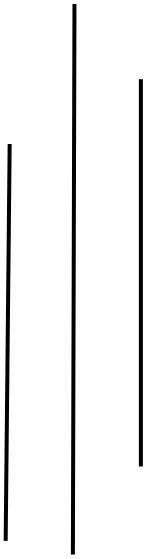
TRIBHUVAN UNIVERSITY INSTITUTE OF ENGINEERINGPULCHOWK CAMPUS



REPORT ON

MITRATA: A SOCIAL NETWORKING SITE

PRESENTED TO

A. PROFESSOR BIBHA STHAPIT

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

BY

AVAHAN TAMRAKAR (077BCT015)

DIVYA SANGKAT KARKI (077BCT028)

KHAGENDRA KARKI (077BCT036)

KRIPESH NIHURE (077BCT037)

ACKNOWLEDGEMENT

First of all, we would like to express our sincere gratitude to our Data Structure and Algorithm teacher, Asst. professor Bibha Sthapit for encouraging us to work on a project work which would enhance our skills and our knowledge related to the Data Structure and Algorithm. Also, we would like to extend our sincere esteems to the Department of Electronics and Computer for providing us opportunity of collaborative undertaking which has helped us to implement the knowledge as project for second year, and developing project of our own which will greatly enhance our knowledge and provide us a new experience of teamwork.

With this project, we were able to apply our knowledge/skill of programming into a real-world problem, more specifically the use of data structures and algorithm. Also, from this project, we got to know some unique idea about project from our friends' side too. It is because of those free and open source resource that many of our work became extremely easier.and those tedious bugs could be solved with ease.

Any kind of suggestion or criticism will be highly appreciated and we will try to address it as far as applicable.

TABLE OF CONTENTS

Objective

This project did not aim to advance our Web development skills within a matter of days or weeks. Instead, the major objective of this project was to encourage us to implement data structures and algorithms into real life problems.The main objectives of the project were:

- To learn efficient data management using LSM tree.
- To learn the concept of Data Structure and Algorithm
- To learn to implement HTTP and websockets.
- To learn to use front end framework like react
- To learn to implement database for an application
- To enhance communication skills and acquire s.kills to work in a teamremotely as well by the help of git

Introduction

The main aim of this project was to deploy something that reflects the application of Data Structures in real life and also to enhance the knowledge of different searching and sorting algorithms that helps in easier data storage and retrieval.

Bringing the concepts of Data Structure and Algorithm and the need for a suitable medium of communication we decided to make a social networking site. We named this site ‘Mitrata’ which translates to friendship in English.

Mitrata is be a web-based app which provides the user with a platform to get to know other people and communicate with them. It is a social networking site like any other sites that exist such as bumble or tinder. The main motive of this app is to help people find a friend and explore if they do have some compatibility or not.

The app allows people who want to use mitrata for making new connections to create a new account by filling out the signup details that include the user credintials, interests,age, gender, a bio that reflects their personality and most imporantly photos. This details of the user is stored in database that is managed by **LSM TREE** for future use . After the user sucessfully creates an acoount, s/he can login to mitrata using the login credintials that was entered during signup. After the user hit login button by filling out the login credintials, the python server that listens on for data recieves the user credintials and searches it in database using **BINARY SEARCH ALGORITHM** for user validation. If the user validation becomes successful, the server sends a pack of data to client side for displaying the user feed as per his/her interest, age and gender.

The feed of the user shows other people who s/he can connect to or be friend of. S/he can choose to select the person as per his/her bio,interests and photos.If the user swipes left on a profile, that profile is declared as disliked by the user. But if the user would like to interact to them or to know more about them, s/he can swipe right on the profile and send friend requests to the person who belong to that profile. This data of likes and dislikes of a user is also stored in real time in database.

Moreover the user feed that is displayed on frontend uses **SINGLY LINEAR LINKED LIST** to store user profiles temporarily within the user login session. It stores two user **profiles** as **node.** We have used linked list with only two nodes because of the heavy data that the node must store that includes people's name,bio,intersets,photos,chat history,likes and dislikes and so on. If more than two node are used, the program becomes memory inefficient. Rather, we considered making two nodes: **One** for displaying first profile in the user feed and the **next node** for holding the data of next profile in the backgroud that will be transferred as first profile after the user swipes the previous first profile. Simultaneously, a function is called that asks backend to send new profile from the database to store into the next node.
After the user sends friend request to the person s/he likes, that person gets a notification asking if s/he also would like to know about to talk to the user. If that person accepts the friend request, their profiles are linked up for chatting in the chat section.

In this way, a new person that enters **Mitrata** with the motive of finding new friend or making new connection can start to talk to someone and find out more about each other.

Linked List

A linked list is a data structure used to store a collection of elements. It consists of a sequence of nodes, where each node stores an element and a reference to the next node in the sequence. The first node is called the head, and the last node is called the tail, and the tail points to a null node.

The advantages of using a linked list include:

1. Dynamic size: A linked list can grow or shrink as needed, making it more flexible than other data structures like arrays, which have a fixed size.
2. Easy insertion and deletion: Adding or removing an element in a linked list only requires changing a few references, whereas with an array, it can require shifting all the elements in the array.
3. Efficient memory usage: Unlike arrays, which require contiguous memory allocation, linked lists can be implemented using non-contiguous memory, making it possible to store elements in memory that is not being used by other data structures.
4. Versatility: Linked lists can be used to implement other data structures, such as stacks, queues, and hash tables.

However, linked lists also have some disadvantages, including:

1. Slower access time: Accessing an element in a linked list can take longer than accessing an element in an array, as each node must be traversed to reach the desired element.
2. More memory overhead: Each node in a linked list requires additional memory to store the reference to the next node, which can increase memory usage compared to other data structures.
3. No random access as in arrays

Linked List in Mitrata

We have used a singly linear linked list to store user profiles temporarily within the user login session. It stores two user profiles as node. We have used linked list with only two nodes because of the heavy data that the node must store that includes people's name,bio,intersets,photos,chat history,likes and dislikes and so on. If more than two node are used, the program becomes memory inefficient. Rather, we considered making two nodes:

- One for displaying first profile in the user feed
- next node for holding the data of next profile in the backgroud that will be used after the user swipes a profile.

Algorithm

Each time the user swipes on a profile(either left or right), two tasks are performed:

- First node is replaced by next node
- A function call asks new profile data from backend
- Next node is filled by newly retrieved data.

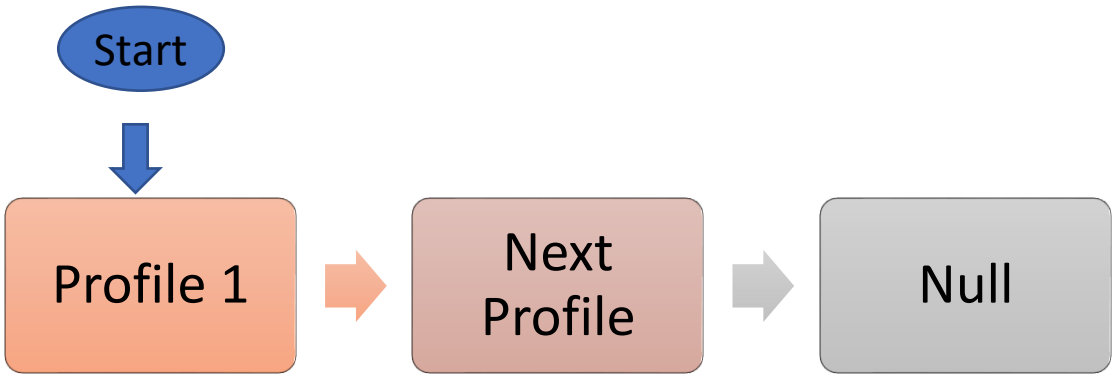


Fig: Block diagram of singly linear linked list

as implemented in Mitrata

LogStructured Merge(LSM) Tree:

LSM (Log-Structured Merge) tree is a data structure used to store and manage large amounts of data efficiently. LSM trees are commonly used in modern databases like Cassandra, HBase, and LevelDB.

Advantages of LSM tree:

1. Fast writes: LSM trees optimize for fast writes by buffering new data in memory before flushing it to disk in large sequential writes. This reduces the number of random writes to the disk, which can be very slow, and increases write throughput.

2. Efficient storage: LSM trees use a tiered storage structure where data is stored in multiple levels, with the newest data in memory and older data on disk. This allows for more efficient use of storage space, as newer data that is frequently accessed is kept in memory while older data that is less frequently accessed is stored on disk.

3. Scalability: LSM trees can scale to handle large amounts of data by partitioning the data into multiple files, each with its own index. This allows for parallelism when reading and writing data, which can significantly improve performance.

LSM tree can be used to:

1. Write buffering: New data is initially buffered in memory before being written to disk. This improves write throughput by reducing the number of random disk writes.

2. Compaction: As data accumulates on disk, LSM trees periodically merge small files into larger ones to reduce the number of files that need to be searched during reads. This process is called compaction, and it ensures that the tree remains efficient even as data grows.

3. Indexing: LSM trees use a variety of indexing techniques, such as B-trees or Bloom filters, to quickly locate data on disk.

4. Fault tolerance: LSM trees support replication and fault tolerance by using multiple copies of data across different nodes or disks. This helps ensure that data is not lost in the event of hardware failures.

5. Read performance: LSM trees can also be optimized for read performance by using bloom filters to reduce the number of disk reads required to locate a key in the tree.

Overall, LSM trees are a powerful data structure that offers fast writes, efficient storage, and scalability for large-scale data management.

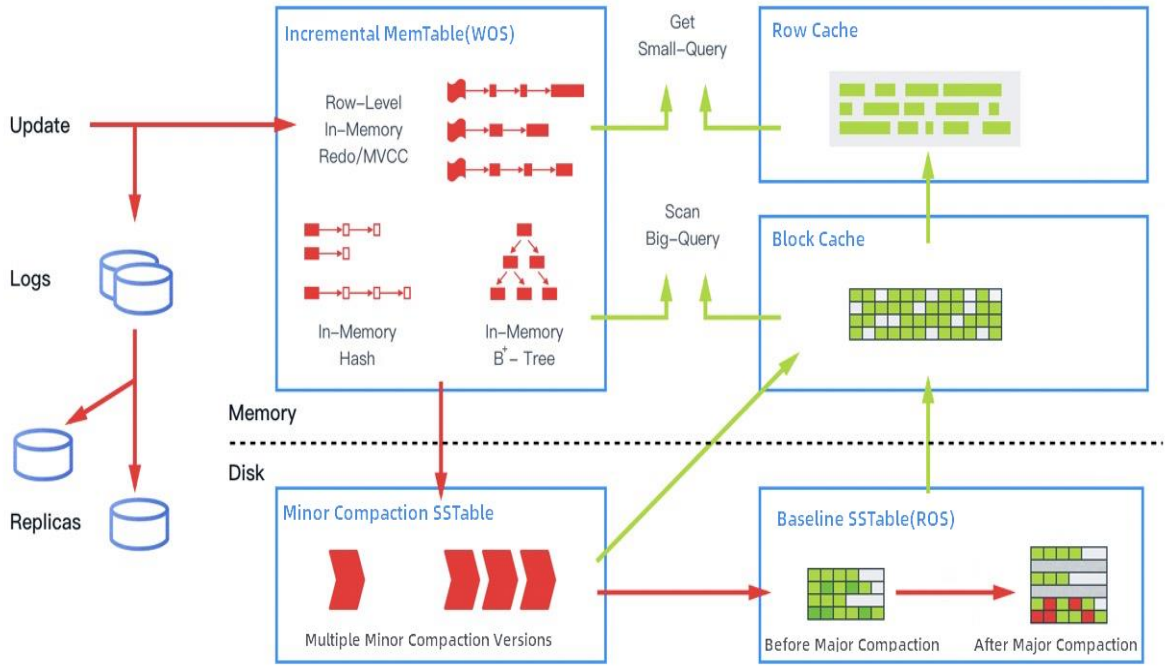


Fig: Architecture of an LSM Tree

LSM Tree in Mitrata

Mitrata uses Three LSM trees, one for storing User Profile, one for storing Notifications and one for storing chat.

Memtable: Memtable which is how the data is stored in the memory initially, is implemented as a Red Black Tree. The use of Red black tree is because the insertion and deletion is faster in it

While flushing the Memtable to the disk the in order traversal of the Red black tree gives the data in sorted form so no further operation is required

Bloom Filter: Bloom filter optimizes the application by saving the time on useless search (search for keys in the database which doesn't exist)

Based on the probability of the false positive (an affirmation for keys which doesn't exist) and the number of data to be inserted we can determine the number of hash function and the data size of the bucket required.

For our application, we have set the false positive to 0.2 and the number of data to be inserted is 100000(rough figure) the data size of the bucket and the number of hash function is 410kb and 2 respectively.

Write Ahead Log : All the data which are to be stored in the database are first temporarily stored in a back up file. This is done so that the system is resilient in the face of a power failure or other unexpected error. This data can then be restored back.

Other DSA elements used

- Chat using Stack

  A stack is a linear data structure that follows the Last-In-First-Out (LIFO) principle. This means that the last element added to the stack is the first one to be removed. A stack has two primary operations: push and pop. When a new element is added to the stack, it is pushed onto the top of the stack. When an element is removed from the stack, it is popped off the top of the stack.

  Stack can be used to store a chat conversation in a chat application. Each message sent and received in the conversation can be pushed onto the top of the stack, and when a message is deleted or popped from the stack, the most recent message is removed.

  Using a stack for storing chat messages allows for easy retrieval of the most recent messages in the conversation. When a user opens the chat window, they can immediately see the most recent messages that have been sent and received.

- Red Black Tree
- Hash tables:

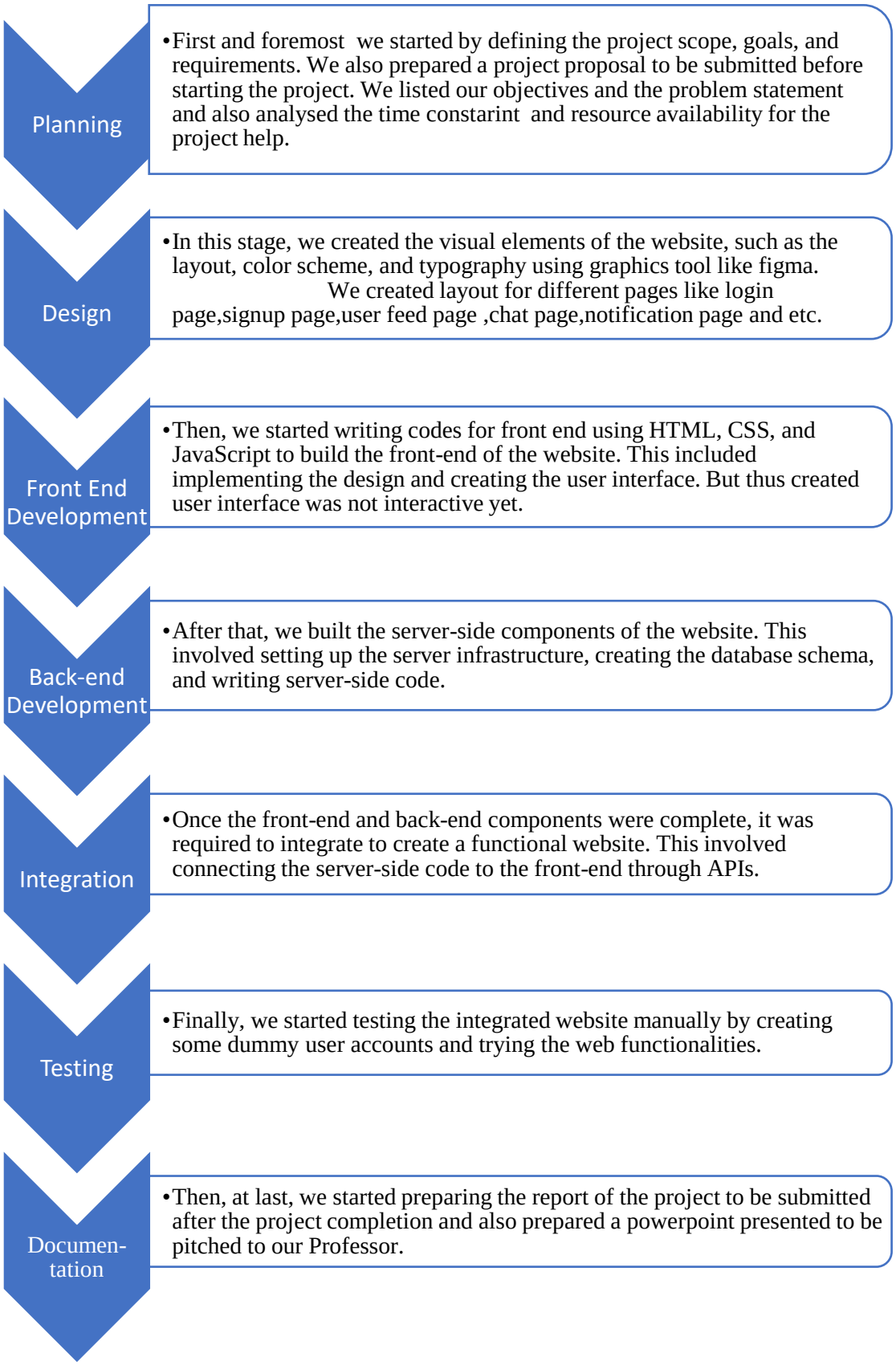
  for storing user information, mapping user IDs to chat sessions, etc.
- Search algorithms:

  for finding and retrieving specific chat history or user information.
- Sorting algorithms:

  for organizing and displaying chat history in a specific order (e.g. by date, by sender, etc.).
- Encryption/decryption algorithms:

  for securing the communication between the clients and the server, protecting sensitive information such as user passwords.

System design flow



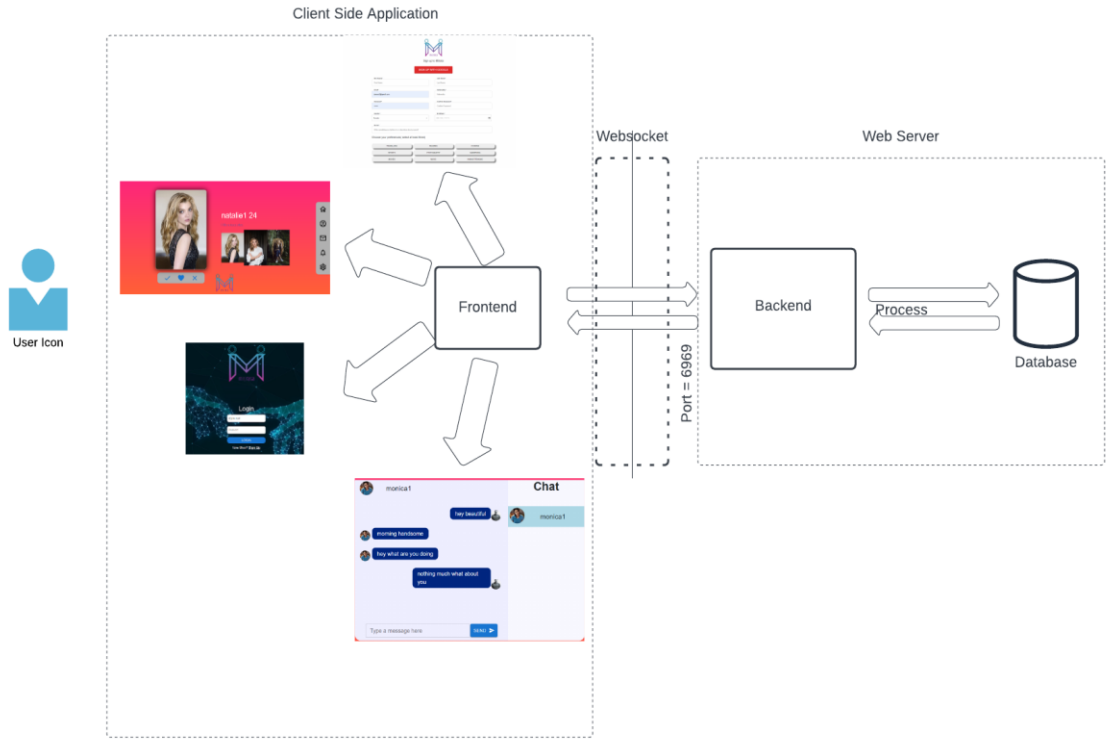
Elements of Mitrata

Web development involves a combination of programming, design, and content creation to produce a functional and visually appealing website.Web development can be divided into following main categories:

- Front-end development involves creating the user interface of a website or web application, including the layout, design, and interactive elements. This typically involves using HTML, CSS, and JavaScript, as well as various front-end frameworks and libraries, like we used **react**.
- Back-end development, on the other hand, involves the server-side of a website or web application, including the database and server-side scripting. This typically involves using languages like **Python**,as in our case.
- Databases are used to store and manage large amounts of data for web applications. Web developers use database management systems like MySQL, PostgreSQL, MongoDB, or Oracle to design, create, and maintain databases, as well as to write SQL queries to retrieve and manipulate data.

  \We have stored data locally for now and used **Log Structured Merge (LSM) Tree to manage (search/sort/easy retrieval)** data in database.

Web servers are responsible for delivering web pages and content to clients that request them. Popular web servers include Apache, Nginx, and Microsoft IIS, which are used to host websites and web applications.



Future Enhancements

Due to the limited time constraint, we couldnot manage to add few features that we had planned. Here are a few things that we will be trying to accomplish in the near future:

1. Implementing remote database
2. Enhancing security and user privacy
3. Deploying the site
4. Developing an mobile app for both IOS and Android
5. Replace the friend request feature with literal matching algorithm

Conclusion

'Mitrata' is a comprehensive and well-planned initiative undertaken to address the need of a simple and easy to use social networking site. We have carefully considered all aspects of the project and came up with a prototype that may require further enhancements before deploying.

In this project, we have explored various data structures and algorithms commonly used in real world. We have learned how to implement these data structures and algorithms in code, and we have also gained insights into their strengths and limitations.

We have found that DSA is a fundamental area of computer science that plays a crucial role in solving complex problems efficiently. By applying DSA concepts, we can develop software systems that are faster, more efficient, and more scalable. Overall, this project has provided us with a solid foundation in DSA and has enabled us to apply these concepts to real-world programming problems. We believe that our findings and outcomes will be valuable for anyone interested in computer science, and we hope that this report will inspire others to explore this fascinating area of study further.