Privacy Network: Design and Implementation of a Security-Conscious, Location-Preserving Geosocial Network

Thesis submitted by

Manas Pratim Biswas (002011001025)
Anumoy Nandy (002011001121)
Kunal Pramanick (302111001005)

under the guidance of

Dr. Munmun Bhattacharya

in partial fulfilment of the requirements
for the award of the degree of

Bachelor of Engineering in Information Technology

Department Of Information Technology
Faculty of Engineering & Technology
Jadavpur University

2020 - 2024

This page has been intentionally left blank

BONDAFIDE CERTIFICATE

This is to certify that this project titled ***"Privacy Network: Design and Implementation of a Security-Conscious, Location-Preserving Geosocial Network"***, submitted to the **Department of Information Technology, Jadavpur University, Salt Lake Campus, Kolkata**, for the award of the degree of **Bachelor of Engineering**, is a bonafide record of work done by **Manas Pratim Biswas (Regn. No.: 153760 of 2020-2021)**, **Anumoy Nandy (Regn. No.: 153823 of 2020-2021)** and **Kunal Pramanick (Regn. No.: 159991 of 2021-2022)**, under my supervision from 03/07/2023 to 15/05/2024.

Countersigned By:

Dr. Munmun Bhattacharya
Assistant Professor
Information Technology

Dr. Bibhas Chandra Dhara
HOD, Information Technology
Jadavpur University

Acknowledgements

We would like to express our sincere gratitude to Dr. Munmun Bhattacharya for allowing us to pursue our final-year project under her supervision. In addition to the meticulous research guidance, her encouraging support for our ideas as well as the constructive criticisms during our experimental and initial developmental stages of the project would not have been possible.

We would also like to thank the entire Department of Information Technology inluding all the professors, lab assistants and non-teaching staff for being extremely cooperative with us throughout the developmental stages of the product - *Privacy Network*.

Above all, we thank our parents for their support and encouragement in our academic pursuits.

Department of Information Technology
Jadavpur University
Salt Lake Campus, Kolkata

Manas Pratim Biswas Anumoy Nandy Kunal Pramanick

JADAVPUR UNIVERSITY

Dept. of Information Technology

Vision:

To provide young undergraduate and postgraduate students a responsive research environment and quality education in Information Technology to contribute in education, industry and society at large.

Mission:

M1: To nurture and strengthen the professional potential of undergraduate and postgraduate students to the highest level.

M2: To provide international standard infrastructure for quality teaching, research and development in Information Technology.

M3: To undertake research challenges to explore new vistas of Information and Communication Technology for sustainable development in a value-based society.

M4: To encourage teamwork for undertaking real life and global challenges.

Program Educational Objectives (PEOs):

Graduates should be able to:

PEO1: Demonstrate recognizable expertise to solve problems in the analysis, design, implementation and evaluation of smart, distributed, and secured software systems.

PEO2: Engage in the engineering profession globally, by contributing to the ethical, competent, and creative practice of theoretical and practical aspects of intelligent data engineering.

PEO3: Exhibit sustained learning capability and ability to adapt to a constantly changing field of Information Technology through professional development, and self-learning.

PEO4: Show leadership qualities and initiative to ethically advance professional and organizational goals through collaboration with others of diverse interdisciplinary backgrounds.

Mission - PEO matrix:

Ms/ PEOs	M1	M2	M3	M4
PEO1	3	2	2	1
PEO2	2	3	2	1
PEO3	2	2	3	1
PEO4	1	2	2	3

(3 – Strong, 2 – Moderate and 1 – Weak)

Program Specific Outcomes (PSOs):

At the end of the program a student will be able to:

PSO1: Apply the principles of theoretical and practical aspects of ever evolving Programming & Software Technology in solving real life problems efficiently.

PSO2: Develop secure software systems considering constantly changing paradigms of communication and computation of web enabled distributed Systems.

PSO3: Design ethical solutions of global challenges by applying intelligent data science & management techniques on suitable modern computational platforms through interdisciplinary collaboration.

This page has been intentionally left blank

Abstract

The recent advancements in mobile and internet technology, coupled with cheap internet data, have witnessed an exponential increase in the usage of internet and people connecting via social media networks. However, the traditional social networking sites such as *Facebook*, *Instagram* or *Twitter* have had witnessed multiple allegations of capturing and selling the user data for their corporate and business purposes. Most importantly, a breach in the user's location data can expose sensitive information regarding that user's frequent places of visit, acquaintances they meet with, food habits and political preferences.

Our project explores the applicability and implementation of a Privacy Network as a scalable Geosocial Networking website. A user-centric design for a secure location sharing is implemented into our project. The final product, *Privacy Network*, is a web application that allows the users to register into our website with their details, add or remove friends and most importantly, decide and set their visibility and other privacy parameters very precisely to ensure a secure and robust social media usage. Even more, the users can query their friends based on parameters including but not limited to *age*, *gender*, *college* and *distance*.

The user auth and data is handled by a MongoDB backend and the user location attributes are handled by a Postgres backend. PostGIS along with PostgreSQL is utilised to create custom SQL queries to fetch the user location details on-demand. Real-time location broadcasting and multi-casting is achieved by upgrading the *HTTP* protocol to *WebSocket* protocol and utilising raw *WebSockets* natively available in the client's browser. A separate WebSocket backend server handles all the WebSocket connections from the client side. Services such as *Vercel* and *Render* are utilised to host the frontend and backend servers. A complete documentation of all the backend API end-points, in compliance with the standardized OpenAPI specifications is available as a Swagger documentation.

However, scaling WebSockets is challenging. Therefore, a *distributed Redis Pub-Sub* based architecture along with a *HAProxy load balancer* is proposed. Moreover, a *change-data-capture* mechanism using *Apache Kafka* is proposed to keep the MongoDB and PostgreSQL databases consistent and in synchronization with each other.

Keywords: typescript, websockets, mern, postgresql, postgis, redis, haproxy, kafka, social-network-analysis

Contents

List of Tables

List of Figures

This page has been intentionally left blank

Chapter 1

Introduction

This chapter gives a basic introduction about the background of Privacy Network, its necessity and inception. The later subsections gives the consequent flow of the project report.

1.1 Geosocial Networking

Geosocial networking[1] is a type of social networking that integrates geographic services and capabilities such as geolocation to enable additional social dynamics. This technology allows users to connect and interact based on their physical locations, enhancing the social networking experience by facilitating real-time, location-based interactions. Popular applications of geosocial networking include location-based services like check-ins, geotagging, and finding nearby friends or events. These functionalities are supported by technologies such as GPS, Wi-Fi positioning, and cellular triangulation. By leveraging these technologies, geosocial networking can provide users with personalized content and recommendations based on their current location, thereby enriching the user experience. However, it also raises significant privacy concerns, as the collection and sharing of location data can lead to potential risks such as unwanted tracking and profiling. Consequently, there is a growing emphasis on developing secure geosocial networking applications that prioritize user privacy and data protection, incorporating advanced encryption and user consent mechanisms to safeguard sensitive location information.

The past decade has seen unprecedented advancements in mobile and internet technologies, resulting in an exponential increase in internet usage and social media connectivity. Affordable internet data has made the online world more accessible than ever before, fundamentally transforming how people interact and communicate. Social media platforms such as Facebook, Instagram, and Twitter have become integral to daily life, providing users with the means to connect, share, and engage with others on a global scale. These platforms, however, have also come under intense scrutiny for their handling of user data, with numerous allegations of data misuse for corporate gain.

1.1.1 Security Vulnerabilities in Geosocial Networking

The convenience and connectivity offered by social media come at a significant cost to user privacy. Traditional social networking sites have been accused of capturing and monetizing user data, leveraging personal information for targeted advertising and other business

purposes. This practice has raised serious concerns about data security and user consent, highlighting a critical need for more transparent and ethical data management practices.

Among the various types of data collected, location data is particularly sensitive. A breach in location data can reveal detailed insights into a user's daily routines, including frequent places of visit, social circles, dietary preferences, and even political affiliations. For instance, if a social media platform tracks a user's location, it can infer whether the user visits specific types of restaurants, attends political rallies, or frequents particular neighborhoods. Such information, if exposed, can lead to a myriad of privacy invasions and security risks, including unwanted surveillance, targeted harassment, and identity theft.

The exploitation of user data by social media giants has prompted a growing public outcry and a demand for greater privacy protections. High-profile incidents, such as the Cambridge Analytica scandal[2], have underscored the potential for misuse of personal data and the far-reaching consequences of such breaches. These events have sparked a broader conversation about the ethical responsibilities of technology companies and the need for robust regulatory frameworks to safeguard user privacy.

1.1.2 Addressing Privacy Tradeoffs

In response to these concerns, there has been a surge in the development of privacy-centric social networking alternatives. These platforms prioritize user data protection and transparency, offering users greater control over their personal information. By employing advanced encryption techniques and decentralized data storage solutions, these new networks aim to mitigate the risks associated with data breaches and unauthorized data exploitation.

In our current approach, we utilise two security parameters that the user can fine tune to filter his/her location. The user can set their own visibility to either *on/off* and can also *limit their visibility* till a certain range of distance. Thereby, The user can control the location shared to the rest of the users in the social network. We utilise a variable called ***isVisible*** which the user can update so as to control his/her visibility. *isVisible true* implies that the user wishes to share his/her locaitons and *isVisible false* denotes that the user does not wish to share his/her location. The idea is that a particular pair of user can have four possibilities for the privacy parametes as shown below:

A **Marker** shows the exact location of the user's friend since the user's friend as allowed his/her location to be shared.

A **Blue Patch** or a Medium visibility means that the exact location will not be shared to the user's friend, rather a Blue Patch would be shown which denotes the user to be present within 10 *kms* radius.

Table 1.1: Privacy Filtration

Connection	*isVisible*	*VisibilityLevel*
Friend	True	High (Marker)
Friend	False	Medium (Blue Patch)
Non-Friend	True	Low (Red Patch)
Non-Friend	False	None

A **Red Patch** or a Low visibility means that the exact location of the stranger will not be shared to the user, rather a Red Patch would be shown which denotes the user to be present within 20 *kms* radius.

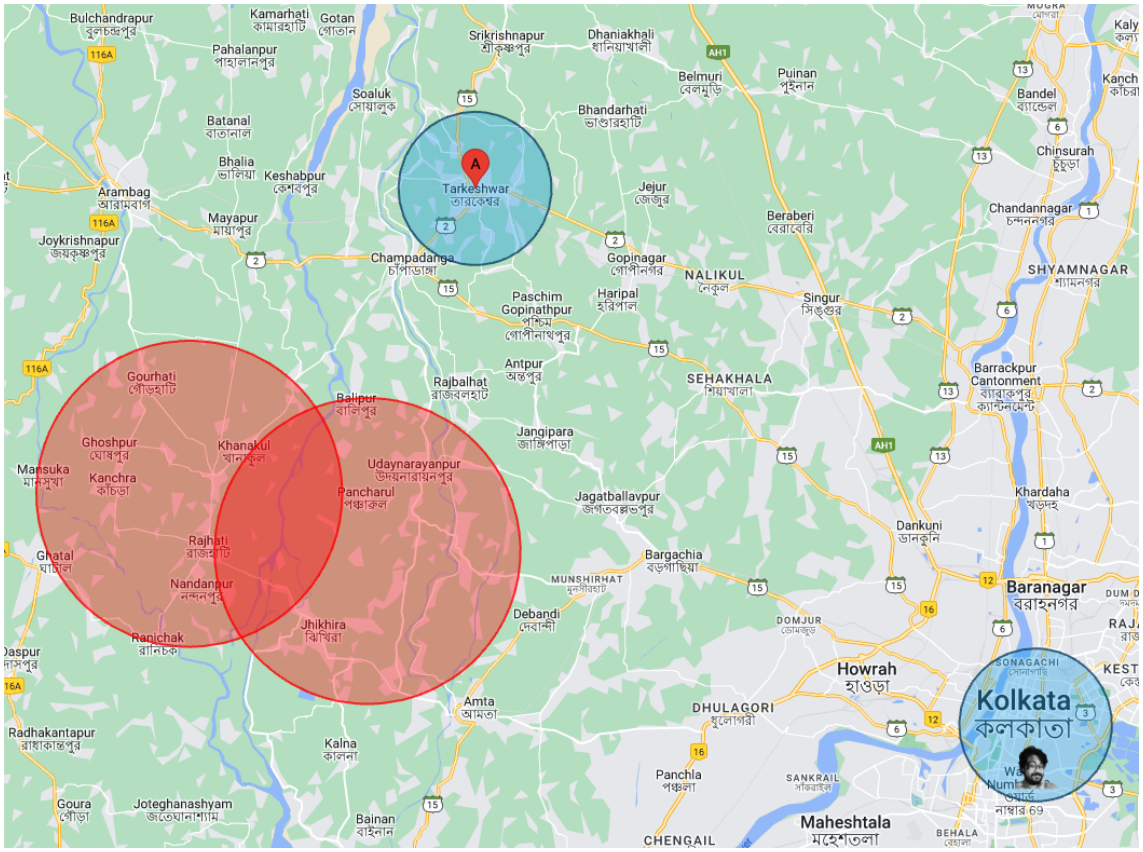

Figure 1.1: Privacy Filtration

1.2 Report Outline: The Privacy Network Approach

The report is organised as follows:

Chapter 2 presents the existing literature on privacy preserving schemes and techniques of location sharing for online mobile online social networks. Further, we summerise the features and technicalities of a similar web application based social networking site called *Loopt* that aimed to solve the similar issue.

Chapter 3 explores the architecture and the implementation details of Privacy Network. We briefly discuss the history and relevance of the MERN stack followed by a thorough scrutiny of the proposed architecture and the consequent coding implementation, CI/CD pipelines and deployment of the web application.

Chapter 4 and Chapter 5 provides a summary of the implementation outputs through a conclusion and discusses various potential future features and proposes advanced architectural to attain a distributed and scalable system.

Chapter 2

Literature Review

This chapter starts by briefly covering the relevant works in designing privacy preserving and efficient location sharing schemes for mobile online social networks. We discuss the case studies and vulnerabilities found through previous works. This is followed by the introduction of a product - Loopt, now discontinued, that tried to solve a very similar problem as that of ours.

2.1 Related Works

In the domain of mobile Online Social Networks (mOSNs), privacy and security have garnered significant research interest. Various privacy-preserving schemes have been proposed, each with distinct advantages and limitations. Early research primarily focused on information privacy, user anonymity, and location privacy. Techniques for maintaining location anonymity often involve encrypting the current location before transmission. K-anonymity methods obfuscate user locations, while dummy locations are used to mask real ones. Location encryption and pseudonym methods, such as mix zones and m-unobservability, are other prominent approaches.

K-anonymity for location privacy involves obfuscating a user's actual location, as demonstrated by researchers like Gruteser and Grunwald[3]. The use of dummy locations, where fake locations are sent along with the real one, has been explored by Kido et al[4]. Location encryption methods, such as those by Khoshgozaran et al.[5], offer another effective means of protecting location privacy. Pseudonym-based methods, mix zones, and m-unobservability, as explored by Beresford and Stajano[6], have also been developed.

Rahman et al.[7] proposed privacy context obfuscation based on various location parameters to achieve location obscurity. The concept of location sharing with privacy protection in online social networks was initially addressed by SmokeScreen, which facilitated location sharing between friends and strangers. This approach was later enhanced by Wei et al.[8] with MobiShare, which separated social and location information into Social Network Services (SNS) and Location-Based Services (LBS), respectively. However, MobiShare faced the issue of LBS potentially revealing social network topologies during queries.

Li et al.[9] advanced this concept with MobiShare+, introducing dummy queries and private set intersection to prevent LBS from accessing users' social information. BMobiShare

further improved transmission efficiency by using a Bloom Filter instead of the private set intersection method, though it incurred high computation costs.

To counter insider attacks, Li et al.[10] proposed a multi-server location-sharing system in 2015, enhancing security at the cost of increased resource demands and inefficiency. These systems, relying on third-party location servers, are susceptible to collusion between LBS and SNS, leading to potential social information exposure and high transmission and storage costs.

Recently, Xiao et al.[11] introduced CenLocShare, which combines SNS and LBS into a single server, reducing communication and storage costs while enhancing user privacy protection. This amalgamation addresses the inefficiencies and security concerns of previous multi-server approaches, offering a more streamlined and secure solution for privacy-preserving location sharing in mOSNs.

Existing solutions however, typically follow two main approaches: using separate servers for location-based and social network information, which incurs high storage and communication overhead, or integrating both into a single server, which may lead to server bottlenecks and vulnerabilities to security attacks. Bhattacharya et al.[12]. proposed a novel privacy-preserving, secure, and efficient location-sharing scheme for mOSNs that addresses these issues. The proposed scheme ensures efficient and flexible location updates, sharing, and queries among social friends and strangers. Security validation is performed through a random oracle-based formal security proof, Burrows-Abadi-Needham (BAN) logic-based authentication proof, and informal security analysis. Additionally, the scheme's security is verified using ProVerif 1.93. Experimental implementation and evaluation demonstrate the scheme's efficiency and practicality.

2.2 Loopt

Loopt, Inc. was an American company headquartered in Mountain View, California. It provided a service for smartphone users to selectively share their location with others. They created an interoperable social-mapping service that allowed individuals to use their location to discover the real world around them – enabling them to find and enjoy the people, places, and events that mean the most right here just by using their mobile phones[13].

Founded in 2005, Loopt received initial funding from Y Combinator[14]. It secured Series A and B financing led by Sequoia Capital and New Enterprise Associates. Thereafter, Loopt partnered with Boost Mobile, Sprint, and Verizon to expand its reach. The iPhone app, Loopt Mix[15], enabled users to find and meet new people nearby. In 2012, Loopt was acquired by Green Dot Corporation for $43.4 million.

Figure 2.1: Loopt Inc

Features:

- **Location Sharing**: Loopt allowed users to share their real-time location with friends and family.
- **Privacy Controls**: Users could customize who could see their location, ensuring privacy.
- **Cross-Platform Support**: The service worked across major mobile operating systems.
- **Integration with Social Networks**: Users could link Loopt with Facebook and Twitter.
- **Proximity-Based Messaging**: Users could send messages and photos based on their location1

Figure 2.2: Loopt: The Geo Social Network

It is noteworthy to mention that Loopt was among the pioneers of location-based social mapping services. Its innovative approach influenced subsequent location-sharing apps and services.

Chapter 3

Architecture

In this chapter, we go thorugh a brief overview and history of the MERN stack and then have a detailed discussion about the architecture of Privacy Network - the frontend, backend, deployment and scaling.

3.1 MERN Stack with TypeScript

MERN stands for *MongoDB*, *Express.js*, *React.js* and *Node.js*. With MongoDB as the Database, Express.js acts a web server framework (integrated with Node.js), React.js is the web client library, and Node.js is the server-side JavaScript runtime. It helps developers to develop Web apps based solely on full-stack JavaScript. Due to the same JavaScript platform in both the frontend and the backend, unofirmity in the codebase is maintained. Naturally, the stack is supported by a vast number of open-source packages and a dedicated community for programmers to increase scalability and maintain software products.

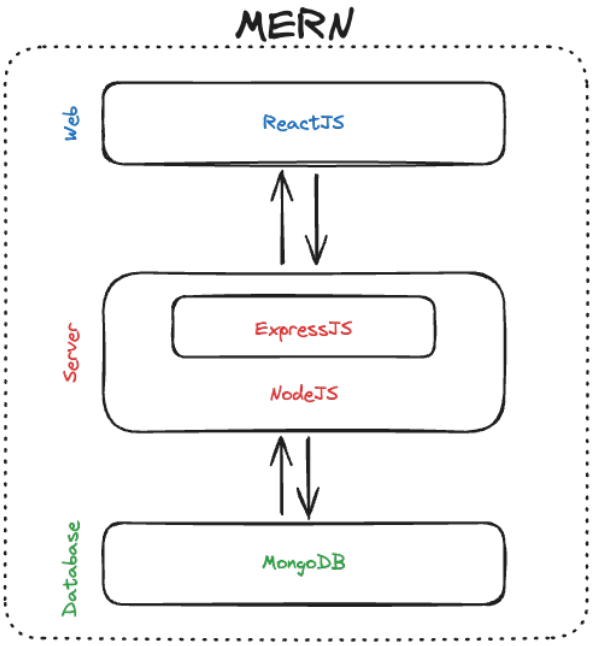

Figure 3.1: MERN stack architecture

3.1.1 Node

We are all familiar with the Google Chrome web browser. A typical web browser like Google Chrome runs HTML, CSS and JavaScript to render a website. The JavaScript code's execution actually happens by the virtue of an engine located inside the chrome browser called ***V8***. Written in C++, V8 has proven to be an extremely powerful engine that can interpret and execute JavaScript very quickly.

Recognizing the robust capabilities of the V8 engine, the founders envisioned decoupling V8 from Chrome to develop a platform capable of executing JavaScript code on the server side. This vision materialized in 2009 when **Ryan Dahl** created ***Node.js***.

Node.js serves a role analogous to that of the *Java Virtual Machine* (JVM), functioning as a platform and runtime environment for applications. While the JVM executes *bytecode*, Node.js directly interprets and executes JavaScript code. In the JVM ecosystem, developers typically write source code in high-level languages such as Java, Scala, Groovy, or Kotlin, which is then compiled into bytecode. Conversely, Node.js natively understands and executes JavaScript, enabling developers to write application code directly in JavaScript. Additionally, languages that compile into JavaScript, such as TypeScript, can be utilized within the Node.js environment.

Non-Blocking programming in Node.js - The Node.js platform is versatile, perhaps not because of the V8 engine or the ability to support the JavaScript language, but in the Non-Blocking style of programming. Operations related to Non-Blocking in Node.js are mostly related to IO, such as reading and writing data to disk or calling Web APIs, for example. Furthermore, the use of Non-Blocking operations makes applications written on a Node.js platform capable of using computational resources (CPU) most efficiently.

In the creation of the Non-Blocking mechanism, Node.js applications operate according to the Event Loop design pattern. The illustration below explains this design in more detail:

- **Event Queue**: Considered as a repository to store Events (events). Event here can be understood as a particular process in the program. Each event contains information to classify what that event is and the accompanying data. Because it is a queue structure, the Event Queue will operate on the principle of First In First Out (FIFO). This ensures that the order in which the Events are retrieved for processing is the same as the order they are put into the Event Queue.
- **Main Thread**: Saying this is the Main Thread because it will be the source and the end of the program. This main Thread is responsible for processing computation when receiving Events from the Event Queue. This is also the only Thread of the Node application that the programmer has control over. That is also the reason why it is still said that a Node application is single-threaded. Because of that, programmers will not have a headache about concurrency problems between threads like in some other platforms like Java.

- **Node API**: The unit responsible for handling IO operations. IO operations will be handled here by the multi-threaded mechanism. Moreover, each IO operation after completion will return the result as an event, and this event will be put in the Event Queue.

With the above three components, the way a Node application behaves like this:

- The Main Thread will run the computation processing statements declared in the source code. Wherever IO manipulation is involved, the Main Thread will make a Non-Blocking call to the Node API and then continue to execute other commands without waiting for the other IO operation to complete.
- When receiving a request from Main Thread, Node API will directly handle IO manipulation by different Threads. Once the IO operation is completed, the Node API will wrap the result as an Event and place it into the Event Queue.
- Events in the Event Queue will be processed by the Main Thread one after another. In the Main Thread, the code used to handle Events is usually declared as callback.

The above process is repetitive, creating a cycle of events in the application. The programming will be imposed in a way that is geared towards event handling instead of the traditional sequential way of thinking.

Node Package Manager - *npm, yarn, pnpm* are some of the dependency managers used and comes together to support each development of Node.js. Most libraries are available on NPM, so it is relatively easy to include them just by running the npm install command line to download them.

NPM is being maximized and operating based on two prominent roles as follows:

- NPM is a Software Registry and is being used very extensively to publish open-source Node.js projects. One can understand that: it is an online platform that allows all users to publish and share several other tools 11 written in JavaScript. It has crossed over 1.3 million code packages as of April 2020.
- NPM is one of the command-line tools that can help interact with other online platforms such as servers or web browsers. It is also a utility that can assist with package installation, package uninstallation, version management, and server-driven management. Also, NPM is used to manage the dependencies needed to run the projects.

3.1.2 Express

Express.js is released under an open-source license, has a large community of support, and is used for commercial applications. Hence, developers can completely rest assured to use this framework for their projects, from small projects to large projects.

Express.js has a ton of support packages and provides additional features for developers to a better program. Nevertheless, it does not slow down the speed of Node.js. The famous Node.js platform today is using Express.js as a core function.

According to the GitHub repository, Express.js was founded by TJ Holowaychuk and was first released on May 22, 2010, with version 0.12. In June 2014, project management rights were acquired by StrongLoop. IBM acquired StrongLoop in September 2015. In January 2016, Express.js was managed by the Node Js Foundation.

Middlewares - They are computer software that connects software components or applications together. Software of this type consists of services that allow interaction between processes running on one or more different machines. Middleware technology has been developed to provide interoperability, catering to commonly used distributed architectures to support and simplify complex distributed applications.

When working with Express.js, developers use various middleware functions with different functionalities:

- Functions receive the request, response of 1 cycle of HTTP request/response.
- Functions can correct requests and responses before sending them to the next() middleware function.
- Functions can update or terminate the request/response cycle.

The middleware function in Express.js often takes three parameters: the request (req) object, the response (res) object, and the next() function. Figure below illustrates a custom middleware used in the project.

3.1.3 MongoDB

MongoDB was first created by MongoDB Inc. in October 2007. It was part of the PaaS (Platform as a Service) product like Windows Azure and Google App Engine. It was later made open source in 2009.

MongoDB is a document-oriented database and a NoSQL database. Therefore, MongoDB will often avoid the relational database's table-based structure to adapt to all documents such as JSON, available in a very flexible schema called BSON.

Advantages of MongoDB:

- Since MongoDB uses data in the JSON form, each collection has different sizes and documents. Nevertheless, they are very flexible when it comes to archiving.
- MongoDB data is usually not bound to each other; it does not support join query like in RDBMS, so when users insert, delete or update, it will not spend too much time to check if it satisfies the constraints like in RDBMS or not.

- MongoDB is easy to scale. In MongoDB, the concept "cluster" refers to clusters of nodes containing data to communicate with each other. Adding a new node to the cluster helps users expand the system quickly.
- The unique identifier *_id* will always be indexed automatically, so the speed of querying information will consistently achieve the highest performance.
- Data query will be cached to RAM with little access to the hard drive so the read and write speed is faster.

3.1.4 React

React.js is a JavaScript library built by Facebook engineers, being used by many famous companies to develop their products such as Yahoo, Airbnb, Facebook, Instagram. It is better suited for large, scalable projects rather than small projects.

Features of React.js:

- The React.js mindset is to build reusable components that make it easy to break down problems and test. It helps us quickly manage and extend the system; if it is Angular, it requires optimal structure and coding style.
- React.js always keeps components stateless (as much as possible), making it easy to manage because it is like a static HTML page. It takes inputs from the outside and only displays it against those inputs, explaining why it is reuse and easy for testing.

Strengths of React.js: React.js is a performance-minded view rendering framework. Many of the heavyweights on the MVVM (Model-View-ViewModel) framework take a long time to display large amounts of data, such as lists. However, with React.js, that is no longer an issue, as it just shows what changes.

One of React.js' more strengths is the virtual DOM (Document Object Model) - which is hidden inside every view and is the reason why React.js achieves good performance. When a view requests a call, everything is included in a virtual copy of the DOM. React.js compares the virtual DOM and the real DOM after the call completes and makes the changes indicated in the above comparison.

For example, if we are looking at a list of 20 products displayed by React.js and change the 2nd product, only that product is re-displayed, and the remaining 19 products remain the same (no need to display, reload or reload the page). React.js used the so-called virtual DOM to increase performance by outputting a virtual view, then checking the difference between the virtual rendering and what is in the DOM and creating a patch.

3.1.5 TypeScript

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.2 Privacy Frontend

3.2.1 A component based React + Material UI

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.2.2 Vite the build tool

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.2.3 React Custom Hooks

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.3 Privacy Backend and Databases

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.3.1 Database Schema, ORMs and ERDs

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.3.2 MongoDB and PostgreSQL

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.3.3 PostGIS for geolocation SQL queries

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.3.4 WebSocket Server for real-time location sharing

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in

voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.3.5 OpenAPI Specs and Swagger documentation

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.4 Cloud Hosting and Deployment

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.4.1 Vercel the cloud service provider for the frontend

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

3.4.2 Render the cloud service provider for the backend

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Chapter 4

Conclusion

This thesis focuses on the implementation of a security-conscious Geosocial Networking web application, named *Privacy Network*. To address privacy concerns, *Privacy Network* prioritizes user control over location data. Users can define their visibility settings with granular control, ensuring a secure and customizable social media experience.

The front-end technologies used to develop the web application are:

- **User Interface**: React, a component-based UI library, is used for a smooth and interactive user experience
- **Styling**: Material UI, TailwindCSS, and raw CSS are
- **Build Tool**: Vite, a lightning-fast development server and build tool, is used an overall smoother development experience.

The backend utilizes a distributed architecture for scalability, featuring:

- **MongoDB server**: Stores user authentication and profile data
- **Postgres server**: Handles user location attributes with PostGIS enabling location-based queries.
- **WebSocket server**: Facilitates real-time communication through WebSockets, allowing for immediate location updates and interactions trough bi-directional fullduplex communication channel.
- **OpenAPI Specs**: A comprehensive documentation for all backend API endpoints, strictly adhereing to the standardized OpenAPI specifications is accessible in the well-known Swagger format.

Git is used for version controlling and GitHub hosts the git repository of our application codebase. Platforms like Vercel and services like Render are utilized to deploy the web application's frontend and backends respectively while GitHub Actions CI/CD pipelines ensuring smooth deployments.

By prioritizing user privacy and utilizing scalable architecture, *Privacy Network* lays the groundwork for a secure and dependable geosocial networking platform.

Privacy Network can be accessed at ***https://privacynetwork.sanam.live***

Chapter 5

Future Work

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

5.1 Scaling WebSocket Servers with Redis Pub-Sub and HAProxy

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

5.2 Change Data Capture for synchronization of SQL-NoSQL Databases

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Bibliography

[1] Wikipedia, "Geolocation networking," 2008-.

[2] BBC, "Meta settles cambridge analytica scandal case for $725m," 2022.

[3] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proc. 1st Int. Conf. Mobile Syst., Appl. Services (MobiSys)*, 2003.

[4] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *Proc. 21st Int. Conf. Data Eng. Workshops (ICDEW)*, 2005.

[5] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Springer*, 2007.

[6] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervas. Comput*, 2003.

[7] M. Rahman, M. Mambo, A. Inomata, and E. Okamoto, "An anonymous on-demand position-based routing in mobile ad hoc networks," in *Proc. Int. Symp. Appl. Internet (SAINT)*, 2006.

[8] W. Wei, F. Xu, and Q. Li, "Mobishare: Flexible privacy-preserving location sharing in mobile online social networks," in *Proc. IEEE INFOCOM*, 2012.

[9] X. Chen, Z. Liu, and C. Jia, "Mobishare+: Security improved system for location sharing in mobile online social networks," *J. Internet Serv. Inf. Secur.*, 2014.

[10] J. Li, H. Yan, Z. Liu, X. Chen, X. Huang, and D. S. Wong, "Location-sharing systems with enhanced privacy in mobile online social networks," *IEEE Syst. J.*, 2017.

[11] X. Xiao, C. Chen, A. K. Sangaiah, G. Hu, R. Ye, and Y. Jiang, "Cenlocshare: A centralized privacy-preserving location sharing system for mobile online social networks," *Future Gener. Comput. Syst.*, 2018.

[12] M. Bhattacharya, S. Roy, K. Mistry, H. P. H. Shum, and S. Chattopadhyay, "A privacy-preserving efficient location-sharing scheme for mobile online social network applications," 2020.

[13] Y. Combinator, "Loopt," 2005.

[14] B. B. Nerd, "Y combinator's first batch (yc so5)," 2024.

[15] LoopTMix, "Loopt: The geo social network."