CLOUD BACKEND INTERNSHIP AT EMOTORAD

An Industrial Internship Report

submitted by

ABHINAV DINESH SRIVATSA

21BDS0340

in partial fulfilment for the award of the degree of

BACHELORS OF TECHNOLOGY

in

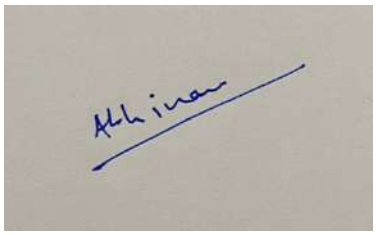
COMPUTER SCIENCE WITH SPECIALISATION IN DATA SCIENCE

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

JULY 2023

DECLARATION BY THE CANDIDATE

I hereby declare that the Industrial Internship report entitled “CLOUD BACKEND INTERNSHIP AT EMOTORAD” submitted by me to Vellore Institute of Technology, Vellore in partial fulfilment of the requirement for the award of the degree of Bachelors of Technology in Computer Science with Specialisation in Data Science is a record of bonafide industrial training undertaken by me under the supervision of Gaurav Gupta, Emotorad. I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.



Name: Abhinav Dinesh Srivatsa
Reg. Number: 21BDS0340

Date: 23-02-2024

Name: Mr. Abhinav Srivatsa

To whomsoever it may concern

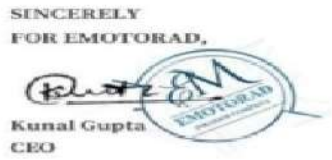
This is to certify that **Mr. Abhinav Srivatsa** has been associated with EMotorad as a **Cloud Backend Intern** for the period from **07-08-2023** to **15-12-2023**.

During his tenure he has carried out his duties responsibly.

We highly appreciate all the contributions he has made in favor of the organization, and we wish him good luck in his future endeavors.

Yours Sincerely,

For INKODOP TECHNOLOGIES PVT LTD.



CEO and Co-Founder

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to CPTO Gaurav Gupta for his invaluable guidance and unwavering support throughout my internship journey. My heartfelt thanks extend to the exceptional members of the product team—Swaraj Chouriwar, Devidutta Nayak, and Badrinath Aryan—whose collaborative spirit and expertise enriched my learning experience. I am grateful for the camaraderie and teamwork exhibited by my colleague, Manoj Gowda, which made the work environment both productive and enjoyable. Additionally, I extend my thanks to the HR professionals, Rishika Sinha, and Rosemary Amolik, for their tireless efforts in facilitating a smooth and welcoming work atmosphere. Together, these individuals have played a pivotal role in shaping my internship, and I am truly grateful for their contributions to my professional development.

Place : Vellore

Abhinav Dinesh Srivatsa

Date : 23rd February 2024

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	vii
1.	**Synopsis of the Report**	1
2.	**About the Organisation: EMotorad**	3
	2.1 Introduction	3
	2.2 Technology	4
	2.3 Future Ventures	5
3.	**Skillset Before Training**	6
	3.1 Programming	6
	3.2 Data Structure and Algorithms	6
	3.3 Computer Hardware and Architecture	7
	3.4 Computer Theory	8
4.	**Knowledge Acquired through Training**	10
	4.1 Go Programming	10
	4.2 Docker	11
	4.3 Kubernetes	12
	4.4 Amazon Web Services	13
	4.5 Apache Flink	15
	4.6 Firebase	16
5.	**Application of Knowledge Acquired**	18
	5.1 Microservices	18
	5.2 Apache Flink Jobs/Streaming Applications	19
6.	**Skillset after Training**	21

LIST OF FIGURES

Figure 1: Detail about Docker containerisation.

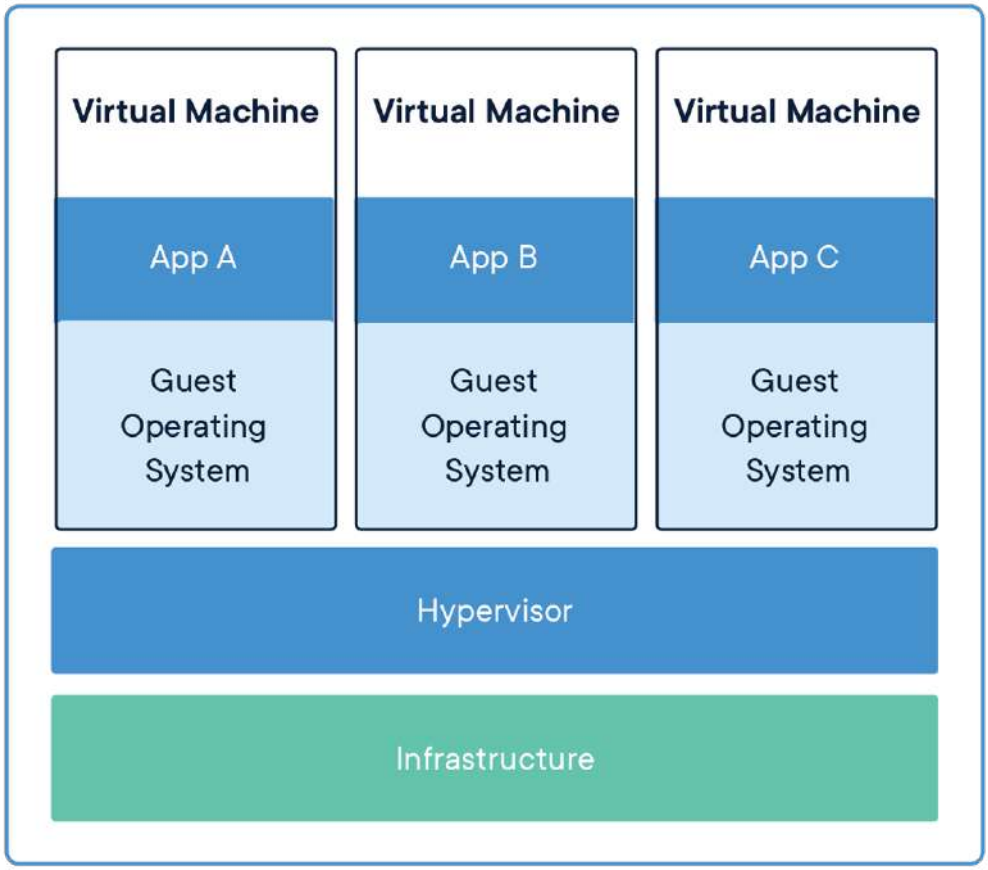


Figure 2: Detail of how Flink processing works.

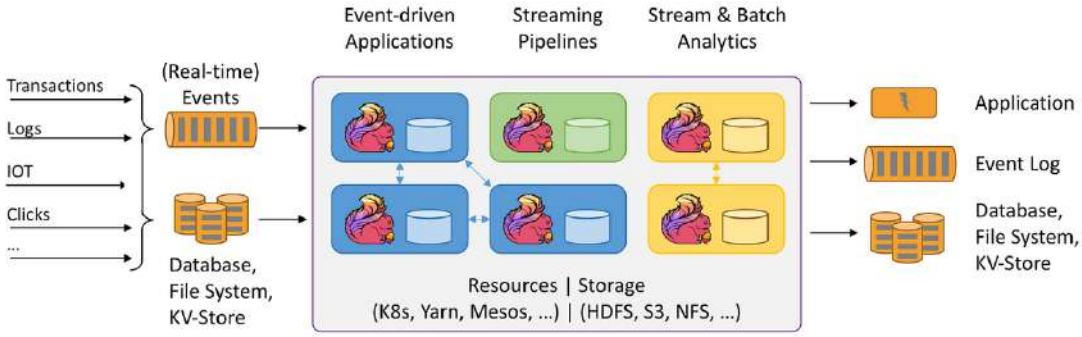


Figure 3: Steps to send push notifications via Firebase.

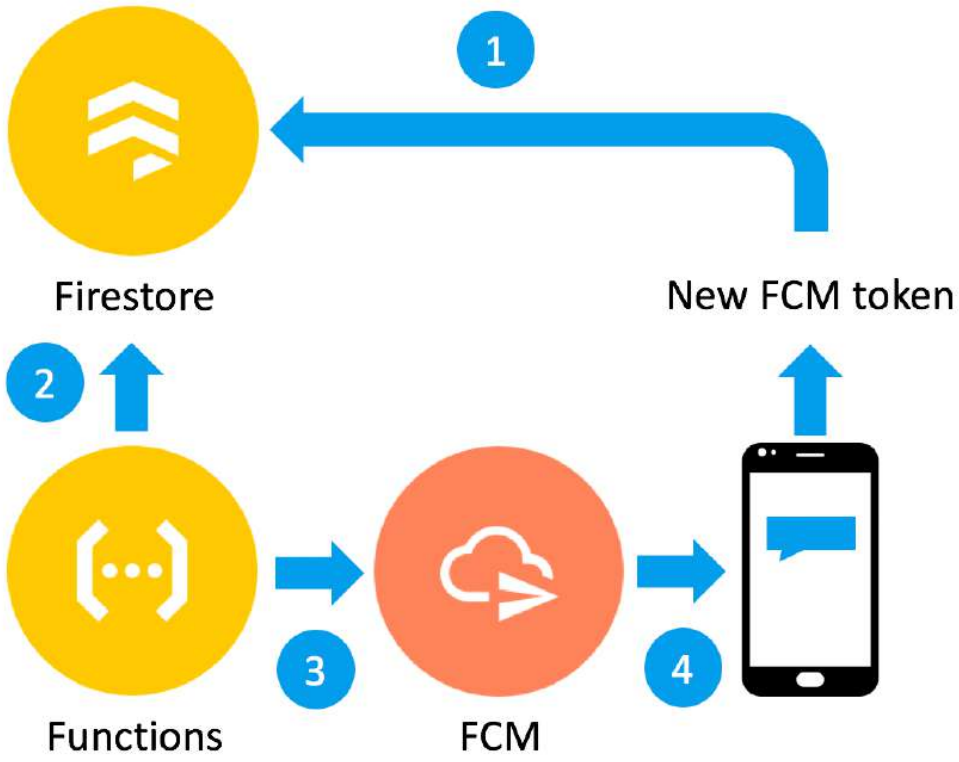
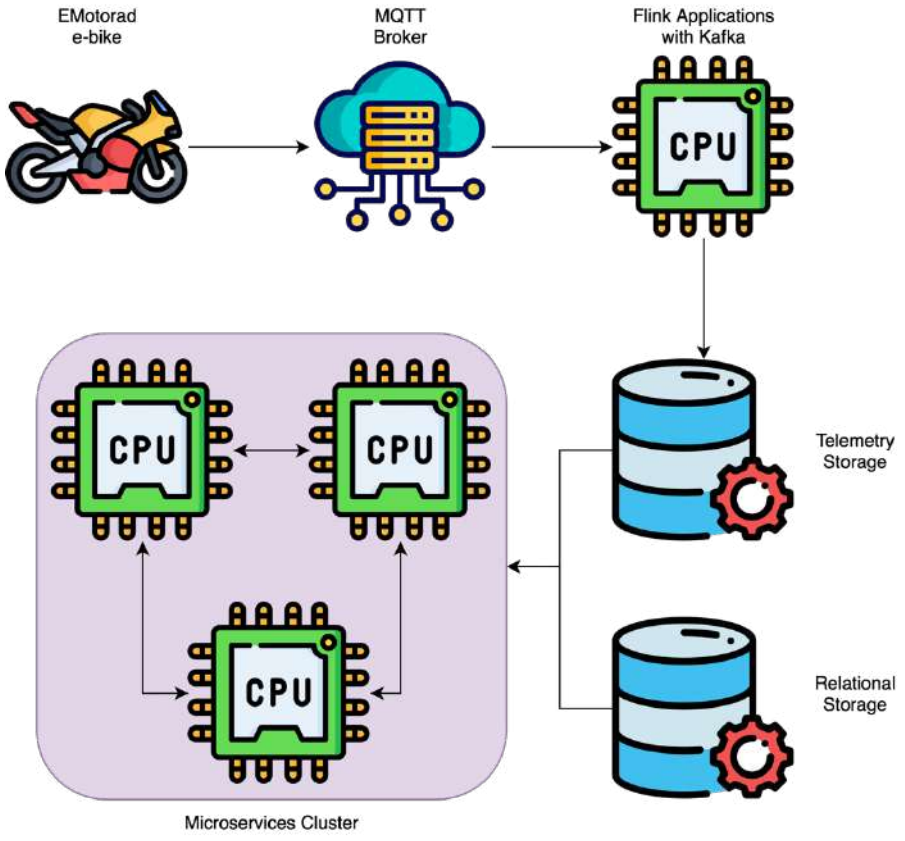


Figure 4: Implemented architecture diagram.



Synopsis of the Report

My internship at EMotorad expanded my skills and knowledge, bridging the gap between my academic background and real-world applications. College courses in programming languages, data structures, and algorithms provided a solid foundation, setting the stage for practical challenges at EMotorad.

College taught me languages like Java, Python, C, C++, and JavaScript, along with algorithmic problem-solving. This background prepared me well for addressing complex tasks in the electric vehicle technology sector. Additionally, I gained insights into computer hardware, architecture, operating systems, network protocols, and databases, offering a holistic view of the interplay between software and hardware.

At EMotorad, I delved into the Go programming language, focusing on its concurrency features. This knowledge proved crucial for handling concurrent tasks efficiently in modern application development.

Containerisation became a key theme during my internship, with Docker transforming how applications are deployed. Understanding container orchestration with Docker Compose and Kubernetes reflected a practical understanding of DevOps practices. Proficiency in Amazon Web Services (AWS) showcased my ability to deploy applications on a cloud infrastructure, from managing databases with Amazon RDS to orchestrating Kubernetes clusters on AWS EKS. Apache Flink played a role in real-time data processing, with applications for MQTT bridging, Protobuf deserialisation, and efficient storage in AWS S3, highlighting adaptability in managing diverse data tasks. Firebase services enhanced user engagement through Cloud Functions, Cloud Messaging, and Cloud Firestore, contributing to a dynamic user experience.

The development of microservices for the EMotorad Companion App was a key aspect of my internship. From secure authentication mechanisms to real-time communication through MQTT and efficient data retrieval using Redis and PostgreSQL, this microservices architecture showcased my ability to design and implement distributed systems.

In summary, my five-month internship journey integrated theoretical knowledge with practical application. EMotorad provided a platform to adapt, learn, and innovate in a dynamic technology domain, shaping me as a versatile and adaptive technologist ready for the evolving tech landscape.

About the Organisation: EMotorad

Introduction:

EMotorad's brand vision encapsulates a commitment to unparalleled excellence in the electric vehicle sector, transcending conventional notions of transportation. Rooted in the aspiration to be recognized as the "Best in the World," EMotorad goes beyond the production of electric cycles, seeking to redefine the entire riding experience. The vision extends to incorporating innovation that not only simplifies but enriches the user journey, positioning EMotorad as a frontrunner in the industry with a relentless pursuit of excellence and sustainability.

In the context of the emerging electric region in India, EMotorad stands at the forefront of this transformative wave. As the nation shifts towards sustainable and eco-friendly mobility solutions, the electric vehicle market in India is experiencing a surge in demand. With a global presence spanning India, EMotorad strategically positions itself to cater to this burgeoning market. India's commitment to reducing carbon emissions and promoting green transportation aligns seamlessly with EMotorad's mission to create a positive impact on both individuals and the environment.

A standout feature of EMotorad's contribution to the electric vehicle landscape in India is its dedication to affordability without compromising on style and innovation. In a region where cost considerations play a pivotal role in consumer decisions, EMotorad addresses this factor head-on. By focusing on producing electric cycles that merge advanced technology with affordability, the brand ensures that a wider segment of the population can embrace the benefits of electric transportation. This commitment is pivotal in making electric cycles more accessible and appealing to a diverse demographic, driving the adoption of sustainable mobility solutions.

Moreover, EMotorad's emphasis on crafting catchy and visually appealing transportation solutions further solidifies its position in the market. Recognizing the importance of aesthetics and design in consumer choices, EMotorad ensures that its electric cycles not only boast advanced technology but also stand out with eye-catching designs. This commitment to creating visually appealing transportation options not only enhances the overall consumer experience but also contributes to reshaping the

narrative around electric vehicles, making them not just a pragmatic choice but a stylish and trendy one.

In essence, EMotorad's vision aligns with the changing landscape of the emerging electric region in India. The brand's commitment to affordability and catchy transportation not only addresses the current market needs but also plays a pivotal role in shaping the future of sustainable and stylish mobility in the country. EMotorad's vision, deeply rooted in excellence and sustainability, positions the brand as a key player in India's electric vehicle revolution.

Technology:

EMotorad's technological prowess is at the core of its electric cycle innovation, seamlessly integrating cutting-edge technologies to elevate user experience and redefine industry standards. The backend data pipeline, a critical component, is constructed with Apache Flink, Kafka, and Cassandra. Apache Flink ensures real-time data processing, enabling EMotorad to provide users with dynamic and responsive features. Kafka, as a distributed event streaming platform, facilitates efficient communication between various components, ensuring a smooth and synchronized operation. Cassandra, a highly scalable NoSQL database, contributes to robust and fault-tolerant storage, enhancing the overall reliability of EMotorad's backend infrastructure.

In the realm of data storage and user management, EMotorad leverages Redis and PostgreSQL. Redis, a high-performance in-memory data store, is employed for caching and quick retrieval of frequently accessed data, optimizing the overall system performance. PostgreSQL, a powerful relational database, handles structured data storage, ensuring data integrity and providing a solid foundation for user-related information.

For analytics, EMotorad utilizes AWS S3 and Trino, previously known as Presto. This combination allows EMotorad to store vast amounts of data in a scalable and cost-effective manner on AWS S3. Trino, with its distributed query engine, empowers EMotorad to analyse this data efficiently, extracting valuable insights and supporting data-driven decision-making processes.

The adoption of a Microservices architecture using Golang further strengthens EMotorad's technological backbone. Golang, renowned for its efficiency and performance, facilitates the development of modular and scalable microservices. This approach enhances the overall scalability, responsiveness, and maintainability of EMotorad's electric bikes, aligning with the brand's commitment to delivering a state-of-the-art riding experience.

Authentication of bikes and packet ingestion is handled through MQTT, with EMQX brokers ensuring secure and efficient communication. This not only ensures the authenticity and security of the electric bikes but also enables seamless packet ingestion, contributing to the real-time data processing capabilities of the backend.
EMotorad's technological infrastructure extends to the cloud, with AWS Cloud serving as the foundation for hosting and managing its Kubernetes environment. The use of AWS Cloud provides scalability, reliability, and ease of management, allowing EMotorad to focus on continuous innovation and improvement while leveraging the benefits of cloud computing.

On the frontend, EMotorad adopts Flutter, a cross-platform framework, for app development on both iOS and Android platforms. This choice emphasizes the company's commitment to delivering a consistent and user-friendly interface, ensuring a seamless experience for riders across various devices.

Future Ventures:

Looking ahead, EMotorad is diversifying its product portfolio by venturing into the domain of electric motorcycles. This strategic expansion reflects the company's commitment to continual evolution and its ambition to offer a comprehensive range of sustainable mobility solutions.

Skillset Before Training

Programming:

Starting with Java, I've got a good grip on the basics of object-oriented programming. I can comfortably write code for simple algorithms, work with classes and objects, and manage data structures. My projects in Java usually involve solving straightforward problems and implementing basic applications using OOPS concepts.

In Python, I've covered the basics, including syntax, data types, and control structures. I can create simple scripts for automation tasks, and I'm familiar with Python's readability and ease of use. While I'm not an expert, I can develop basic programs and have used it for tasks like data analysis and simple web applications.

In C, my focus has been on grasping fundamental concepts like variables, loops, and functions. I can write basic C programs, understanding the essentials of syntax and logic. My projects in C usually involve tasks such as implementing algorithms and basic system-level programming. In C++, I've delved into the basics of object-oriented programming. I can create simple classes, implement inheritance, and use basic templates. My projects in C++ typically involve building console applications, and I have a decent understanding of its syntax and core concepts.

JavaScript, for me, revolves around the basics of web development. I understand how to use it to make web pages interactive. I can create simple functions, manipulate the DOM, and handle user inputs. My exposure to JavaScript includes basic projects involving client-side scripting for web applications.

Data Structures and Algorithms:

In the realm of sorting techniques, I've delved into various algorithms, including Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, and Radix Sort. Understanding their time and space complexities has equipped me to choose the most suitable sorting method for diverse scenarios.

Lists, being fundamental data structures, have been a focus of study. From arrays to linked lists, including both singly and doubly linked lists, I've grasped the advantages

and limitations of each. Circular linked lists have also been explored, enhancing my understanding of linked structures.

Maps, particularly hash maps, have been introduced with an emphasis on hash functions and efficient data retrieval. Additionally, I've explored tree-based data structures such as Binary Search Trees (BST), AVL Trees, and Red-Black Trees, comprehending their applications and advantages.
In the domain of stacks and queues, I've learned their implementations and use cases. Stacks have been applied to problems like expression evaluation and backtracking, while queues have found utility in algorithms like breadth-first search.

The study of trees has covered binary trees and their applications, along with an exploration of balanced trees like AVL and Red-Black Trees for optimized search operations.

I have studied algorithms like branch and bound, which I applied in solving optimization problems. Techniques for efficiently pruning search spaces. Greedy algorithms for decision-making strategies based on locally optimal choices. Applications in Minimum Spanning Trees and Huffman Coding. Also divide and conquer a paradigm of breaking down problems into subproblems. Applications in algorithms like Merge Sort and Quick Sort.

Computer Hardware and Architecture:
My exploration of computer architecture and hardware has provided me with a comprehensive understanding of foundational concepts and advanced technologies in this domain. I've delved into both Harvard and von Neumann architectures, discerning their distinctive characteristics and applications.

Memory systems have been a focal point, with an in-depth study of Random Access Memory (RAM) and its vital role in storing and retrieving data. RAID (Redundant Array of Independent Disks) configurations have been explored, providing insights into data redundancy and performance optimization through parallel disk access.

In the realm of microcontrollers, I've gained proficiency in the 8051-microcontroller architecture. This versatile microcontroller is widely used in embedded systems, and my studies have covered its internal structure, instruction set, and applications in various electronic devices.

Furthermore, my coursework has extended to ARM processors, where I've delved into the intricacies of Advanced RISC Machines. Understanding ARM architecture has equipped me with the knowledge of efficient instruction sets and their applications in diverse computing environments.

The curriculum has provided a solid foundation in computer organization and assembly language programming, emphasizing the interaction between hardware and software. Practical applications of these concepts have been explored, enhancing my ability to design and optimize software for specific hardware architectures.

Computer Theory:

The curriculum covered fundamental concepts such as finite automata (FA), non-deterministic finite automata (NFA), epsilon-NFA, and Turing machines. These topics provided a solid foundation for understanding pattern recognition, with finite automata serving as essential models for recognising patterns in strings. The inclusion of non-deterministic and epsilon-NFA expanded my insights into more flexible and expressive pattern recognition. Additionally, studying Turing machines offered a deep dive into the theoretical underpinnings of computation, providing a comprehensive understanding of algorithmic limits and capabilities.

Compiler design was another significant aspect of my academic journey. Topics such as SLR (Simple LR), CLR (Canonical LR), and lookahead compilers like LR(0) were explored. This coursework proved crucial for comprehending the intricate process of translating high-level programming languages into machine code. The study of different parsing techniques, including SLR and CLR, enhanced my knowledge of compiler optimisation and efficiency. This understanding is essential for bridging the gap between high-level code and machine-executable instructions.

Moving on to database systems, the curriculum covered a spectrum of crucial topics. Relational databases, a cornerstone in data management, were thoroughly explored. The study of storage mechanisms, including B-trees and B+ trees, provided insights into efficient data retrieval and storage. The exploration of actors in the context of database systems indicated a deeper dive into distributed computing and parallel processing. This is particularly relevant in today's computing landscape, where scalable and performant databases are essential for handling vast amounts of data.

The curriculum covered slotted algorithms, offering insights into time-based scheduling mechanisms crucial for network efficiency. Understanding the OSI (Open Systems Interconnection) model and the IP model provided a conceptual framework for comprehending the complexities of network communication. I gained knowledge about various protocols and standards, including TCP/IP, UDP, HTTP, and DNS, which play pivotal roles in facilitating seamless communication and data exchange across networks. The comprehensive study of these elements equipped me with the skills needed to design, implement, and troubleshoot network architectures.

Shifting focus to operating systems, my coursework encompassed fundamental concepts crucial for understanding the core functionalities of computer systems. System calls, as integral communication interfaces between software and the operating system kernel, were explored. Delving into the kernel, the heart of the operating system, provided insights into how it manages resources and facilitates communication between hardware and software components. Additionally, I gained a deeper understanding of various operating system-related terms such as process management, memory management, file systems, and device drivers. Exploring operating system fundamentals, including multitasking, multiprocessing, and virtual memory, further enriched my knowledge in creating efficient and robust computing environments. This exposure laid the groundwork for comprehending the intricate interplay between hardware and software within the realm of operating systems.

Knowledge Acquired through Training

Go Programming:

This experience provided a hands-on exploration of Go's unique features and its application in building server-side applications. One notable aspect was my exposure to the HTTP framework "Chi," which proved to be a powerful tool for crafting robust and scalable web applications. Learning how to leverage Chi for routing, middleware, and handling HTTP requests enhanced my ability to design efficient and maintainable server-side code.

A distinctive feature of my Go learning experience was the introduction to channels and goroutines. Channels, a core concurrency primitive in Go, became instrumental in orchestrating communication between concurrent parts of the program. This concurrency model, coupled with goroutines, offered a lightweight and efficient mechanism for handling concurrent tasks, thereby contributing to the development of highly concurrent and performant applications.

Notably, Go served as my first foray into a statically typed server-sided HTTP language. This marked a significant shift in my understanding of programming languages, as the static typing of Go brings a level of safety and efficiency to the development process. The strict type checking at compile-time enhances code reliability and makes it easier to catch potential errors early in the development lifecycle.

The EMotorad internship not only provided me with a practical foundation in Go but also fostered a deeper appreciation for statically typed languages, concurrent programming, and efficient server-side development. These skills have proven invaluable in my ongoing journey as a developer, enhancing my ability to tackle complex challenges in the ever-evolving landscape of software engineering.

Docker:

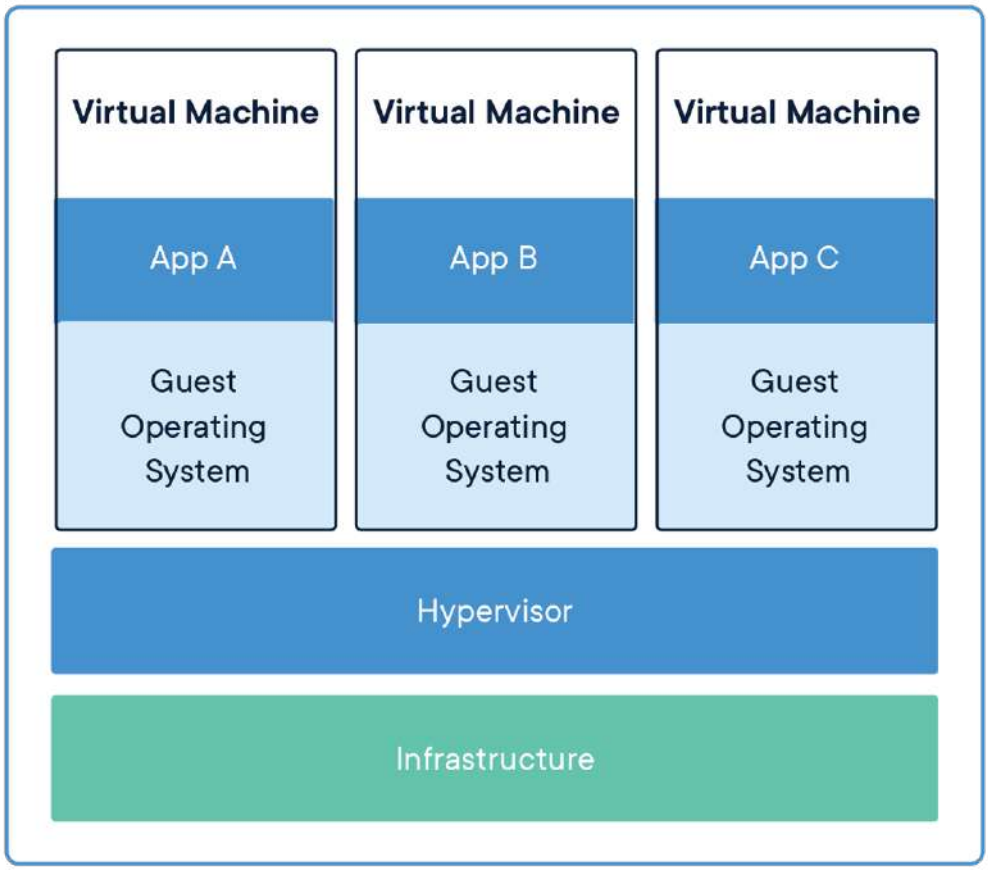


Figure 1: Detail about Docker containerisation.

EMotorad presented an exciting opportunity to delve into containerisation and understand its significance in modern software development. Docker emerged as a transformative tool that revolutionised how applications are deployed, ensuring consistency and reproducibility across various environments.

One of the key aspects of my Docker education was gaining proficiency in creating and managing containers using Docker technology. Understanding containerisation principles allowed me to encapsulate applications and their dependencies, ensuring seamless deployment across different environments, irrespective of the underlying infrastructure.

The exploration of Docker introduced me to the concept of container orchestration, where I learned about tools like Docker Compose and Kubernetes. These tools played a crucial role in simplifying the management of multi-container applications and

orchestrating their deployment at scale. Learning how to define and manage Docker Compose files empowered me to efficiently configure and deploy complex, multi-service applications.

A notable focus during my Docker experience was comprehending the importance of containerising applications for multiple architectures and platforms. Docker's ability to provide a consistent environment, regardless of the underlying system, opened doors to a new level of portability. This was particularly significant in the context of diverse architectures and platforms, enabling applications to run seamlessly on different machines, cloud services, and even across various operating systems.

The versatility of Docker in handling applications for multiple architectures and platforms proved invaluable. This capability not only streamlined the development and deployment process but also fostered a deeper understanding of the complexities involved in creating robust, cross-platform applications.

Kubernetes:

My venture into Kubernetes, during my experience at EMotorad, marked a pivotal chapter in my journey through modern DevOps practices. Kubernetes, with its robust orchestration capabilities, introduced me to a paradigm shift in managing containerised applications at scale.

A foundational aspect of my Kubernetes education involved working with various core components, such as deployments, statefulsets, pods, and services. Understanding deployments enabled me to efficiently manage the deployment lifecycle of applications, ensuring scalability, availability, and ease of updates. Statefulsets, with their emphasis on persistent storage and ordered deployment, added a layer of complexity that I learned to navigate adeptly.

Pods became a fundamental building block, encapsulating one or more containers and facilitating their co-located execution. The orchestration of these pods via services provided a seamless way to expose and connect components within a cluster, creating a cohesive application architecture.

Delving into Helm charts proved instrumental in simplifying the deployment and management of complex applications. Helm, as a package manager for Kubernetes, enhanced my ability to define, install, and upgrade even the most intricate application stacks with ease.

Exploring Kubernetes ingresses further deepened my understanding of managing external access to services within the cluster. I learned to configure and optimise ingress resources to efficiently route traffic and handle various aspects of external connectivity.

A notable dimension of my Kubernetes experience was deploying on AWS EKS, showcasing the platform's adaptability across cloud providers. While AWS EKS played a role in my practical application of Kubernetes, the emphasis lies on the broader understanding of Kubernetes itself, irrespective of the underlying infrastructure.

Amazon Web Services:
The utilisation of AWS services at EMotorad significantly influenced the architecture and functionality of the projects I worked on. Here's a deeper dive into each service and a brief reflection on how I acquired proficiency in these technologies over time.

Amazon RDS provided a seamless solution for managing and scaling PostgreSQL databases. I became adept at configuring and optimising database instances, leveraging automated backups, and implementing high availability setups. My journey with RDS involved a combination of hands-on experience, documentation exploration, and collaborative problem-solving with teammates.

Integrating Elastic Load Balancers into Kubernetes deployments required understanding traffic distribution, health checks, and scalability. My learning journey involved exploring AWS documentation, experimenting with load balancing configurations, and fine-tuning settings to ensure optimal performance.

Amazon ECR became a key component in the containerisation journey. I learned to manage Docker container images efficiently, creating repositories, and integrating

ECR seamlessly with Kubernetes deployments. The learning process for ECR involved a combination of official documentation, community forums, and practical hands-on experimentation.

Amazon EKS emerged as a go-to solution for deploying and managing Kubernetes clusters. I gradually developed expertise in deploying applications on EKS, configuring worker nodes, and optimising cluster performance. Learning EKS was a progressive process, involving experimentation, online tutorials, and practical application in real-world scenarios.

Implementing efficient caching with Amazon MemoryDB for Redis involved understanding Redis fundamentals, exploring serverless models, and optimising cache configurations. My proficiency in MemoryDB for Redis evolved through a combination of online courses, practical application, and collaborative problem-solving.

Employing Amazon Keyspaces for serverless Cassandra databases required understanding NoSQL principles, configuring databases, and optimising queries. My learning journey with Keyspaces involved studying Cassandra concepts, experimenting with data modelling, and troubleshooting performance issues.

Amazon S3 served as the cornerstone for versatile data storage. My understanding of S3 evolved through managing buckets, configuring access controls, and optimising data retrieval. My learning journey with S3 involved a mix of official documentation, interactive labs, and troubleshooting in real-world scenarios.

Managing event streaming with Amazon MSK involved configuring Kafka clusters, ensuring fault tolerance, and optimising performance. My exploration of MSK included studying Kafka fundamentals, consulting AWS documentation, and applying best practices in real-world implementations.

Apache Flink:

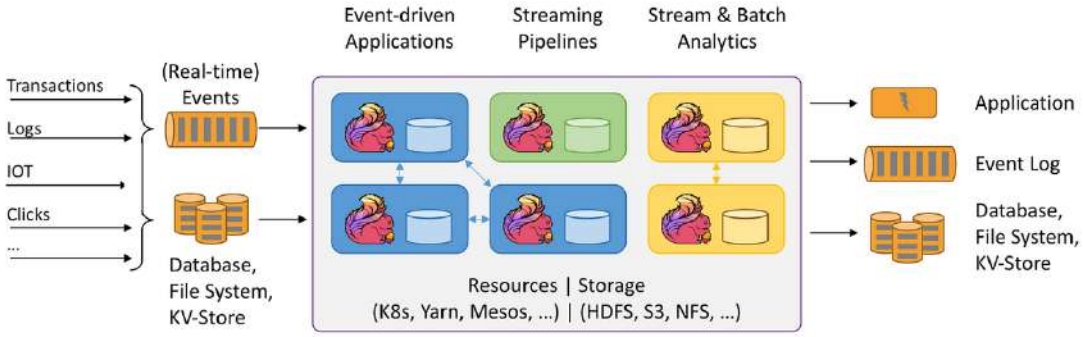


Figure 2: Detail of how Flink processing works.

Apache Flink marked a transformative phase where real-time data processing took centre stage. Leveraging the DataStream API and Kafka as my data source, I delved into creating resilient and scalable stream processing applications.

Apache Flink's DataStream API allowed for the development of robust applications with Kafka as the foundational data source. The seamless integration facilitated the consumption of streaming data, offering low-latency and fault-tolerant processing.

Exploring different sinks for data output broadened the horizon of application possibilities. Leveraging Cassandra, Redis, Kafka, and AWS S3 as sinks, I could efficiently channel processed data to diverse storage solutions. Each sink brought its unique advantages, from NoSQL database capabilities to scalable caching mechanisms and robust storage in the cloud.

Hosting Flink applications on Kubernetes brought a new layer of flexibility and scalability. The Flink-operator streamlined the deployment and management of Flink applications, optimising resource utilisation and ensuring smooth scaling based on demand.

My learning journey with Apache Flink and associated technologies was a gradual process. Initially, I familiarised myself with the DataStream API, understanding its intricacies and the integration points with Kafka. Hands-on projects involving different sinks allowed me to grasp the nuances of each technology, from setting up connections to optimising data flows.

I learned to host Flink applications on Kubernetes using the Flink-operator, which needed a deeper exploration of container orchestration and Kubernetes intricacies. Understanding how to configure and manage Flink clusters within a Kubernetes environment broadened my expertise in deploying and scaling stream processing applications effectively.

The learning curve involved a combination of self-study, practical application, and collaboration with experienced peers. Tackling real-world scenarios and troubleshooting challenges enhanced my understanding of Apache Flink and its ecosystem, providing a comprehensive skill set for building and maintaining sophisticated stream processing pipelines.

Firebase:

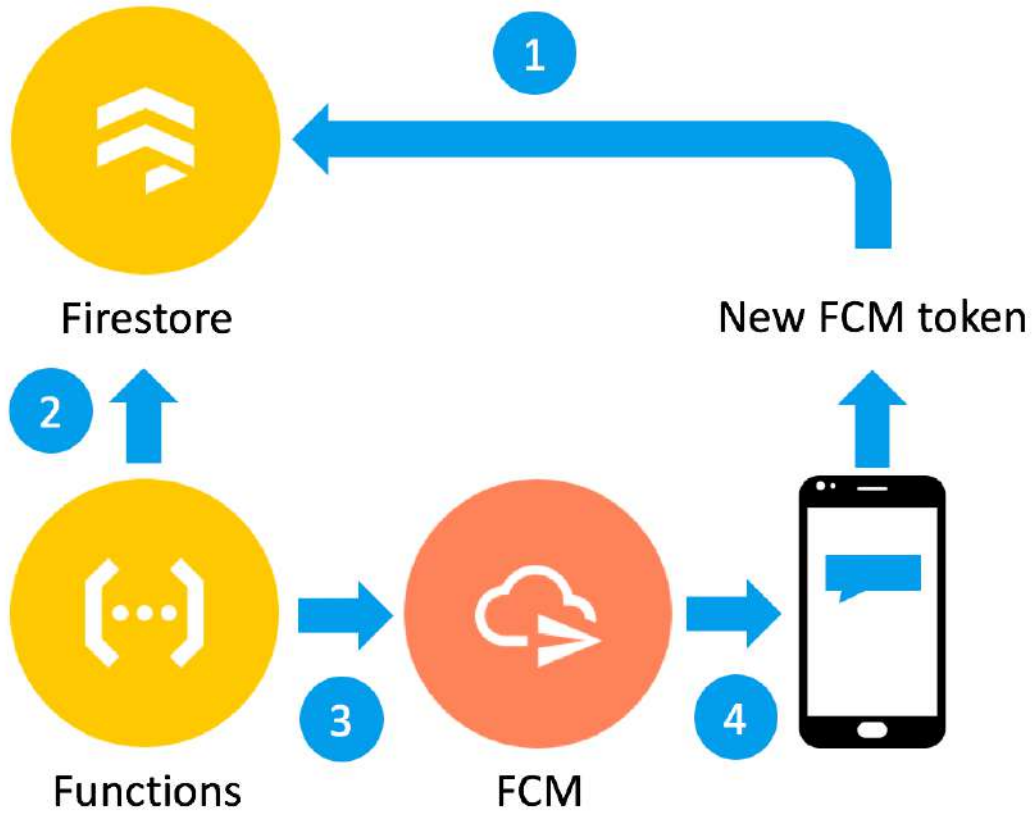


Figure 3: Steps to send push notifications via Firebase.

Firebase emerged as a pivotal technology stack, offering a seamless and integrated solution for various aspects of application development. Leveraging Firebase's suite of services, I focused on enhancing user engagement through Cloud Functions, Cloud Messaging, and Cloud Firestore.

Firebase Cloud Functions provided a serverless environment for executing backend code. I harnessed this capability to extend the functionality of my applications by deploying server-side logic in response to various triggers. Whether it was responding to changes in the database or handling authentication events, Cloud Functions empowered me to tailor the backend behaviour according to specific business requirements.

Firebase Cloud Messaging (FCM) played a crucial role in establishing effective communication channels with users. Utilising FCM, I implemented push notifications to reach out to customers in real-time. This not only enhanced user engagement but also enabled timely communication and updates, contributing to a more dynamic and responsive user experience.

Cloud Firestore served as the NoSQL database solution within Firebase, and I utilised it to store user FCM tokens. This allowed for efficient retrieval and management of user data, particularly relevant when orchestrating targeted push notifications through Cloud Messaging. The real-time synchronisation offered by Firestore further streamlined the handling of user data, ensuring that the most up-to-date information was readily available.

Practical application of Firebase involved a hands-on approach to configuring and deploying Cloud Functions, crafting FCM push notifications, and structuring data within Cloud Firestore. Integrating these Firebase services not only streamlined backend development but also significantly enhanced the user interaction capabilities of the applications.

Application of Knowledge Acquired

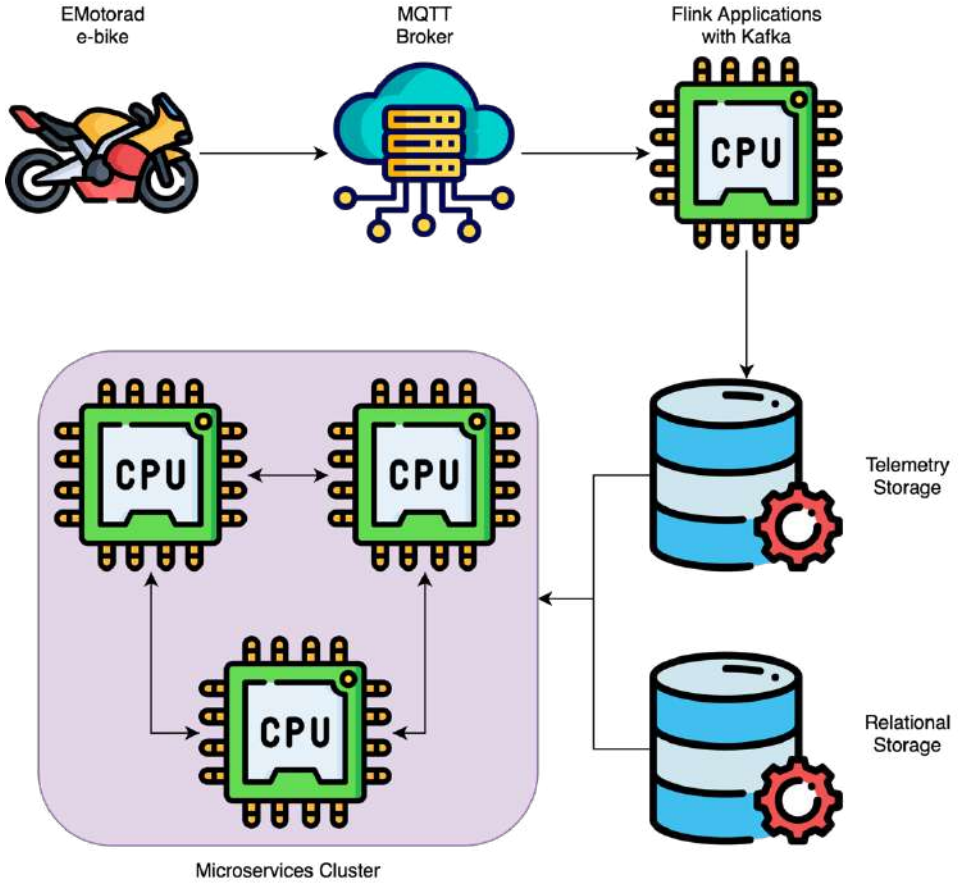


Figure 4: Implemented architecture diagram.

Microservices:

During my internship at EMotorad, I applied my knowledge of the Go programming language to develop a set of microservices that played a crucial role in shaping the EMotorad Companion App. These microservices, designed to enhance user experience and streamline various functionalities, showcased the versatility of Go in building robust and efficient applications.

I implemented a secure and user-friendly authentication mechanism. By integrating SMS-based authentication, users could seamlessly verify their identity. PASETO tokens, a modern and secure alternative to traditional JWTs, were employed to ensure secure and tamper-proof user authentication.

The user profile addition service allowed users to create and enhance their profiles. I crafted a seamless and intuitive process for users to input and update their personal

information, contributing to a more personalised and engaging user experience. I created microservices for mapping user profiles to their respective bikes. This functionality facilitated a streamlined association between users and their EMotorad bikes, offering a tailored experience based on individual bike specifications.

A crucial aspect of the EMotorad Companion App was real-time communication through MQTT. I developed an MQTT authentication service, ensuring secure and authenticated communication between devices and the server, contributing to a responsive and dynamic user experience.

I implemented microservices to fetch fitness data from the Cassandra database. The efficiency of Go allowed for seamless data retrieval and formatting, providing users with accurate and real-time insights into their fitness activities. The EMotorad Companion App offered users the capability to customise and set specifications for their bikes. I designed microservices that allowed users to define and adjust their bike specifications, tailoring the app to individual preferences.

I integrated Redis for efficient caching and quick data retrieval, optimising the performance of critical functionalities. PostgreSQL served as the main database, providing a reliable and scalable foundation for storing and managing diverse data sets integral to the app's operation.

This application of Go in microservices showcased the language's efficiency, conciseness, and suitability for building distributed systems. The EMotorad Companion App, powered by these Go-based microservices, not only delivered a seamless user experience but also underscored the adaptability of Go in creating robust and scalable applications.

Apache Flink Jobs/Streaming Applications:

I developed Apache Flink applications that played a pivotal role in facilitating seamless data processing and communication across various components of our system. These applications leveraged the power of Apache Flink to orchestrate real-time data streams efficiently.

One of the key Flink applications I crafted served as an MQTT bridge, acting as a crucial link between MQTT brokers and Kafka. This application subscribed to MQTT brokers, capturing incoming messages, and seamlessly pushed them to Kafka. By harnessing Flink's capabilities, I ensured robust and real-time communication, allowing for the efficient flow of messages between disparate systems.

Another Flink application focused on deserialising Protobuf messages from Kafka, transforming them into objects, and subsequently aggregating and sinking the processed data into Cassandra, Redis, and Kafka. Leveraging RocksDB as a state backend added a layer of efficiency to the stateful operations within the application. This stateful Flink job ensured that the processed data was not only aggregated but also maintained with fault tolerance and resilience.

The third application in my Flink portfolio tackled the deserialisation of Protobuf messages into objects. Following this, the application stored the serialised data in Parquet file format, which was then persisted in AWS S3. This Flink application demonstrated the versatility of Flink in handling diverse data processing tasks, from deserialisation to efficient storage in cloud-based object storage.

These applications collectively showcased the flexibility and scalability of Apache Flink in managing complex data workflows. Each application addressed specific requirements within the EMotorad ecosystem, contributing to a cohesive and efficient data processing pipeline.

Skillset After Training

My college curriculum has undeniably played a pivotal role in shaping the strong foundation upon which I built my experiences at EMotorad. The comprehensive coursework in programming languages, data structures, algorithms, and computer theory provided me with a solid understanding of fundamental concepts. The exposure to Java, Python, C, C++, and JavaScript, along with in-depth studies on sorting techniques, data structures, and algorithmic paradigms, laid the groundwork for my proficiency in various programming languages and problem-solving skills. Moreover, the exploration of computer hardware and architecture, coupled with a deep dive into operating systems, network protocols, and database systems, equipped me with a holistic understanding of the intricate interplay between software and hardware components. This robust academic background seamlessly aligned with the practical applications I encountered at EMotorad, allowing me to leverage theoretical knowledge in real-world scenarios and contribute effectively to the development of innovative solutions. The synergy between my college education and hands-on experiences during the internship has undoubtedly been instrumental in shaping my multifaceted skill set.

My internship at EMotorad has been an incredibly enriching journey, expanding my skill set and providing hands-on experience across a spectrum of technologies and practices. Throughout this experience, I've deepened my understanding of the Go programming language, leveraging its unique features for server-side development. The exploration of Go's concurrency model, particularly channels and goroutines, has empowered me to handle concurrent tasks efficiently, a skill paramount in modern application development.

Diving into containerisation with Docker has equipped me with the ability to create and manage containers, ensuring consistency and reproducibility in diverse environments. My foray into container orchestration using Docker Compose and Kubernetes reflects a comprehensive understanding of deploying and managing multi-container applications at scale, aligning with modern DevOps practices.

Amazon Web Services (AWS) became a significant part of my skill set, as I gained hands-on experience with services like Amazon RDS, Elastic Load Balancers, ECR, EKS, MemoryDB for Redis, Keyspaces for Cassandra, S3, and Amazon MSK. This proficiency in configuring, optimising, and integrating AWS services showcases my ability to architect and deploy applications on a cloud infrastructure.

Kubernetes has become a cornerstone of my expertise, with a deep understanding of its core components, deployment strategies, Helm charts, ingresses, and deployment on AWS EKS. This knowledge extends to efficiently managing containerised applications at scale, reflecting a grasp of contemporary container orchestration practices.

Venturing into Apache Flink demonstrated my competence in real-time data processing and stream processing applications. From developing Flink jobs for MQTT bridging to Protobuf deserialisation and efficient storage in AWS S3, I've showcased the ability to leverage Flink's capabilities in orchestrating complex data workflows.

Firebase has become a valuable addition to my toolkit, enabling me to enhance user engagement through Cloud Functions, Cloud Messaging, and Cloud Firestore. The integration of these Firebase services demonstrates my proficiency in tailoring backend logic and managing data in real-time for a dynamic user experience.

The application of my knowledge through the development of microservices for the EMotorad Companion App has been a highlight of my internship. From implementing secure authentication mechanisms to enabling real-time communication through MQTT and efficient data retrieval using Redis and PostgreSQL, my microservices architecture underscores my ability to design and implement distributed systems with a focus on performance and scalability.