIST OF FIGURES




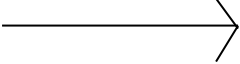
FIGURE NO	NAME OF THE FIGURE	PAGE NO.
2.2.1	Module Diagram	12
4.1.1	Use case Diagram	23
4.1.2	Class Diagram	24
4.1.3	Object Diagram	25
4.1.4	State Diagram	26
4.1.5	Sequence Diagram	27
4.1.6	Collaboration Diagram	28
4.1.7	Activity Diagram	29
4.1.8	Component Diagram	30
4.1.9	E-R Diagram	31
4.1.10	Data Flow Diagram	32
4.1.11	Deployment Diagram	35
4.2	System Architecture	36
7.1	Home Page	49

LIST OF SYMBOLS

S.NO	NAME	NOTATION	DESCRIPTION
1.	Class	<div style="display: flex; align-items: center; justify-content: space-around;"> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>+ public</i> <i>-private</i> </div> <div style="border: 1px solid black; padding: 5px; text-align: center;"> <i>Class Name</i> <i>-attribute</i> <i>-attribute</i> </div> </div>	Represents a collection of similar entities grouped together.
2.	Association	<div style="display: flex; align-items: center; justify-content: space-around; margin-bottom: 10px;"> <div style="border: 1px solid black; padding: 5px;">Class A</div> <div style="border: 1px solid black; padding: 5px;">NAME</div> <div style="border: 1px solid black; padding: 5px;">Class B</div> </div> <div style="display: flex; align-items: center; justify-content: space-around;"> <div style="border: 1px solid black; padding: 5px;">Class A</div> <div style="border: 1px solid black; padding: 5px;">Class B</div> </div>	Associations represents static relationships between classes. Roles represents the way the two classes see each other.
3.	Actor		It aggregates several classes into a single classes.
5.	Aggregation	<div style="display: flex; align-items: center; justify-content: space-around; margin-bottom: 10px;"> <div style="border: 1px solid black; padding: 5px;">Class A</div> <div style="border: 1px solid black; padding: 5px;">Class A</div> </div> <div style="display: flex; align-items: center; justify-content: space-around;"> <div style="border: 1px solid black; padding: 5px;">Class B</div> <div style="border: 1px solid black; padding: 5px;">Class B</div> </div>	Interaction between the system and external environment

5.	<i>Relation</i> (uses)	<i>Uses</i>	Used for additional process communication.
6.	Relation (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 of the process.
9.	Initial State		Initial state of the object
10.	Final state		Final 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.
		 v	

16.	Data Process/State		A circle in DFD represents a state or process which has been triggered due to some event or action.
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 ABBREVIATION

S.NO	ABBREVIATION	EXPANSION
1.	DB	Data Base
2.	JVM	Java Virtual Machine
3.	JSP	Java Server Page
4.	PWS	Personalized Web Search
5.	UPS	User Personalized Search
6.	JRE	Java Runtime Environment

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Artificial intelligence (AI) and automated decision-making have the potential to improve accuracy and efficiency in healthcare applications. In particular, AI is proven to outperform medical experts in certain domains. Two examples are the classification of rhythms in electrocardiography signals with deep neural networks in and prediction of breast cancer using the AI system presented in; more related studies can be found in. However, the application of AI and machine learning for automated decision-making in healthcare comes with challenges, such as security and privacy. For instance, a patient's privacy is violated if sharing his/her medical data with a third-party data recipient reveals that he/she has a medical condition. This becomes more challenging considering that, in healthcare systems, the data could be distributed over a number of sources rather than being stored in a central database. In distributed settings, hospitals need to apply data mining methods to extract useful patterns from patients' data. Although hospitals may individually be able to use their limited resources and locally stored health information to perform data mining, the use of available health information across several hospitals leads to obtaining more valuable and accurate information. However, this is a challenging task due to privacy and legal concerns. Hospitals often need to comply with privacy regulations that restrict sharing health information about patients with other parties, e.g., other hospitals, family doctors, and specialists. A similar problem exists when the data is distributed over patients' personal devices, such as mobile phones or wearable devices. Traditionally, it was assumed that all sources holding part of the data might share their information with a trusted party. However, such an assumption, i.e., putting this level of trust in a third party, is not feasible in every scenario because the privacy of data sources cannot be protected from the third party. In order to address the privacy concern, one solution would be to perturb the data before sharing it. However, perturbation-based solutions have limitations in satisfying data privacy and data utility requirements. This is because the utility of the data will decrease if the perturbation is not precisely controlled, and the privacy will not be preserved if the perturbation is not sufficient. Similarly, anonymization techniques, e.g., share an altered version of data to prevent the re-identification of data subjects. Moreover, methods providing differential privacy share data while preserving the privacy of individuals by adding noise. Nevertheless, in these techniques, there is

always a trade-off between data privacy and data utility. Previous studies also consider cryptographic techniques and secure multi-party computation methods for conducting privacy-preserving data mining. However, such approaches are inefficient, mainly when dealing with largescale data, due to considerable communication and computation costs. Several techniques, e.g., have been proposed to address these types of overheads in the privacy-preserving machine learning algorithms and to improve their efficiency. In this paper, we target the problem of learning from data held on multiple sources without explicit sharing of raw information. We assume that the learning data is horizontally partitioned, meaning that different records of data are stored on different sources. We focus on the classification problem and structured health data, which can be stored in spreadsheets. We build upon our previous work and propose a scalable privacy-preserving framework for distributed machine learning based on the extremely randomized trees algorithm, which has a linear overhead in the number of parties and can handle missing values. We refer to our approach as k-PPD-ERT (Privacy-Preserving Distributed Extremely Randomized Trees), in which k is the number of colluding parties in our approach. We use two popular publicly available healthcare datasets for performance evaluation, i.e., the Heart Disease and the Breast Cancer Wisconsin (Diagnostic) datasets. This data represents medical applications where missing values are present, and our algorithm is designed to handle such scenarios. Finally, we present the implementation of our technique on Amazon’s AWS cloud and evaluate it in a real-world setting based on the mental health datasets associated with the Norwegian Introducing Mental health through Adaptive Technology (INTROMAT) project. The remainder of this paper is organized as follows. Section II reviews the state of the art of distributed privacy-preserving machine learning techniques to address the discussed problem. Section III covers the background related to the extremely randomized trees algorithm and secure multi-party computation. In Section IV, we illustrate our proposed k-PPD-ERT method, which is an adaptation and extension of the ERT algorithm for distributed settings. Section V illustrates the distributed extremely randomized trees algorithm through a small example. In Section VI, we evaluate the performance, overhead and privacy of the proposed technique. Section VII serves as the conclusion of this article.

1.2 SCOPE OF THE PROJECT

In this paper, we target the problem of learning from data held on multiple sources without explicit sharing of raw information. We assume that the learning data is horizontally partitioned, meaning that different records of data are stored on different sources. We focus on the classification problem and structured health data, which can be stored in spreadsheets. We build upon our previous work [28] and propose a scalable privacy-preserving framework for distributed machine learning based on the extremely randomized trees algorithm, which has a linear overhead in the number of parties and can handle missing values. We refer to our approach as k-PPD-ERT (Privacy-Preserving Distributed Extremely Randomized Trees), in which k is the number of colluding parties in our approach. We use two popular publicly available healthcare datasets for performance evaluation, i.e., the Heart Disease [29] and the Breast Cancer Wisconsin (Diagnostic) [30] datasets.

1.3 OBJECTIVE

The objective of this the topic of collaborative learning from distributed data has been discussed in the literature for many years. A wide range of distributed learning techniques has been proposed in the literature that do not explicitly consider privacy aspects. Nevertheless, such techniques indirectly contribute to privacy preservation by limiting the amount of data that has to be shared with other parties or transferred to central servers or the cloud.

1.4 EXISTING SYSTEM:

- Artificial intelligence and machine learning have recently attracted considerable attention in the healthcare domain.
- The data used by machine learning algorithms in healthcare applications is often distributed over multiple sources, for instance, hospitals or patients' personal devices.
- One main difficulty lies in analyzing such data without compromising patients' privacy and personal data, which is a primary concern in healthcare applications.
- Therefore, in these applications, we are interested in running machine learning algorithms over distributed data without disclosing sensitive information about the data subjects.
- Artificial intelligence (AI) and automated decision-making have the potential to improve accuracy and efficiency in healthcare applications. In particular, AI is proven to outperform medical experts in certain domains.

➤ 1.4.1 Existing System Disadvantages:

- One main difficulty lies in analyzing such data without compromising patients' privacy and personal data, which is a primary concern in healthcare applications.
- Distributed data without disclosing sensitive information about the data subjects.

1.4.2 LITERATURE SURVEY

TITLE: Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network.

AUTHOR: A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng.

YEAR: 2019.

DESCRIPTION:

Computerized electrocardiogram (ECG) interpretation plays a critical role in the clinical ECG workflow¹. Widely available digital ECG data and the algorithmic paradigm of deep learning² present an opportunity to substantially improve the accuracy and scalability of automated ECG analysis. However, a comprehensive evaluation of an end-to-end deep learning approach for ECG analysis across a wide variety of diagnostic classes has not been previously reported. Here, we develop a deep neural network (DNN) to classify 12 rhythm classes using 91,232 single-lead ECGs from 53,549 patients who used a single-lead ambulatory ECG monitoring device. When validated against an independent test dataset annotated by a consensus committee of board-certified practicing cardiologists, the DNN achieved an average area under the receiver operating characteristic curve (ROC) of 0.97. The average F_1 score, which is the harmonic mean of the positive predictive value and sensitivity, for the DNN (0.837) exceeded that of average cardiologists (0.780). With specificity fixed at the average specificity achieved by cardiologists, the sensitivity of the DNN exceeded the average cardiologist sensitivity for all rhythm classes. These findings demonstrate that an end-to-end deep learning approach can classify a broad range of distinct arrhythmias from single-lead ECGs with high diagnostic performance similar to that of cardiologists. If confirmed in clinical settings, this approach could reduce the rate of misdiagnosed computerized ECG interpretations and improve the efficiency of expert human ECG interpretation by accurately triaging or prioritizing the most urgent conditions.

TITLE: International evaluation of an AI system for breast cancer screening.

AUTHOR: S. McKinney et al.

YEAR: 2020.

DESCRIPTION:

Screening mammography aims to identify breast cancer at earlier stages of the disease, when treatment can be more successful¹. Despite the existence of screening programmes worldwide, the interpretation of mammograms is affected by high rates of false positives and false negatives². Here we present an artificial intelligence (AI) system that is capable of surpassing human experts in breast cancer prediction. To assess its performance in the clinical setting, we curated a large representative dataset from the UK and a large enriched dataset from the USA. We show an absolute reduction of 5.7% and 1.2% (USA and UK) in false positives and 9.4% and 2.7% in false negatives. We provide evidence of the ability of the system to generalize from the UK to the USA. In an independent study of six radiologists, the AI system outperformed all of the human readers: the area under the receiver operating characteristic curve (AUC-ROC) for the AI system was greater than the AUC-ROC for the average radiologist by an absolute margin of 11.5%. We ran a simulation in which the AI system participated in the double-reading process that is used in the UK, and found that the AI system maintained non-inferior performance and reduced the workload of the second reader by 88%. This robust assessment of the AI system paves the way for clinical trials to improve the accuracy and efficiency of breast cancer screening.

TITLE: Digital privacy in mental healthcare: Current issues and recommendations for technology use.

AUTHOR: S. D. Lustgarten, Y. L. Garrison, M. T. Sinnard, and A. W. Flynn.

YEAR: 2020.

DESCRIPTION:

Mental healthcare providers increasingly use technology for psychotherapy services. This progress enables professionals to communicate, store information, and rely on digital software and hardware. Emails, text messaging, telepsychology/telemental health therapy, electronic medical records, cloud-based storage, apps/applications, and assessments are now available within the provision of services. Of those mentioned, some are directly utilized for psychotherapy while others indirectly aid providers. Whereas professionals previously wrote notes locally, technology has empowered providers to work more efficiently with third-party services and solutions. However, the implementation of these advancements in mental healthcare involves consequences to digital privacy and might increase clients' risk of unintended breaches of confidentiality. This manuscript reviews common technologies, considers the vulnerabilities therein, and proposes suggestions to strengthen privacy.

TITLE: Federated self-supervised learning of multisensory representations for embedded intelligence.

AUTHOR: I. Goienetxea, J. M. Martinez-Otzeta, B. Sierra, and I. Mendiola

YEAR: 2019

DESCRIPTION:

Smartphones, wearables, and Internet-of-Things (IoT) devices produce a wealth of data that cannot be accumulated in a centralized repository for learning supervised models due to privacy, bandwidth limitations, and the prohibitive cost of annotations. Federated learning provides a compelling framework for learning models from decentralized data, but conventionally, it assumes the availability of labeled samples, whereas on-device data are generally either unlabeled or cannot be annotated readily through user interaction. To address these issues, we propose a self-supervised approach termed scalogram-signal correspondence learning based on wavelet transform (WT) to learn useful representations from unlabeled sensor inputs as electroencephalography, blood volume pulse, accelerometer, and WiFi channel-state information. Our auxiliary task requires a deep temporal neural network to determine if a given pair of a signal and its complementary view (i.e., a scalogram generated with WT) align with each other, by optimizing a contrastive objective. We extensively assess the quality of learned features with our multiview strategy on diverse public data sets, achieving strong performance in all domains. We demonstrate the effectiveness of representations learned from an unlabeled input collection on downstream tasks with training a linear classifier over pretrained network, usefulness in low-data regime, transfer learning, and cross-validation. Our methodology achieves competitive performance with fully supervised networks and it works significantly better than pretraining with autoencoders in both central and federated contexts. Notably, it improves the generalization in a semisupervised setting as it reduces the volume of labeled data required through leveraging self-supervised learning.

TITLE: Federated self-supervised learning of multisensory representations for embedded intelligence.

AUTHOR: A. Saeed, F. D. Salim, T. Ozcelebi, and J. Lukkien.

YEAR: 2021.

DESCRIPTION:

Smartphones, wearables, and Internet-of-Things (IoT) devices produce a wealth of data that cannot be accumulated in a centralized repository for learning supervised models due to privacy, bandwidth limitations, and the prohibitive cost of annotations. Federated learning provides a compelling framework for learning models from decentralized data, but conventionally, it assumes the availability of labeled samples, whereas on-device data are generally either unlabeled or cannot be annotated readily through user interaction. To address these issues, we propose a self-supervised approach termed scalogram-signal correspondence learning based on wavelet transform (WT) to learn useful representations from unlabeled sensor inputs as electroencephalography, blood volume pulse, accelerometer, and WiFi channel-state information. Our auxiliary task requires a deep temporal neural network to determine if a given pair of a signal and its complementary view (i.e., a scalogram generated with WT) align with each other, by optimizing a contrastive objective. We extensively assess the quality of learned features with our multiview strategy on diverse public data sets, achieving strong performance in all domains. We demonstrate the effectiveness of representations learned from an unlabeled input collection on downstream tasks with training a linear classifier over pretrained network, usefulness in low-data regime, transfer learning, and cross-validation. Our methodology achieves competitive performance with fully supervised networks and it works significantly better than pretraining with autoencoders in both central and federated contexts. Notably, it improves the generalization in a semisupervised setting as it reduces the volume of labeled data required through leveraging self-supervised learning.

TITLE: Real-time classification technique for early detection and prevention of myocardial infarction on wearable devices.

AUTHOR: S. K. Kermanshahi, J. K. Liu, R. Steinfeld, and S. Nepal

YEAR: 2019

DESCRIPTION:

Continuous monitoring of patients suffering from cardiovascular diseases and, in particular, myocardial infarction (MI) places a considerable burden on health-care systems and government budgets. The rise of wearable devices alleviates this burden, allowing for long-term patient monitoring in ambulatory settings. One of the major challenges in this area is to design ultra-low energy wearable devices for long-term monitoring of patients' vital signs. In this work, we present a real-time event-driven classification technique, based on support vector machines (SVM) and statistical outlier detection. The main goal of this technique is to maintain a high classification accuracy while reducing the complexity of the classification algorithm. This technique leads to a reduction in energy consumption and thus battery lifetime extension. We validate our approach on a well-established and complete myocardial infarction (MI) database (Physio bank, PTB Diagnostic ECG database [1]). Our experimental evaluation demonstrates that our real-time classification scheme outperforms the existing approaches in terms of energy consumption and battery lifetime by a factor of 3, while maintaining the classification accuracy at a medically-acceptable level of 90%.

1.5 PROPOSED SYSTEM

- In this paper, we propose a secure verifiable semantic.
- Searching scheme that treats matching between queries and documents as an optimal matching task. We treat the document words as “suppliers,” the query words as “consumers,” and the semantic information as “product,” and design the minimum word transportation cost (MWTC) as the similarity metric between queries and documents.

1.5.1 PROPOSED SYSTEM ADVANTAGE

- Providing more security
- Reducing storage cost.
- For secure semantic optimal matching on the ciphertext,

CHAPTER 2

PROJECT DESCRIPTION

2.1 GENERAL:

We assume that the data owner is trusted, and the data users are authorized by the data owner. The communication channels between the owner and users are secure on existing security protocols such as SSL, TLS. With regard to the cloud server, our scheme resists a more challenging security model which is beyond the “semi-honest server” used in other secure semantic searching schemes. In our model, the dishonest cloud server attempts to return wrong/forged search results and learn sensitive information, but would not maliciously delete or tamper with the outsourced documents. Therefore, our secure semantic scheme should guarantee the verifiability, and confidentiality under such a security model.

2.2 METHODOLOGIES

2.2.1MODULES NAME:

1. User Interface Design
2. Data User
3. Data Owner
4. Cloud Server

2.2.2 MODULES EXPLANATION:

1. User Interface Design

In this module we design the windows for the project. These windows are used for secure login for all users. To connect with server user must give their username and password then only they can able to connect the server. If the user already exists directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page.

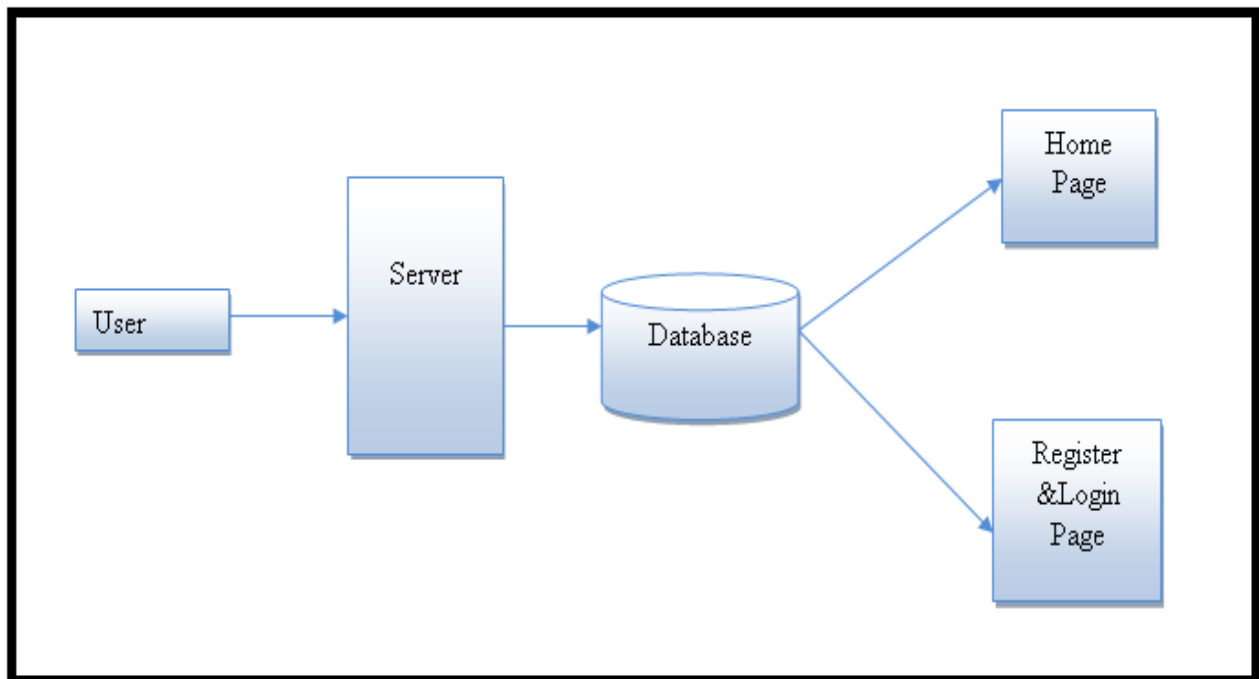


Figure 1: User Interface Design

2. Data User

This is the first module Data User can register and Login. After login Data User have an option of searching the files as a file name. Data user can also have a download file it will show an encrypted data. Data user can also send a trapdoor request to the server. Server can accept the the request and then data user can takes permissions from the owner then the file it will downloaded in plain text.

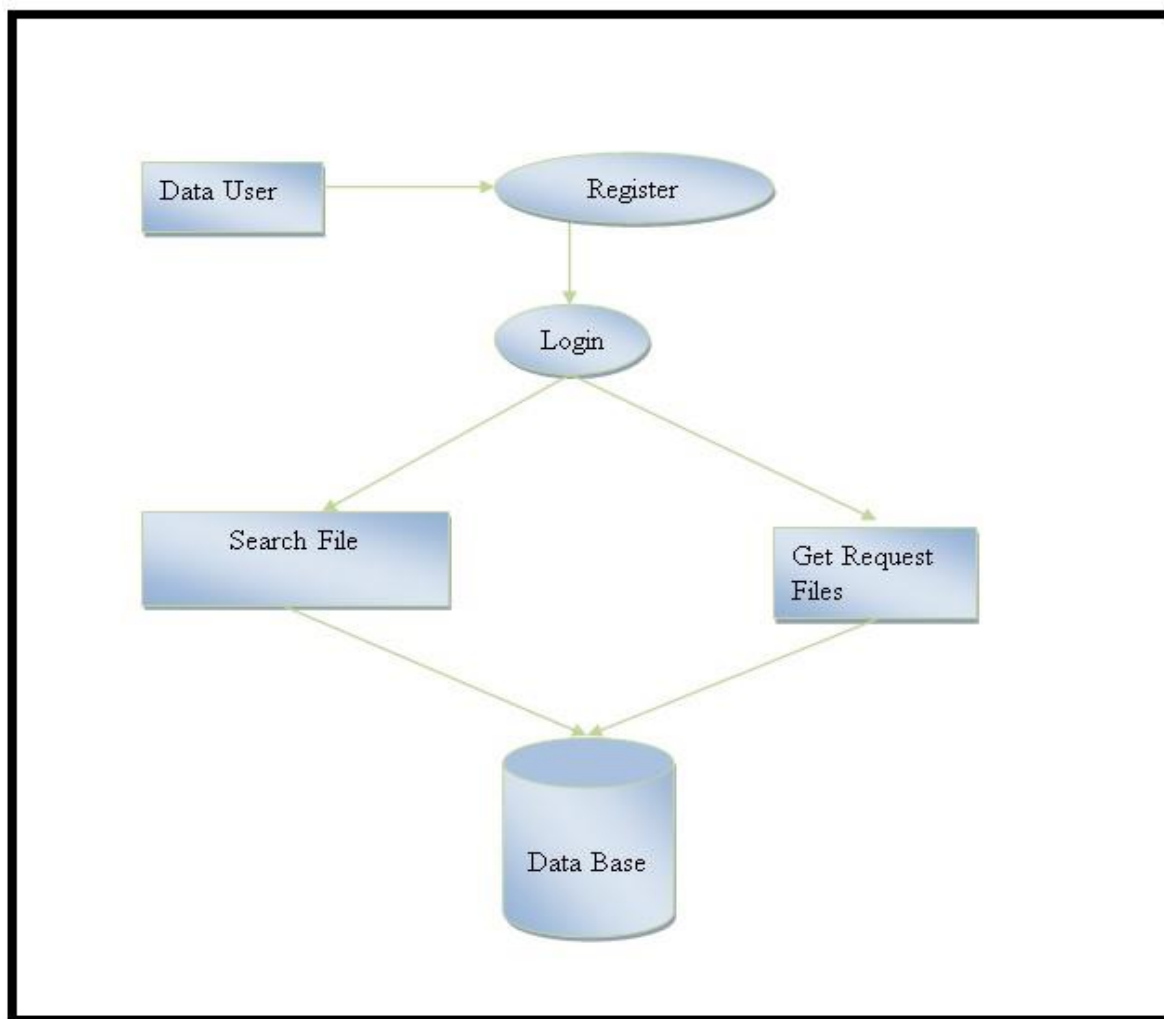


Figure 2: Data User

3. Data Owner

This is the Second module of this project. In this module Data Owner should register and Login. Data Owner will Uploads the files into the database. Data owner can also send request to the data user.

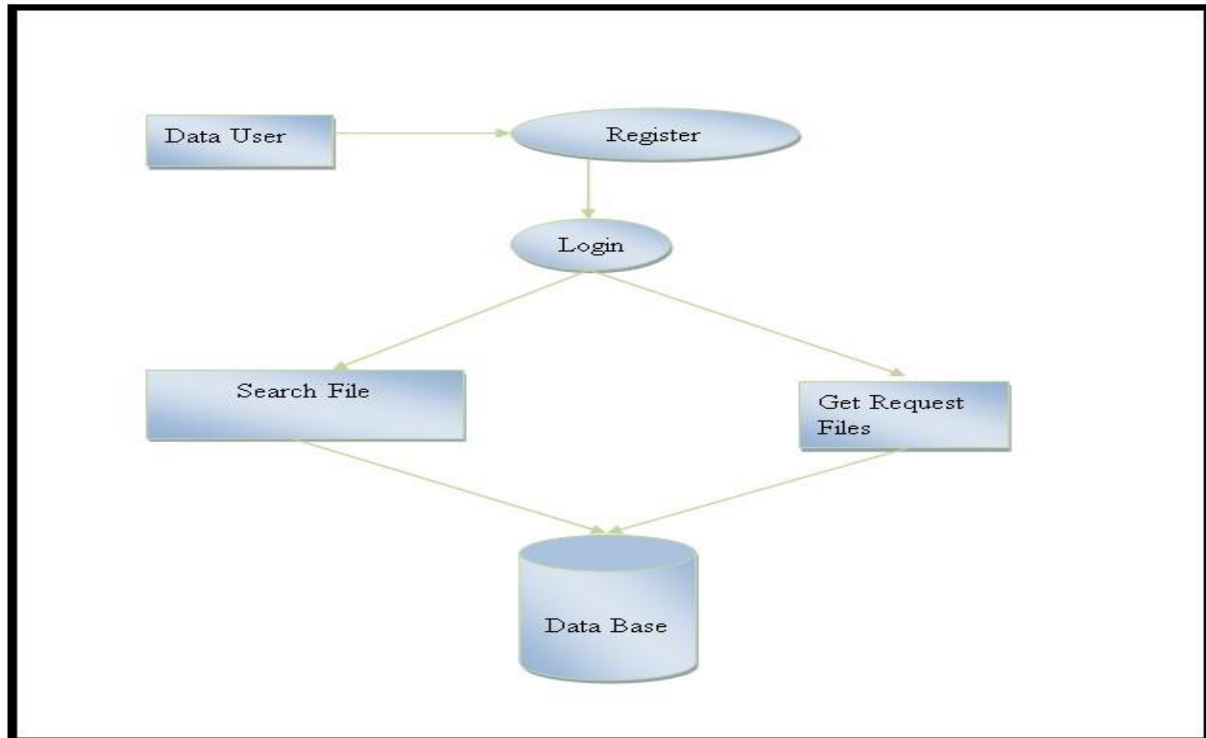


Figure 3: Data Owner

4. Cloud Server

This is the third module of this project. In this module Cloud Server can login. After login it will see all data owners' information. Cloud server can see all users' information. Cloud server can see an all stored data files. Cloud server can give keys request to the user. Cloud server can also see an attacker information of file.

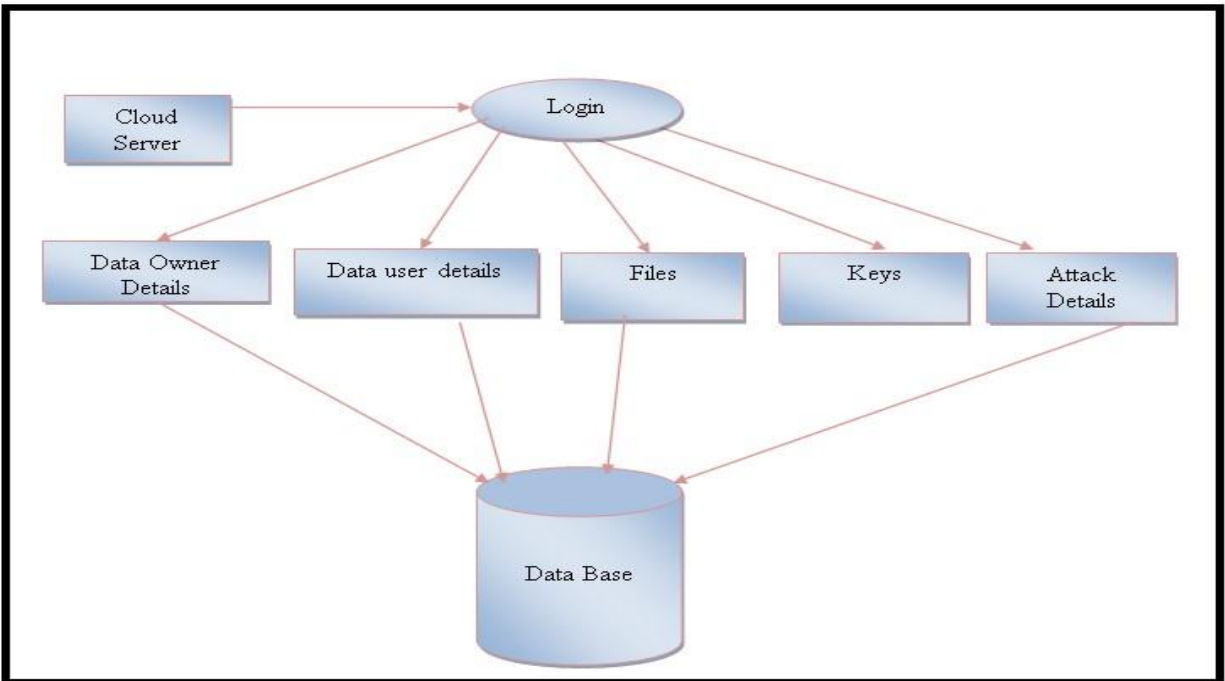


Figure 4: Cloud Server

2.2.3 GIVEN INPUT EXPECTED OUTPUT:

➤ User Interface Design

Input : Enter Login name and Password

Output : If valid user name and password then directly open the home page otherwise show error message and redirect to the registration page.

➤ Data User

Input : Data User Login name and Password

Output: If valid user name and password then directly open the Data user home page otherwise show error message and redirect to the data user login page.

➤ Data Owner

Input : Enter the owner name and password

Output : If valid owner name and password then directly open the data owner home page otherwise show error message and redirect to the data owner login page.

➤ Cloud Server

Input : Enter the Cloud Server name and password

Output: If valid Cloud Server name and password then directly open the Cloud Server home page otherwise show error message and redirect to the cloud Server login page.

2.3 TECHNIQUE USED OR ALGORITHM USED

Proposed Algorithm

➤ **linear programming (LP) problems to obtain the encrypted MWTC**

Treating the matching between queries and documents as an optimal matching task, we formulate the word transportation (WT) problem following the optimal transportation problem of linear programming. We utilize WT problems to calculate the minimum word transportation cost (MWTC) as the similarity metric between queries and documents.

We introduce the forward indexes as semantic information of documents. We define each keyword and its weight in the forward index of a document as the keywords distributions for the document. Therefore, we need to select keywords for each document and calculate the weight of each keyword in a specific document. Without loss of generality, we use TF-IDF (term frequency inverse document frequency) as a criterion to select keywords in our scheme.

EXISTING SYSTEM

➤ **kNN algorithm**

In introduced homomorphic encryption to encrypt relevance scores and realize a multi keyword ranked search scheme under the vector space model. Encryption techniques. It is not supporting multi-keyword ranked searching schemes that can resist against several attacks brought by OPE-based schemes. Secure Semantic Searching

CHAPTER 3

REQUIREMENTS ENGINEERING

3.1 GENERAL

We have conducted experiments on our collected dataset and extensive results have demonstrated that our model outperforms all other existing models. In the future, we will investigate more tasks under this framework, such as event summarization and event attribute mining in social media.

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 should what the system do and not how it should be implemented.

- PROCESSOR : DUAL CORE 2 DUOS.
- RAM : 2GB DD RAM
- HARD DISK : 250 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 teams and tracking the team's progress throughout the development activity.

SOFTWARE REQUIREMENTS

- FRONT END : J2EE (JSP, SERVLET)
- BACK END : MY SQL 5.5
- OPERATING SYSTEM : WINDOWS 7
- IDE : ECLIPSE

3.4 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, Firstly, the system is the first that achieves the standard notion of semantic security for data confidentiality in attribute-based deduplication systems by resorting to the hybrid cloud architecture.

3.5 NON-FUNCTIONAL REQUIREMENTS

➤ EFFICIENCY

Our multi-modal event tracking and evolution frameworks suitable for multimedia documents from various social media platforms, which can not only effectively capture their multi-modal topics, but also obtain the evolutionary trends of social events and generate effective event summary details over time. Our proposed Meta model can exploit the multi-modal property of social event, which can effectively model social media documents including long text with related images and learn the correlations between textual and visual modalities to separate the visual-representative topics and non-visual-representative topics.

CHAPTER 4

DESIGN ENGINEERING

4.1 GENERAL

Design Engineering deals with the various UML [Unified Modelling 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.

UML DIAGRAMS

4.1.1 USE CASE DIAGRAM

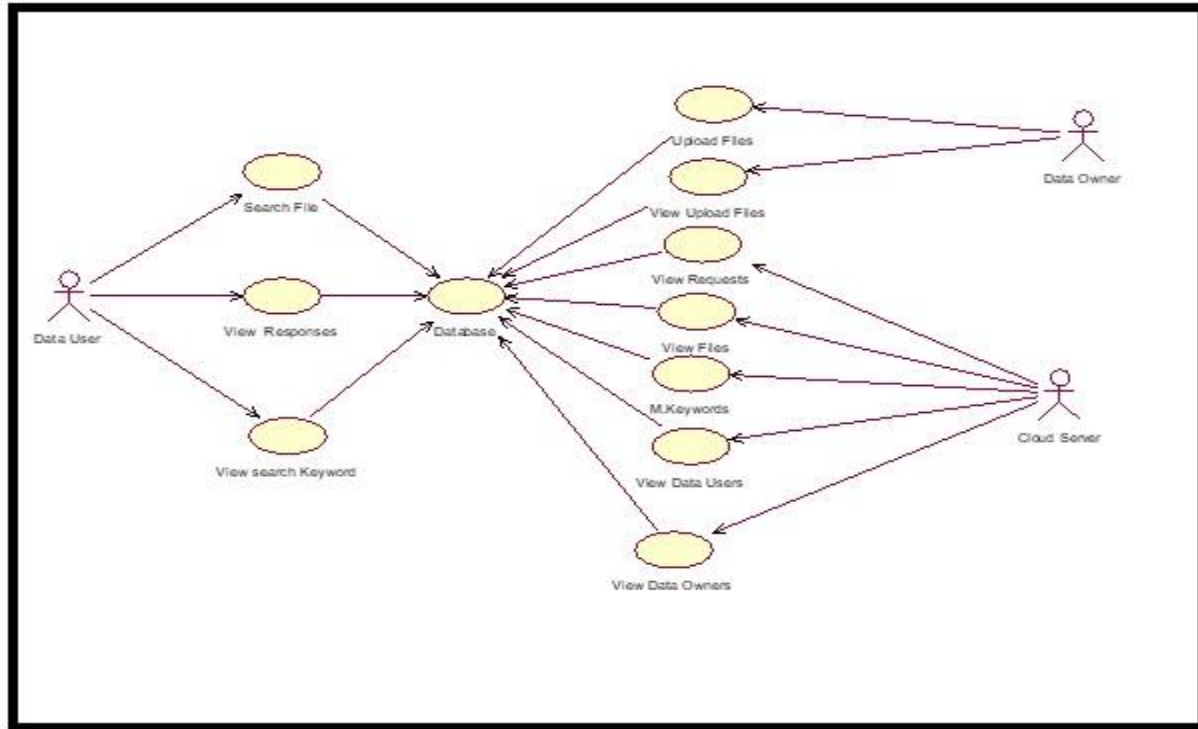


Figure 4.1.1: Use Case Diagram

EXPLANATION:

The main purpose of a use case diagram is to show that we have an actor is a data user, data owner and Cloud server. Different types of actions perform. Data user is an actor to perform some actions on search file, View responses and View search keywords. Data owner is is an actor. It will have an actions on an upload files and view uploaded files. Cloud server can also an actor. It will have an actions on view responses, View files, Match keywords, view data users and view data owners this all perform by the actor.

4.1.2 CLASS DIAGRAM

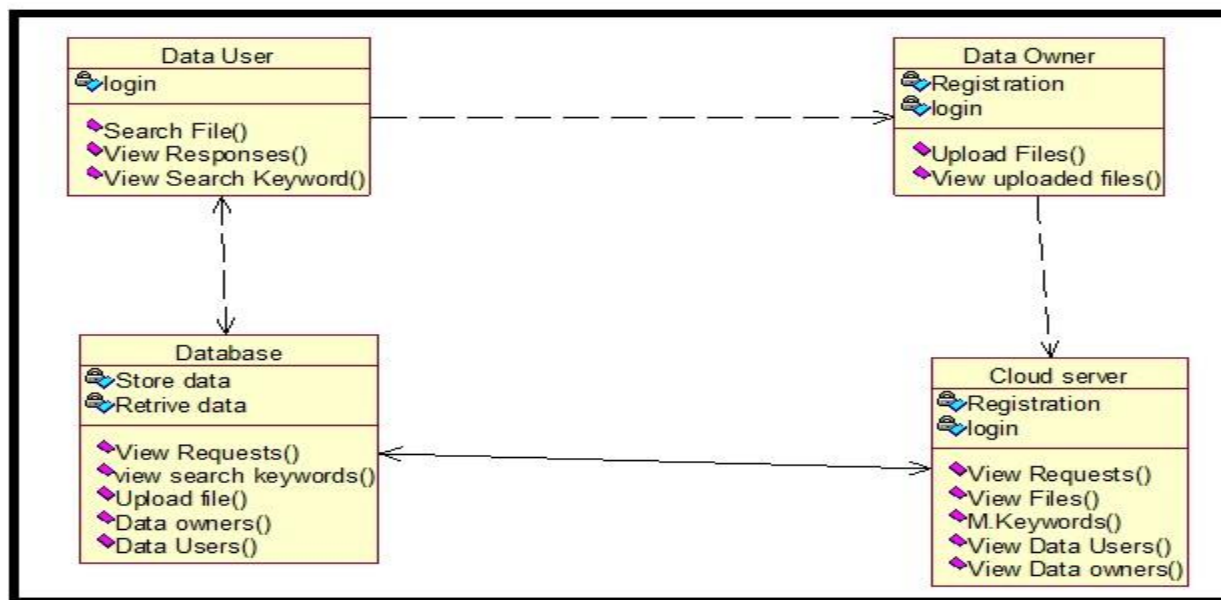


Figure 4.1.2: Class Diagram

EXPLANATION:

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project data user has a class it have an attributes and operations. Data user has a login is an attributes in the class. Search files, view responses and view search keywords this all are operations. Data owner is a class. Registration and login are the attributes. Upload files and View Uploaded files these are the are the operations. Cloud server and database with the linking together all information's storing at a database.

4.1.3 OBJECT DIAGRAM

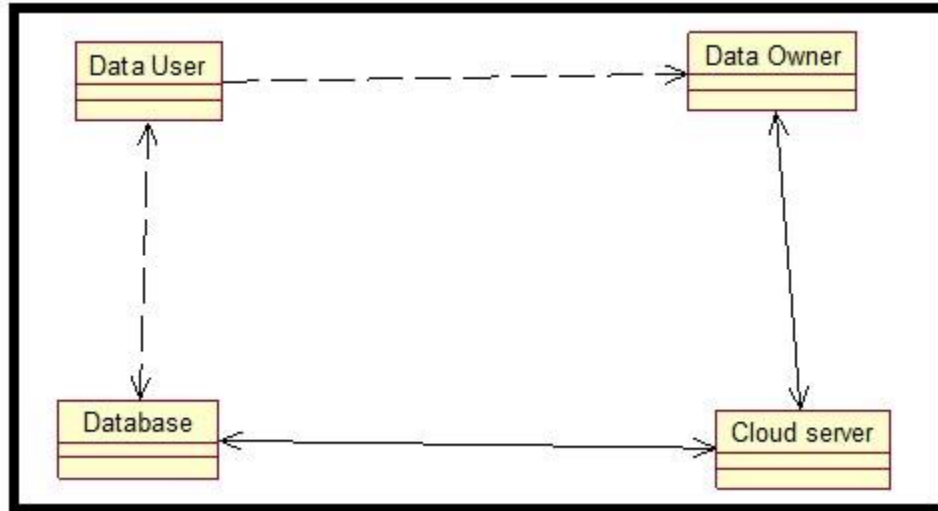


Figure 4.1.3: Object Diagram

EXPLANATION:

In the above diagram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security. Data user is link with a data owner information. Data user and data owner is with a cloud server. All information gather at a database.

4.1.4 STATE DIAGRAM

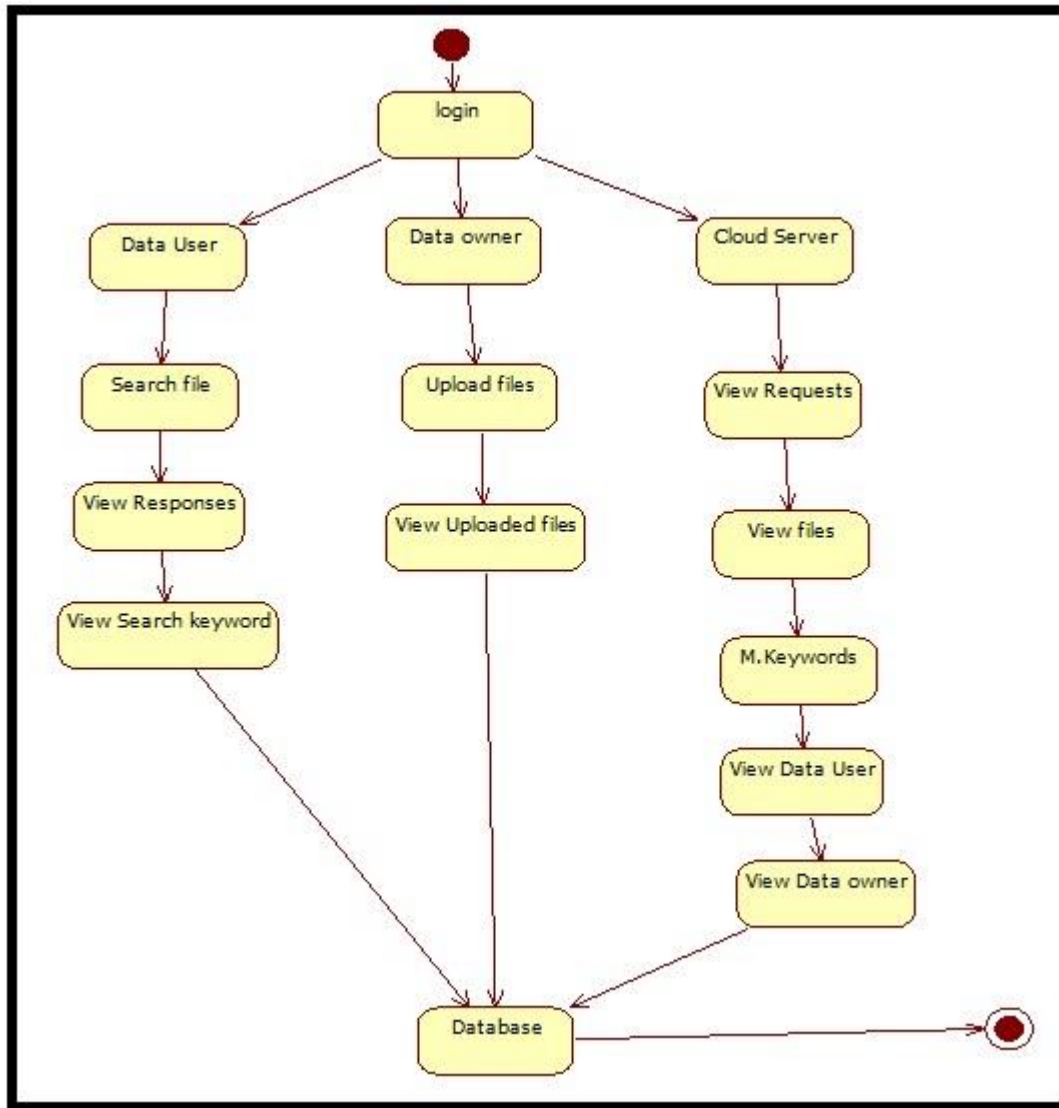


Figure 4.1.4: State Diagram

EXPLANATION:

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described a Data user has a login. After it has a performs an actions. Data owner has a login. Data owner can have an upload data and view uploaded data operation. Cloud server has also a perform an actions it will have an actions on a database.

4.1.5 SEQUENCE DIAGRAM

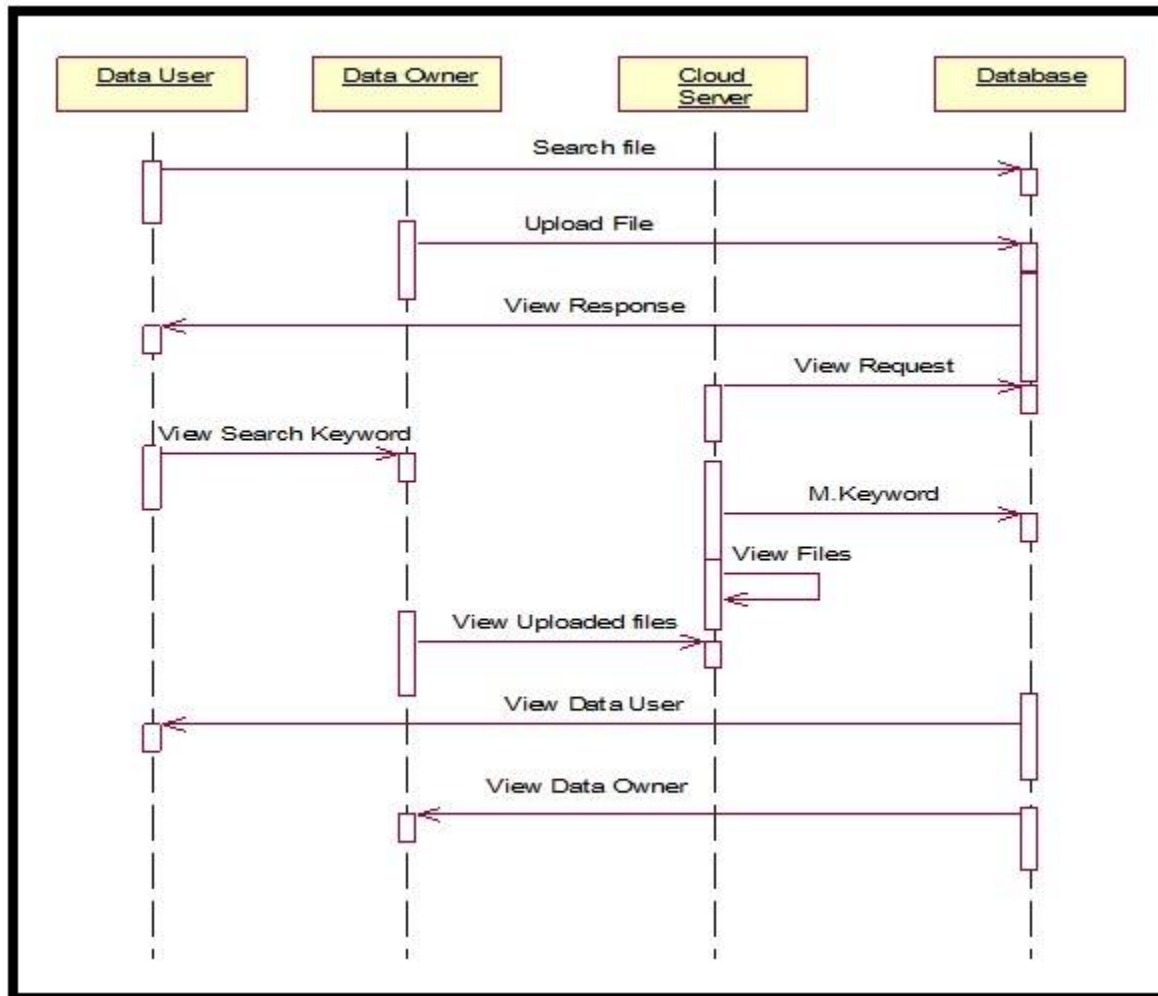


Figure 4.1.5: Sequence Diagram

EXPLANATION:

Sequence diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, sequence diagrams can be sequential. In sequence diagram data user data owner and cloud server are connected with a database. It is used to describe the business and operational step-by-step workflows of components in a system. In the Uml has clearly in sequence order.

4.1.6 COLLABORATION DIAGRAM

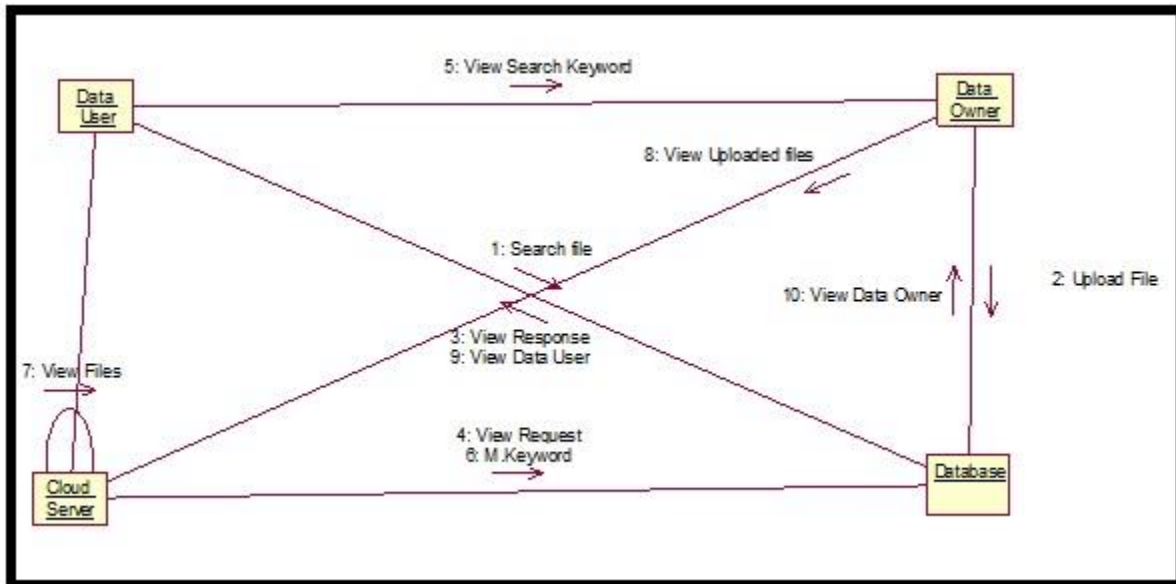


Figure 4.1.6: Collaboration Diagram

EXPLANATION:

A collaboration diagram, also called a communication diagram or interaction diagram, It is also a communicate with a data user data owner with a database. Cloud server has communicate with a database. It is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML).

4.1.7 ACTIVITY DIAGRAM

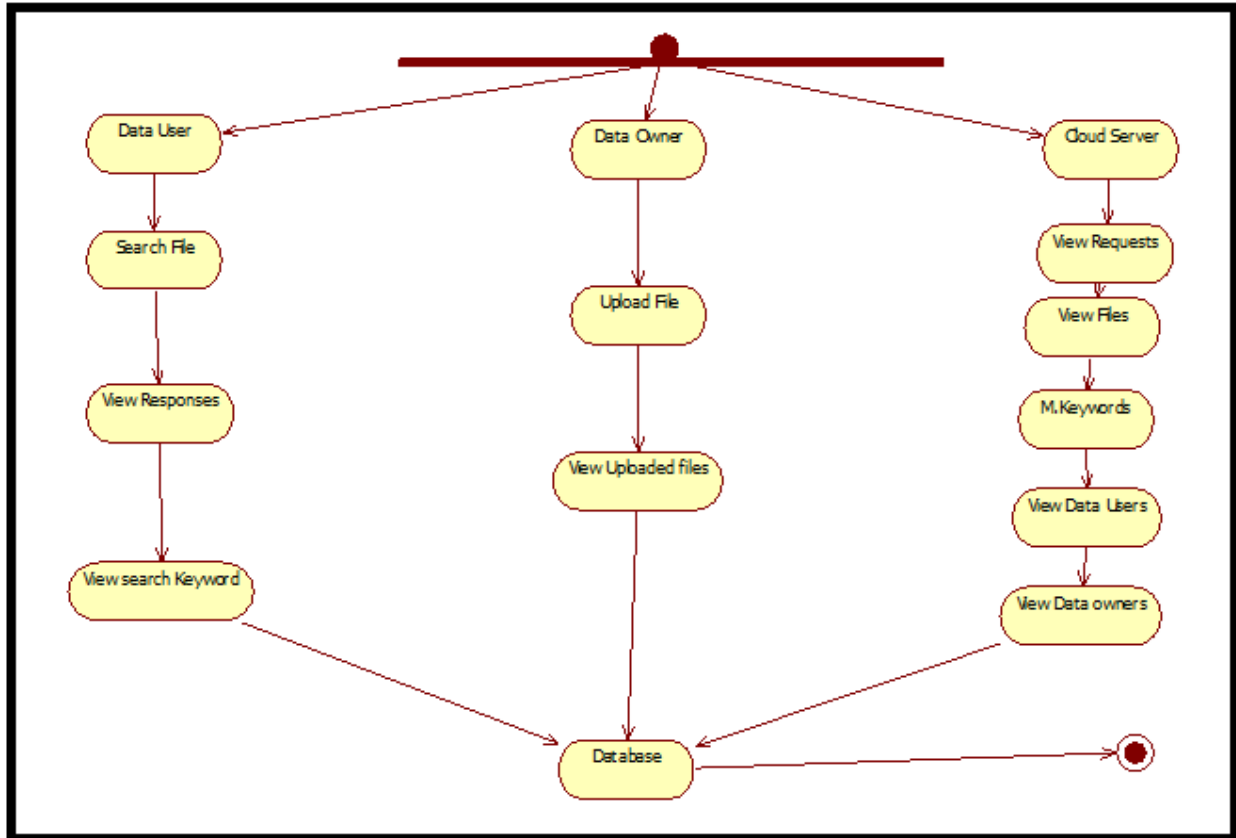


Figure 4.1.7: Activity Diagram

EXPLANATION:

Activity diagrams are graphical representations of workflows of stepwise activities of a data user data owner and cloud server actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. Data user can also a search a file view a responses file and it will search with a keyword.

4.1.8 COMPONENT DIAGRAM

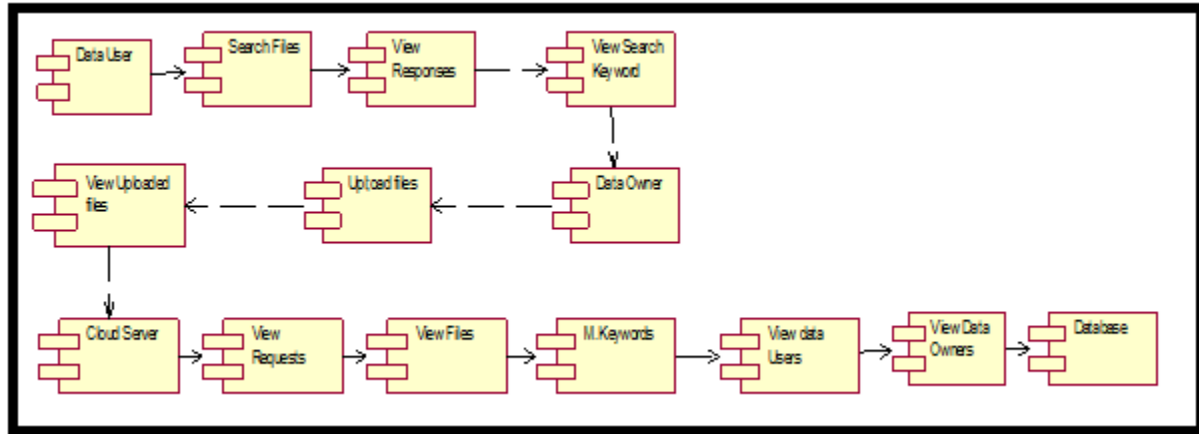


Figure 4.1.8: Component Diagram

EXPLANATION:

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Data user can connect with a data owner information. Data owner can also a depend on a cloud server all store at a database. Results are to be showed to user by database aggregators. All boxes are components and arrow indicates dependencies.

4.1.9 E-R DIAGRAM

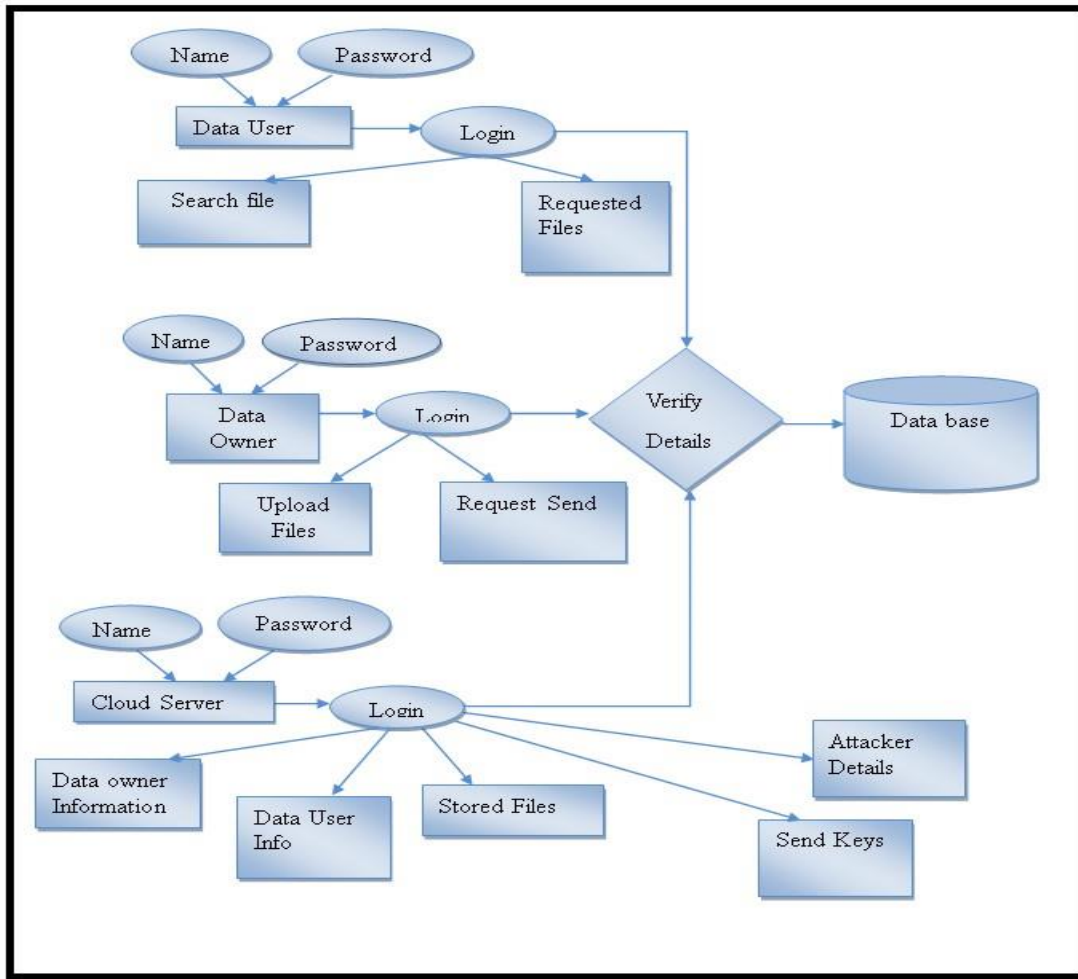


Figure 4.1.9: ER Diagram

EXPLANATION:

Entity-Relationship Model (ERM) is a data user has a username and password if the username and password has a correct then it will login. After login data user can search file and have a requested file. Data owner has a username and password the username and password has correct it will verify with a database. Data owner can also have an upload a files and then have a requested send. Cloud server can also have a username and password user name and password has a correct it will verify from the database. Cloud server can also have a data owner information.

4.1.10 DATA FLOW DIAGRAM

Level 0

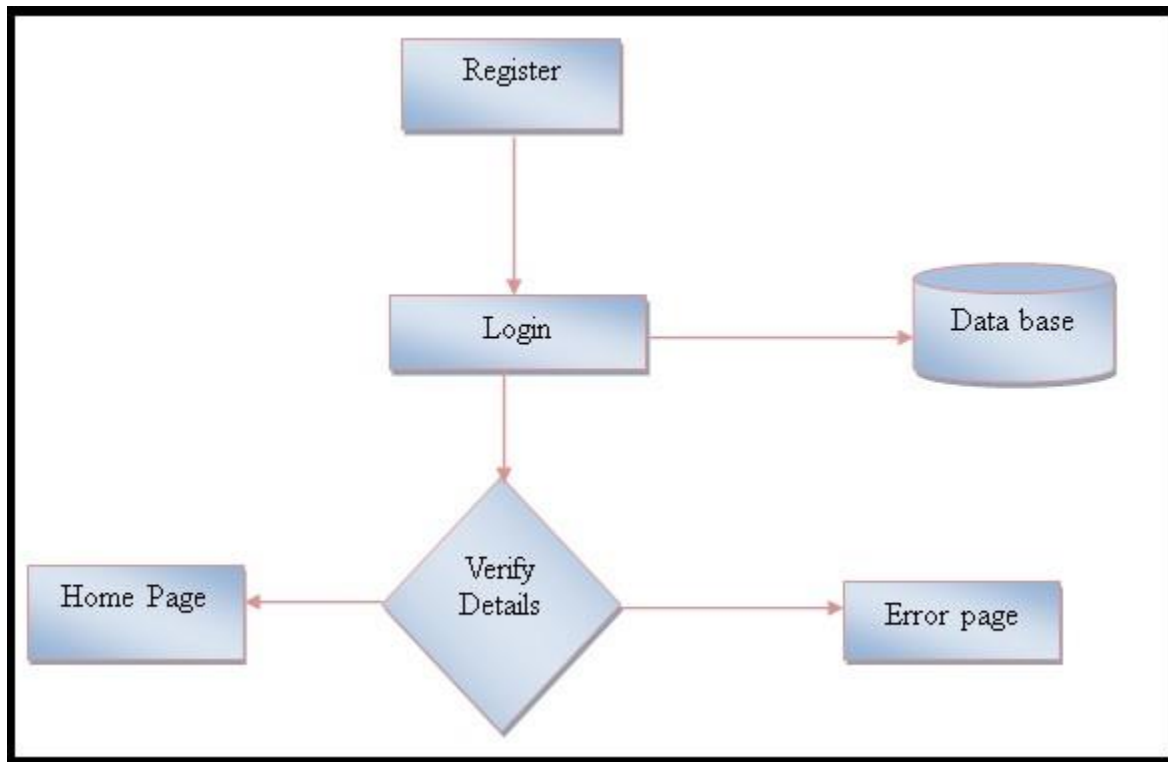


Figure 4.1.10: Data Flow Diagram - level 0

Level 1

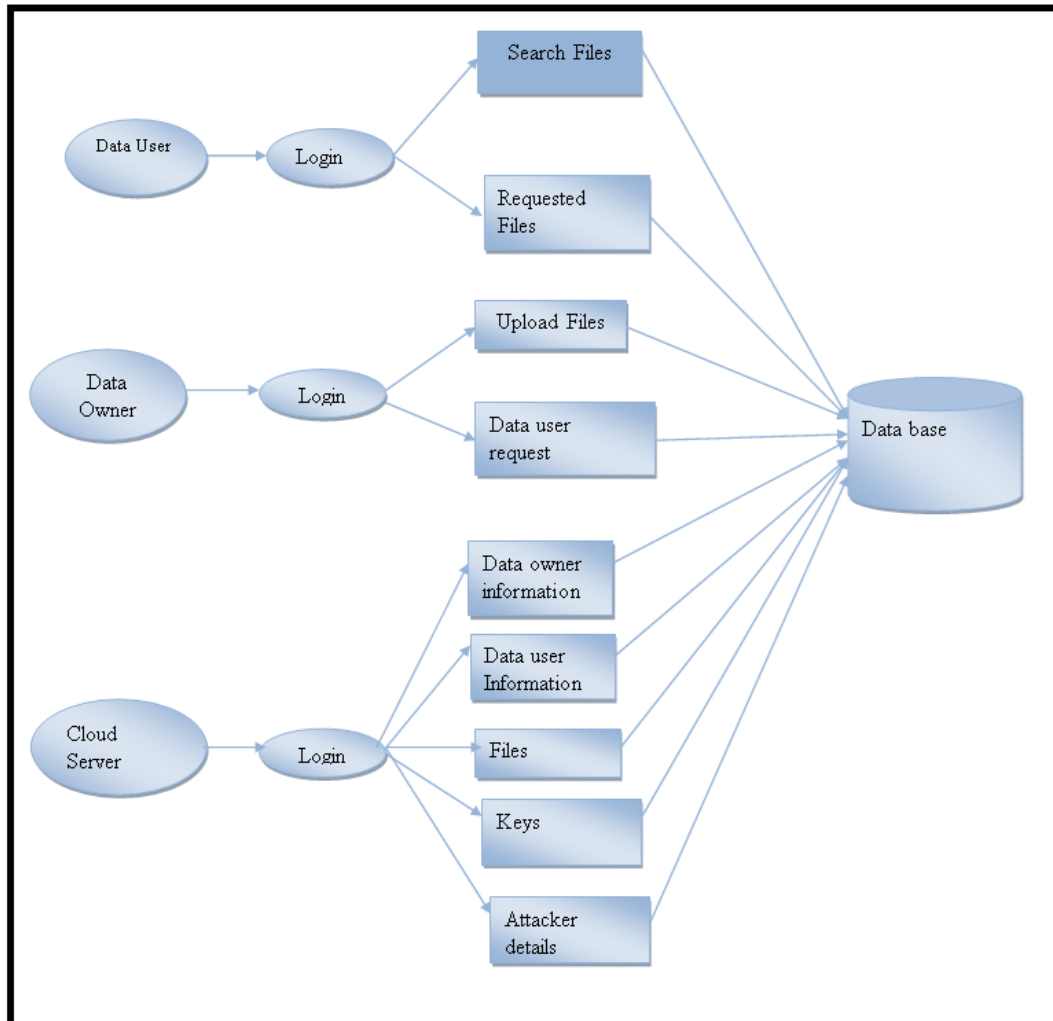


Figure 4.1.10: Data Flow Diagram - level 1

EXPLANATION:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored. Data user can also a login. Data user can have a search keyword and have a requested files. Data owner can also have a login. Data owner can also have an uploaded file and data user request. Cloud server can also a login. Cloud server can also have a data owner information data users information data .Cloud server can also have a keys. Cloud server can also have a file attacker details. All information is gather and store at a database.

4.1.11 DEPLOYMENT DIAGRAM

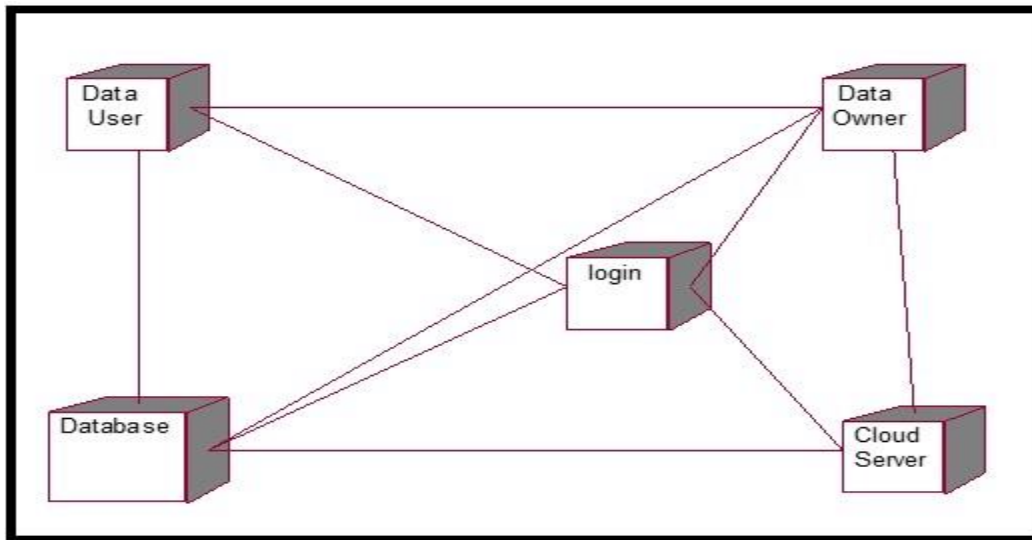


Figure 4.1.11: Deployment Diagram

EXPLANATION:

Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. Data user can first login. Afterwards it will connect with a database.

4.2 SYSTEM ARCHITECTURE

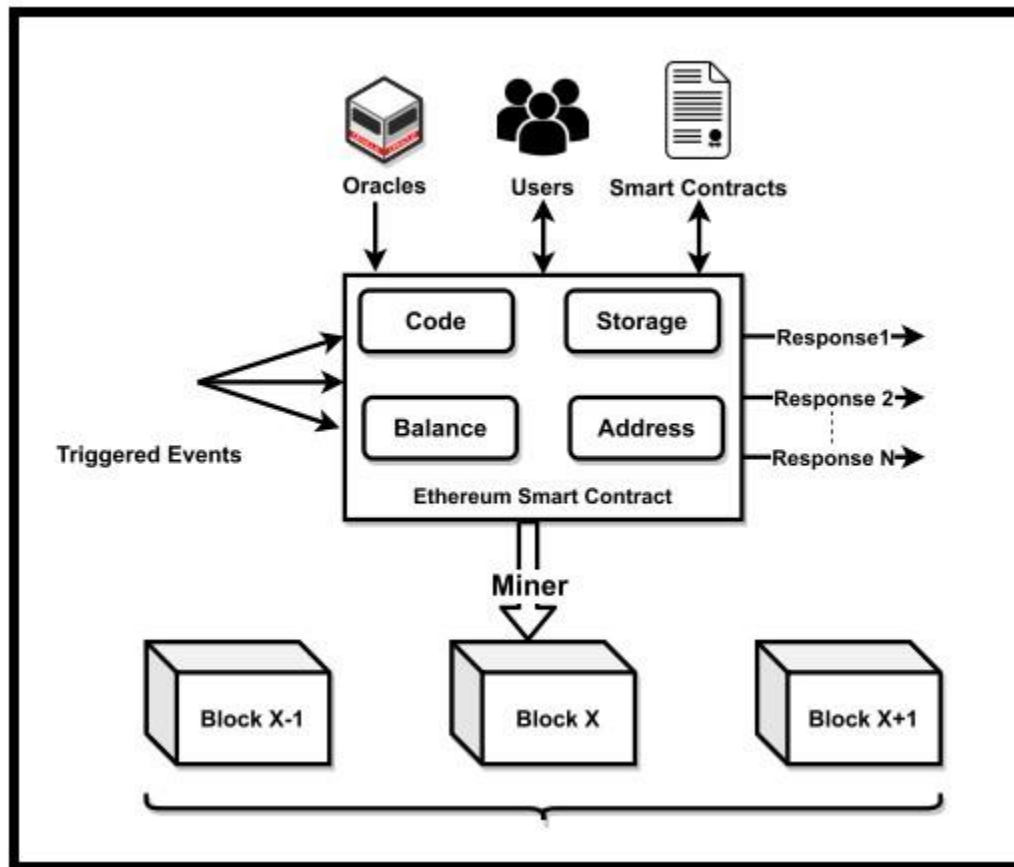


Figure 4.2: System Architecture

EXPLANATION

In this project data owner has a register all details and then login. Data owner can be an upload a document. Data owner can have a send request to the data user. Data user can search a query with uploaded document. The file has also a download it will show an encryption format. Data user also a send a request to the cloud server. Cloud server can a login. It will accept a key approve. Cloud server can also see all the data information's. Cloud server can also see all the user information. Cloud server can see all the stored information. Cloud server can approve a key request from the user. Then data owner has get the request data owner can send a secret key to the user. Then user can also download a file. If the user has given wrong keys it gets warning the user has a block permanently. The file it gets an attacks.

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

Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web 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 versatility, efficiency, and portability, Java has become invaluable to developers by enabling them to:

- Write software on one platform and run it on virtually any other platform
- Create programs to run within a Web browser and Web services
- 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.

OBJECTORIENTED

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 additional features as needed.
2. **Encapsulation**: It is the mechanism of combining the information and providing the abstraction.
3. **Polymorphism**: As the name suggests 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 SWING OVERVIEW

ABSTRACT WINDOW TOOLKIT (AWT) IS CROSS-PLATFORM

Swing provides many controls and widgets to build user interfaces with. Swing class names typically begin with a J such as JButton, JList, JFrame. This is mainly to differentiate them from their AWT counterparts and in general is one-to-one replacements. Swing is built on the concept of Lightweight components vs AWT and SWT's concept of Heavyweight components. The difference between the two is that the Lightweight components are rendered (drawn) using purely Java code, such as draw Line and draw Image, whereas Heavyweight components use the native operating system to render the components. Some components in Swing are actually heavyweight components. The top-level classes and any derived from them are heavyweight as they extend the AWT versions. This is needed because at the root of the UI, the parent windows need to be provided by the OS. These top-level classes include JWindow, JFrame, JDialog and JApplet. All Swing components to be rendered to the screen must be able to trace their way to a root window of one of those classes.

NOTE: It generally it is not a good idea to mix heavyweight components with lightweight components (other than as previously mentioned) as you will encounter layering issues, e.g., a lightweight component that should appear "on top" ends up being obscured by a heavyweight component. The few exceptions to this include using heavyweight components as the root pane and for popup windows. Generally speaking, heavyweight components will render on top of lightweight components and will not be consistent with the look and feel being used in Swing. There are exceptions, but that is an advanced topic. The truly adventurous may want to consider reading this article from Sun on mixing heavyweight and lightweight components.

5.2.4 EVOLUTION OF COLLECTION FRAMEWORK:

Almost all collections in Java are derived from the `java.util.Collection` interface. `Collection` defines the basic parts of all collections. The interface states the `add ()` and `remove ()` methods for adding to and removing from a collection respectively. Also required is the `toArray ()` method, which converts the collection into a simple array of all the elements in the collection. Finally, the `contains ()` method checks if a specified element is in the collection. The `Collection` interface is a sub interface of `java.util.Iterable`, so the `iterator ()` method is also provided. All collections have an iterator that goes through all of the elements in the collection. Additionally, `Collection` is a generic. Any collection can be written to store any class. For example, `Collection<String>` can hold strings, and the elements from the collection can be used as strings without any casting required.

There are three main types of collections:

- Lists: always ordered, may contain duplicates and can be handled the same way as usual arrays
- Sets: cannot contain duplicates and provide random access to their elements
- Maps: connect unique keys with values, provide random access to its keys and may host duplicate values

LIST

Lists are implemented in the JCF via the `java.util.List` interface. It defines a list as essentially a more flexible version of an array. Elements have a specific order, and duplicate elements are allowed. Elements can be placed in a specific position. They can also be searched for within the list. Two concrete classes implement List. The first is `java.util.ArrayList`, which implements the list as an array. Whenever functions specific to a list are required, the class moves the elements around within the array in order to do it. The other implementation is `java.util.LinkedList`. This class stores the elements in nodes that each have a pointer to the previous and next nodes in the list. The list can be traversed by following the pointers, and elements can be added or removed simply by changing the pointers around to place the node in its proper place.

SET:

Java's `java.util.Set` interface defines the set. A set can't have any duplicate elements in it. Additionally, the set has no set order. As such, elements can't be found by index. Set is implemented by `java.util.HashSet`, `java.util.LinkedHashSet`, and `java.util.TreeSet`. `HashSet` uses a hash table. More specifically, it uses a `java.util.HashMap` to store the hashes and elements and to prevent duplicates. `Java.util.LinkedHashSet` extends this by creating a doubly linked list that links all of the elements by their insertion order. This ensures that the iteration order over the set is predictable. `Java.util.TreeSet` uses a red-black tree implemented by a `java.util.TreeMap`. The red-black tree makes sure that there are no duplicates. Additionally, it allows Tree Set to implement `java.util.SortedSet`.

The `java.util.Set` interface is extended by the `java.util.SortedSet` interface. Unlike a regular set, the elements in a sorted set are sorted, either by the element's `compareTo()` method, or a method provided to the constructor of the sorted set. The first and last elements of the sorted set can be retrieved, and subsets can be created via minimum and maximum values, as well as beginning or ending at the beginning or ending of the sorted set. The `SortedSet` interface is implemented by `java.util.TreeSet`.

`Java.util.SortedSet` is extended further via the `java.util.NavigableSet` interface. It's similar to `SortedSet`, but there are a few additional methods. The `floor()`, `ceiling()`, `lower()`, and `higher()` methods find an element in the set that's close to the parameter. Additionally, a descending iterator over the items in the set is provided. As with `SortedSet`, `java.util.TreeSet` implements `NavigableSet`.

MAP:

Maps are defined by the `java.util.Map` interface in Java. Maps are simple data structures that associate a key with a value. The element is the value. This lets the map be very flexible. If the key is the hash code of the element, the map is essentially a set. If it's just an increasing number, it becomes a list. Maps are implemented by `java.util.HashMap`, `java.util.LinkedHashMap`, and `java.util.TreeMap`. `HashMap` uses a hash table. The hashes of the keys are used to find the values in various buckets. `LinkedHashMap` extends this by creating a doubly linked list between the elements. This allows the elements to be accessed in the order in which they were inserted into the map. `TreeMap`, in contrast to `HashMap` and `LinkedHashMap`, uses a red-black tree. The keys are used as the values for the nodes in the tree, and the nodes point to the values in the map.

THREAD:

Simply put, a *thread* is a program's path of execution. Most programs written today run as a single thread, causing problems when multiple events or actions need to occur at the same time. Let's say, for example, a program is not capable of drawing pictures while reading keystrokes. The program must give its full attention to the keyboard input lacking the ability to handle more than one event at a time. The ideal solution to this problem is the seamless execution of two or more sections of a program at the same time.

CREATING THREADS

Java's creators have graciously designed two ways of creating threads: implementing an interface and extending a class. Extending a class is the way Java inherits methods and variables from a parent class. In this case, one can only extend or inherit from a single parent class. This limitation within Java can be overcome by implementing interfaces, which is the most common way to create threads. (Note that the act of inheriting merely allows the class to be run as a thread. It is up to the class to `start ()` execution, etc.)

Interfaces provide a way for programmers to lay the groundwork of a class. They are used to design the requirements for a set of classes to implement. The interface sets everything up, and the class or classes that implement the interface do all the work. The different set of classes that implement the interface have to follow the same rules.

5.3 CONCLUSION

Swing's high level of flexibility is reflected in its inherent ability to override the native host operating system (OS)'s GUI controls for displaying itself. Swing "paints" its controls using the Java 2D APIs, rather than calling a native user interface toolkit. The Java thread scheduler is very simple. All threads have a priority value which can be changed dynamically by calls to the `threads set Priority ()` method. Implementing the above concepts in our project to do the efficient work among the Server.

CHAPTER 6

IMPLEMENTATION

6.1 GENERAL

The Implementation is nothing but source code of project.

6.2 IMPLEMENTATION:

CODING:

```
package com.controller;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

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;
```

```

import weka.classifiers.Evaluation;

import weka.classifiers.trees.J48;

import weka.core.Instances;

/**
 * Servlet implementation class Result
 */
@WebServlet("/Result")

public class Result extends HttpServlet {

    private static final long serialVersionUID = 1L;


    public Result() {

        super();

    }


    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

```

```

// Try block to check for exceptions

try {

    // Create J48 classifier by

    // creating object of J48 class

    J48 j48Classifier = new J48();


    // Dataset path

    String breastCancerDataset

        = "C:\\Users\\KISHAN KUMAR\\eclipse-
workspace\\VTJDM12_2022\\WebContent\\dataset\\breast-cancer.arff";


    // Creating bufferedreader to read the dataset

    BufferedReader bufferedReader

        = new BufferedReader(

            new FileReader(breastCancerDataset));

```

```

// Create dataset instances

Instances datasetInstances

    = new Instances(bufferedReader);

// Set Target Class

datasetInstances.setClassIndex(

    datasetInstances.numAttributes() - 1);

// Evaluating by creating object of Evaluation

// class

Evaluation evaluation

    = new Evaluation(datasetInstances);

// Cross Validate Model with 10 folds

evaluation.crossValidateModel(

    j48Classifier, datasetInstances, 10,

    new Random(1));

System.out.println(evaluation.toSummaryString("\nResults", false));

String s=evaluation.toSummaryString("\nresults", false);

request.setAttribute("message", s);

```

```
}
```

```
// Catch block to handle the exceptions
```

```
catch (Exception e) {
```

```
    // Print message on the console
```

```
    System.out.println("Error Occurred!!!! \n"
```

```
        + e.getMessage());
```

```
}
```

```
request.getRequestDispatcher("index.jsp?page=result").forward(request,  
response);
```

```
}
```

```
}
```

CHAPTER 7

SNAPSHOTS

7.1 GENERAL:

This project implements like web application using COREJAVA and the Server process is maintained using the SOCKET & SERVERSOCKET and the Design part is played by Cascading Style Sheet.

7.2 VARIOUS SNAPSHOTS

7.2.1 Index Page:

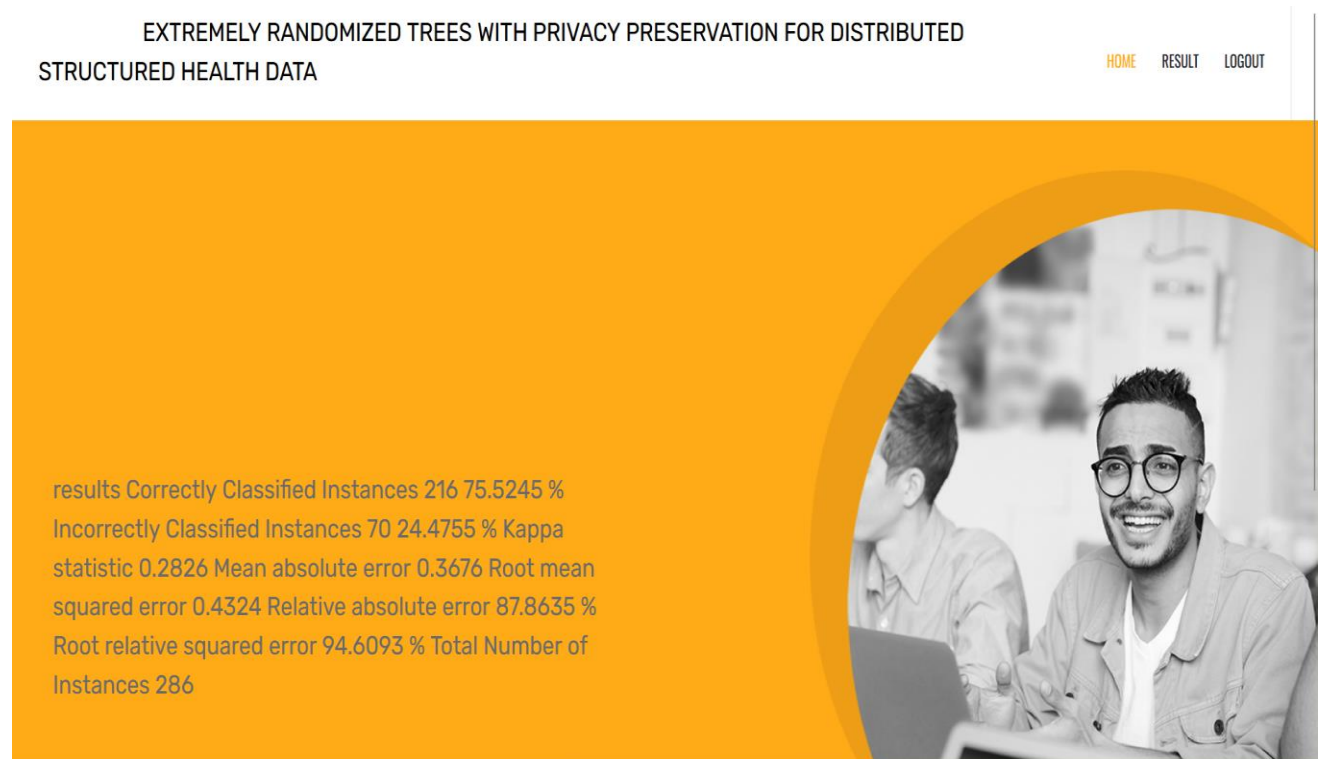



Fig 7.2.1 Index Page

7.2.2 Data Owner Registration Page:



REGISTRATION

Hospital Name

Hospital Address

Email ID

Password

[LOGIN](#)

[Register](#)

Fig 7.2.2 Data Owner Registration Page

7.2.3 Data Owner Login Page

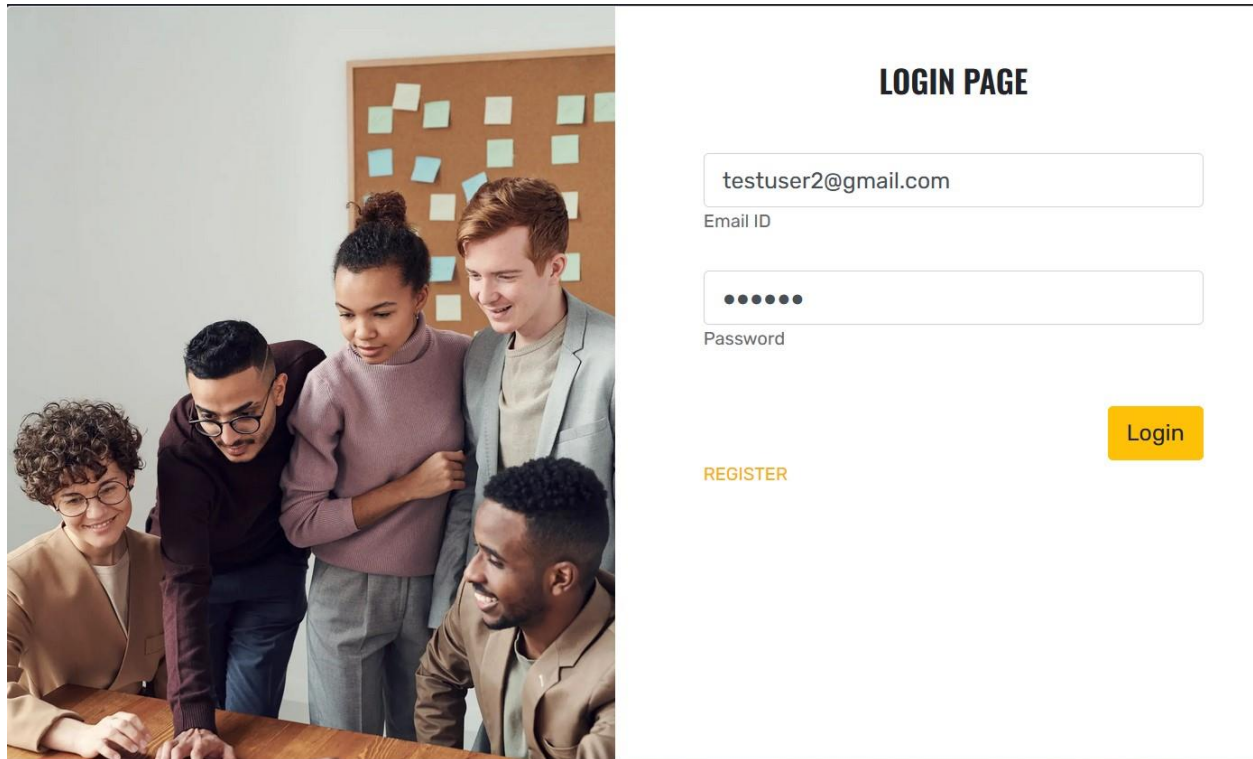


Fig: 7.2.3 Data Owner Login Page

7.2.4 DataSet Page:

The screenshot displays the SQLyog Community application window. The title bar reads 'SQLyog Community - [New Connection/vtjdm12_2022 - root@localhost]'. The menu bar includes File, Edit, Favorites, Database, Table, Others, Tools, Powertools, Window, and Help. The toolbar contains various icons for database operations. The left sidebar shows a tree view of the database structure for 'root@localhost', including schemas like 'cropinfo', 'information_schema', 'mysql', 'performance_schema', 'test', and 'vtjdm12_2022'. Under 'vtjdm12_2022', there are folders for 'Tables', 'Views', 'Stored Procs', 'Functions', 'Triggers', and 'Events'. The 'Tables' folder is expanded, showing 'hospitaldatasets', 'register', 'Columns', and 'Indexes'. The main pane is titled 'Table Data' and shows the data for the 'register' table in the 'vtjdm12_2022' database. The table has columns: id, hospitalName, emailid, password, and address. The data is displayed in a grid view with 3 rows. The status bar at the bottom indicates 'Ready', '3 row(s)', 'Ln 1, Col 1', and 'Connections: 1'. A link to 'Upgrade to SQLyog Professional/Enterprise/Ultimate' is visible in the bottom right corner.

Reasons for upgrading to Professional/Enterprise/Ultimate: SQL Formatter beautifies your SQL code instantly

Query

1

1 Messages 2 Table Data 3 Info 4 History

Limit rows: 0 # of rows: 100 Refresh

id	hospitalName	emailid	password	address
5	testuser	testuser@gmail.com	123456	testuser
6	testuser1	testuser1@gmail.com	123456	hyderabad
7	testuser2	testuser2@gmail.com	123456	sec
(Auto)	(NULL)	(NULL)	(NULL)	(NULL)

Database: vtjdm12_2022 Table: register

Ready 34°C Mostly sunny 3 row(s) Ln 1, Col 1 Connections: 1 Upgrade to SQLyog Professional/Enterprise/Ultimate

7.2.4 DataSet Page:

CHAPTER 8

SOFTWARE TESTING

8.1 GENERAL

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

8.3 TYPES OF TESTS

Sl. No	Test scenario	User action	Expected result	Actual Result	Remarks
1.	Registration	Users registering into the system.	Register into the system.	Successfully alert registered message.	Pass
2.	Login	1. Entered correct password.	1. Log into the system. 2. Alert generated.	1. Successfully logged in. 2. Successfully generated the alert.	Pass
3.	Data User	Search File, Gets the requests from owner	Messages sending data user alert is generated.	Successfully generated the alert and messages sending	Successful
4.	Data Owner	Upload a Files and send request for user	Data owner has to actions	Successfully generated the alert to data owner message	Successful
4.	Cloud Server	Data owner information, data user information, files, keys and attack details	Messages Alert is generated	Successfully generated the alert for cloud server messages	Successful

8.3.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input 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. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.3.2 FUNCTIONAL TEST

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures: interfacing systems or procedures must be invoked.

8.3.3 SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

8.3.4 PERFORMANCE TEST

The Performance test ensures that the output be produced within the time limits,and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

8.3.5 INTEGRATION TESTING

- Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.
- The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

8.3.6 ACCEPTANCE TESTING

- User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

ACCEPTANCE TESTING FOR DATA SYNCHRONIZATION:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
- The Route add operation is done only when there is a Route request in need
- The Status of Nodes information is done automatically in the Cache Updation process

8.3.7 BUILD THE TEST PLAN

- Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identify the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

CHAPTER 9

APPLICATION

9.1 GENERAL

In this paper, we present the privacy-preserving distributed extremely randomized trees algorithm for learning without privacy concerns in the healthcare domain. We have evaluated our proposed algorithm extensively using two popular structured healthcare datasets and two mental health datasets associated with the Norwegian Introducing Mental health through Adaptive Technology (INTROMAT) project. Our approach outperforms the state of the art in distributed tree-based models by up to 11.2% in terms of F1-score, 11.8% in terms of ACC, and 0.232 in terms of MCC for the Depression augmented dataset, and by up to 12.9% in terms of F1-score, 13.2% in terms of ACC, and 0.261 in terms of MCC for the Psykose augmented dataset. Moreover, we present the implementation of our technique on Amazon’s AWS cloud, as a proof of concept, to evaluate the latency and scalability of our framework. The proposed algorithm has linear overhead with respect to the number of parties and can also handle datasets with missing values. We demonstrated our framework’s efficiency in terms of prediction performance, scalability, and overheads, as well as privacy. The proposed framework provides the possibility of developing high-quality and accurate machine learning models without privacy concerns and is expected to contribute to a better healthcare system in the long term. As future work, we plan to explore the possibility of extending the proposed framework to settings where the parties do not follow the honest-but curious security model, which is beyond the scope of this work. REFER.

9.2 FUTURE ENHANCEMENT

As future work, we plan to explore the possibility of extending the proposed framework to settings where the parties do not follow the honest-but curious security model, which is beyond the scope of this work. REFER

CHAPTER 10

CONCLUSION & REFERENCES

10.1 CONCLUSION:

In this paper, we present the privacy-preserving distributed extremely randomized trees algorithm for learning without privacy concerns in the healthcare domain. We have evaluated our proposed algorithm extensively using two popular structured healthcare datasets and two mental health datasets associated with the Norwegian Introducing Mental health through Adaptive Technology (INTROMAT) project. Our approach outperforms the state of the art in distributed tree-based models by up to 11.2% in terms of F1-score, 11.8% in terms of ACC, and 0.232 in terms of MCC for the Depression augmented dataset, and by up to 12.9% in terms of F1-score, 13.2% in terms of ACC, and 0.261 in terms of MCC for the Psykose augmented dataset. Moreover, we present the implementation of our technique on Amazon’s AWS cloud, as a proof of concept, to evaluate the latency and scalability of our framework. The proposed algorithm has linear overhead with respect to the number of parties and can also handle datasets with missing values. We demonstrated our framework’s efficiency in terms of prediction performance, scalability, and overheads, as well as privacy. The proposed framework provides the possibility of developing high-quality and accurate machine learning models without privacy concerns and is expected to contribute to a better healthcare system in the long term. As future work, we plan to explore the possibility of extending the proposed framework to settings where the parties do not follow the honest-but curious security model, which is beyond the scope of this work. REFER.

10.2 REFERENCES

- [1] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng, “Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network,” *Nature Med.*, vol. 25, no. 1, pp. 65–69, Jan. 2019.
- [2] S. McKinney et al., “International evaluation of an AI system for breast cancer screening,” *Nature*, vol. 577, pp. 89–94, Jan. 2020.
- [3] X. Liu, L. Faes, A. U. Kale, S. K. Wagner, D. J. Fu, A. Bruynseels, T. Mahendiran, G. Moraes, M. Shamdass, C. Kern, J. R. Ledsam, M. K. Schmid, K. Balaskas, E. J. Topol, L. M. Bachmann, P. A. Keane, and A. K. Denniston, “A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: A systematic review and meta-analysis,” *Lancet Digit. Health*, vol. 1, no. 6, pp. e271–e297, Oct. 2019.
- [4] R. Aggarwal, V. Sounderajah, G. Martin, D. S. W. Ting, A. Karthikesalingam, D. King, H. Ashrafian, and A. Darzi, “Diagnostic accuracy of deep learning in medical imaging: A systematic review and meta-analysis,” *npj Digit. Med.*, vol. 4, no. 1, p. 65, Dec. 2021.
- [5] S. D. Lustgarten, Y. L. Garrison, M. T. Sinnard, and A. W. Flynn, “Digital privacy in mental healthcare: Current issues and recommendations for technology use,” *Current Opinion Psychol.*, vol. 36, pp. 25–31, Dec. 2020.
- [6] D. Pascual, A. Amirshahi, A. Aminifar, D. Atienza, P. Ryvlin, and R. Wattenhofer, “EpilepsyGAN: Synthetic epileptic brain activities with privacy preservation,” *IEEE Trans. Biomed. Eng.*, vol. 68, no. 8, pp. 2435–2446, Aug. 2021.
- [7] A. Saeed, F. D. Salim, T. Ozcelebi, and J. Lukkien, “Federated self-supervised learning of multisensory representations for embedded intelligence,” *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1030–1040, Jan. 2021.
- [8] F. Forooghifar, A. Aminifar, and D. Atienza, “Resource-aware distributed epilepsy monitoring using self-awareness from edge to cloud,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 6, pp. 1338–1350, Dec. 2019.
- [9] D. Sopic, A. Aminifar, A. Aminifar, and D. Atienza, “Real-time eventdriven classification technique for early detection and prevention of myocardial infarction on wearable systems,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 5, pp. 982–992, Oct. 2018.

- [10] D. Sopic, A. Aminifar, A. Aminifar, and D. Atienza, “Real-time classification technique for early detection and prevention of myocardial infarction on wearable devices,” in Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS), Oct. 2017, pp. 1–4.
- [11] R. Zanetti, A. Arza, A. Aminifar and D. Atienza, “Real-time EEG-based cognitive workload monitoring on wearable devices,” IEEE Trans. Biomed. Eng., vol. 69, no. 1, pp. 265–277, Jan. 2022. [Online]. Available: <https://ieeexplore.ieee.org/document/9464276>, doi: 10.1109/TBME.2021.3092206.
- [12] F. Emekci, O. D. Sahin, D. Agrawal, and A. El Abbadi, “Privacy preserving decision tree learning over multiple parties,” Data Knowl. Eng., vol. 63, no. 2, pp. 348–361, Nov. 2007.
- [13] J. S. Davis and O. Osoba, “Improving privacy preservation policy in the modern information age,” Health Technol., vol. 9, no. 1, pp. 65–75, Jan. 2019.
- [14] J. Vaidya, B. Shafiq, W. Fan, D. Mehmood, and D. Lorenzi, “A random decision tree framework for privacy-preserving data mining,” IEEE Trans. Dependable Secure Comput., vol. 11, no. 5, pp. 399–411, Sep. 2014.
- [15] P. Jurczyk and L. Xiong, “Distributed anonymization: Achieving privacy for both data subjects and data providers,” in Proc. IFIP Annu. Conf. Data Appl. Secur. Privacy. Berlin, Germany: Springer, 2009. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-03007-9_13
- [16] L. Sweeney, “K-anonymity: A model for protecting privacy,” Int. J. Uncertainty, Fuzziness Knowl.-Based Syst., vol. 10, no. 5, pp. 557–570, Oct. 2002.
- [17] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkatasubramanian, “Ldiversity: Privacy beyond K-anonymity,” ACM Trans. Knowl. Discovery Data, vol. 1, no. 1, p. 3, 2007. [18] N. Li, T. Li, and S. Venkatasubramanian, “T-closeness: Privacy beyond K-anonymity and L-diversity,” in Proc. IEEE 23rd Int. Conf. Data Eng., Apr. 2007, pp. 106–115.
- [19] A. Aminifar, Y. Lamo, K. Pun, and F. Rabbi, “A practical methodology for anonymization of structured health data,” in Proc. 17th Scand. Conf. Health Informat. 2019, pp. 127–133. [Online]. Available: https://ep.liu.se/en/conference-article.aspx?series=eep&issue=161&Article_No=22
- [20] A. Aminifar, F. Rabbi, V. K. I. Pun, and Y. Lamo, “Diversity-aware anonymization for structured health data,” in Proc. 43rd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC), Nov. 2021, pp. 2148–2154.