Grenze International Journal of Engineering and Technology, Jan Issue

Design and Implementation of a Scalable IoT based Vehicle Tracking System

Ch.Vandana[1], R. Anirudh Reddy[2],Vasudeva Reddy T[3],Abhisarika Reddy T[4],Jyothi Supriya U[5] and Avinash Reddy Arutla[6]
[1-6]Department of ECE, B V Raju Institute of Technology, Narsapur, Telangana
Email: {vandana.ch, anirudhreddy.r, vasu.tatiparthi,19211a04m1,19211a04m8,19211a0417}@bvrit.ac.in

***Abstract*— Transport always did play a significant role in shaping how humans developed over the millennia. Movement of goods or people from source to destination has always been the primary goal offered, and it was always improved over centuries. There have been always due improvements and modern transportation systems have come a long way, however there are still challenges that need to be addressed. In this paper we present an optimized micro services based backend system with a cross platform mobile application built on Flutter for tracking a transport vehicle. The proposed database design and the mobile application, excel in all the areas of scalability, security, reliability, speed, compatibility, integration and cost effectiveness and proposes a new method of cheap and effective system development than the existing systems**

***Index Terms*— Full Stack Applications, Firebase, Scalable Backend Architecture, Flutter, Raspberry Pi, Vehicle Tracking and Management.**

I. INTRODUCTION

One of the specialized sectors of the automation industry, Internet of Things (IoT), is rapidly taking over the Wireless Sensor Network (WSN) market. There have been major breakthroughs and development in the IoT industry, most of them limited to Home and Industrial Automation. The high research and development costs make efficient use of this technology inaccessible to general public. Although there have been previous developments in generic vehicle tracking and management systems, the system design followed is outdated as per the current industry trends and technological advancements. These outdated systems although being used at a wide scale in private applications, they pose a significant risk of slower load times, database outages, and high scalability issues. To eliminate this problem, we propose a system which has been designed by modern microservices practices and use of a scalable cloud database called Firestore.

The live location feed of the system can be observed through a mobile application, which is designed on Flutter. Flutter is a modern flexible open-source SDK by Google. The usage of Flutter is recommended here due to its ability to build native looking applications from a single codebase. Flutter also provides APIs for accessing platform specific features and services. Firestore is a highly revered service in the high availability database domain. Firestore has been recommended here due to its easy scalability and management.

The current market for vehicle tracking systems is dominated by Automotive big players, the proposed work excels the archaic and pricy alternatives present in the market and in the previous works referred. The system can be easily modified according to the requirements of any enterprise or organization without the need of major system design alterations thus reducing costs and labor efforts.

Grenze ID: 01.GIJET.10.1.504

II. Literature Survey

Deebika Shree [1] provided a design technique for tracking passenger count inside a regular transport automotive. The algorithm used behind the updation of RFID count from multiple sensors over the network has been considered for formulating an improvised version in our approach.
Mostofa Kamal [2] designed a webhosted landing page and an android app. Their work paved path for an idea to maintain a local database to host GPS data and present it through a web solution.
Asif Ahmed [3] presented an application of safety tracking for school kids. Their paper presented an insightful idea for future safety implementations into the app.
Saurabh S. Chakole [4] implementation included the usage of a GSM module to relay the information in real time. This paper presented useful insights on how NMEA data can be decrypted and relayed to the necessary stakeholders.
Meet J Shah et al. [5] presented the idea on how the same system can be monetized. The paper also presented rigorous testing by implementing Monte Carlo Analysis. The monetization part was carried out by presenting money as credits inside the application and the method to top it up.
K. Gowri Subadra et al.'s [6] work has been useful in the understanding of older tracking systems. The comparison of ARM level implementation of tracking systems helped gain insights on the hardware's resource management. The usage of RFID sensors and GSM sensors to present with an announcement system for disabled people in bus stops helped with the database information delay time for our application.
K. Premkumar et al. [7] gave an introduction to the usage of Google Maps API for displaying the live location of the bus. This work has been especially important in getting a broader picture for the development of an android application.
Ricardo Salazar-Cabrera et al. [8] presented a review paper on sustainability of transit vehicles in a conjusted environment. The paper presented various inputs on how intelligent systems can manipulate the route path. The paper's insights on designing a maintainable cloud architecture helped design our backend system into individual service APIs.
Sharmin Akter et al. [9] presented a unique approach on the usage of cloud for ticketing. This paper helped us associate the later algorithm in our implementation of security checks in our system. The valid ticket check algorithm which checked whether each ticket is valid or was optimized and used in our system.
Hafiizh Nur M. A. et al. [10] presented a rapid tracking algorithm for finding the location of the bus. The BRT algorithm provided the system with the necessary speed it required. The use of real time database here helped them to achieve perfect response time in their applications.

III. Proposed Methodology

The proposed system places emphasis on the usage of newer technologies to build generic applications. The well working approach of the existing systems, is currently outdated per the new tech stacks which have been introduced in the modern IT world. This proposed system uses a combination of a highly reliable database and an easy to build and maintain mobile application. The proposed system can be broadly split into Private and Public type systems, while the hardware remains similar, the user facing application varies. To reduce the development time, the usage of a newer application framework has been taken into consideration here, Flutter. The mobile application is to be built on Flutter, a framework on the Dart language. Flutter provides immense customizability with its widgets, which provides a faster building time rather than the PHP or Java apps which held the previous systems. This framework supports native application speed for both Android and iOS platforms .Flutter also packs immense community support, which is essential for this cost efficient project.
Self managed databases and backend systems have been replaced by Firebase services in this system. The overall system has been broadly categorized into two categories

A. Public System

The public system doesn't have an authentication and authorization schema for the mobile application, since there isn't a need for information restriction present. The hardware access is restricted here to prevent unauthorized changes or damages to the enterprise under which it operates on.

B. Private System

The access for information is granted only to authorized users, where information security is the first priority. Educational institution buses and private vehicles are the primary examples here. The authentication service is provided by Firebase authentication. Since the activity cannot be predetermined, Firebase scales the resources

accordingly, which makes it an ideal choice here. Firestore database has been used here to store the location and passenger count details from the automotive. The columns in the database can be readily customizable using a CLI tool. The system hardware relies on Internet to update the data from the attached sensors, GPS and RFID sensors are assumed for results here. The hardware can consist of any microcomputer board which can run a Python script, the development for this system was done on a Raspberry Pi 4 for its versatility and computational power. An UART NEO-6M GPS sensor and a RC522 RFID sensor are interfaced to establish a proof of result for this system.

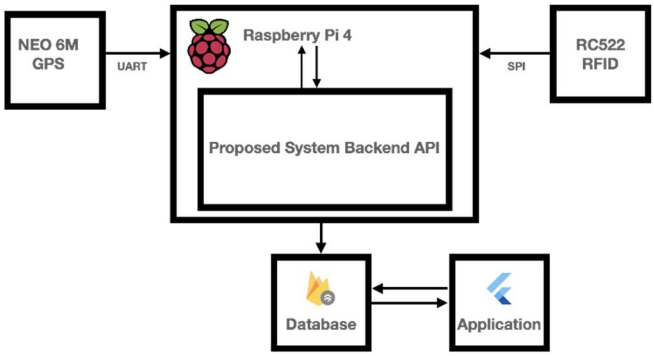

Fig 1. Hardware Workflow

A USB-C power adapter powers the hardware. On the initial boot, the system is designed to run a cron job to start the backend server which pushes the sensor data to Firestore database. Similar to a table in a Database, we have collections in Firestore database. Our architecture provides an unique Collection to every automobile which is brought onto this Smart Transport System platform.[6]The mobile application is built on the Flutter framework which is based on Dart language. The advantage of using this opposed to the earlier systems [2][3][5] is the development time on this framework is extensively less than its competitors. [5]

Logging into the application is fairly simple, and common for all the users. When a combination of username and password are entered, the application sends the credentials to Firebase for authentication. Depending on the type of token that Firebase returns, the user is categorized and taken to their respective screen on flutter. The main motive behind this system lies with its availability for customization between any entities, and using generic tools to build and maintain this system will provide cost effectiveness in the long term.

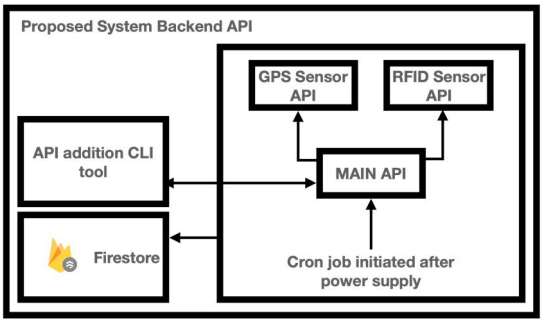

Fig 2. Backend Internal Working

The system has three user flows with it, the flows pertain to how the application behaves under the usage of different stakeholders in the maintenance and usage of the system. The flows are

- Admin Flow
- Passenger Flow
- Driver Flow

A. Admin Flow

The Admin flow is concerned with the behavior of the application, when the admin accesses it. The admin of the system has unlimited privileges on the accessing, restriction and blocking of information to the other stakeholders in any of the tracker modules (system) which are part of the admin's enterprise. Some of the functionalities an admin can control over the system are :

1. Altering details pertaining to the automotive

2. Blocking / Unblocking tracking details
3. Accessing the location of the bus
4. Adding more sensors to the system, thus adding more functionality.

Admin privileges will be granted by the root user who has access to the firebase authentication service. Initially, a root user is created abiding by the law of lowest permissions. The below mentioned flow is a sample flow of how, an admin can oversee an educational institution's bus.

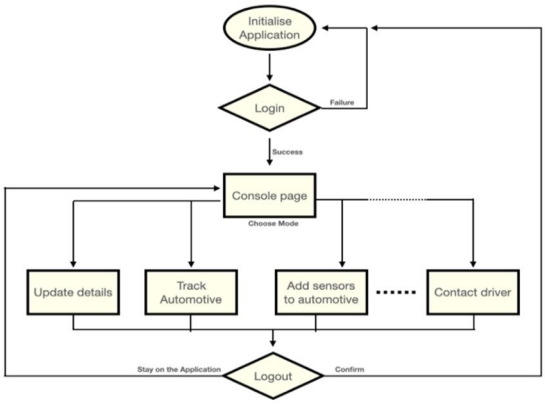

Fig 3. Admin Flow

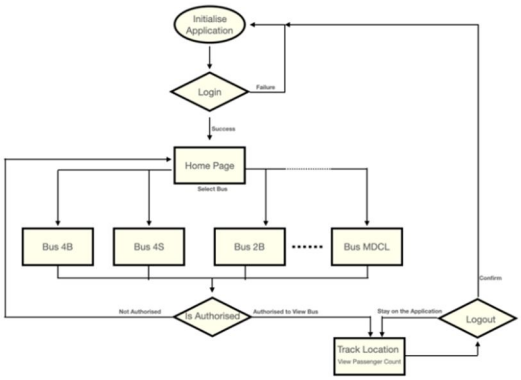

Fig 4. Passenger Flow

B. Passenger Flow

Users who are granted the required set of permissions by the admin, are able to access the network of automobiles under the organization/ enterprise are termed passengers here. The passengers here are able to access the functionalities which are enabled by the admin and are supported by the on board hardware system.[10]

In the example constructed to display the capabilities of the Smart Transport System, the user here will be able to access the live location of the bus, and the passengers count here. The passengers count will be updated using a RFID tag[3][6] which will be scanned by the passenger during entry and exit. The speciality of this system is its ability to scale up and down based on the requirements.

C. Driver Flow

The automotive driver has access to this flow. The login page remains same for all the users who use this application. As soon as the hardware system receives power, the cron job is initiated and the backend of the hardware starts running.[10] The driver is responsible for providing internet access to the system, either through a mobile hotspot or a vehicle internet facility which is provided by the organization/ enterprise under which the device is operating on

IV. RESULTS

The proposed methodology describes in great detail about the usage of Firebase authentication and Firestore. The Firestore database is a NoSQL database, and the data is stored in JSON format. Since there isn't any sensitive

content present on the Firestore database, it can be openly shown, whereas that isn't the case with Firebase Authentication where secret keys are visible. As can be inferred from Figure 6., the Firestore database has a number of collections each inferring to a unique vehicle number. Each collection contains a document, which can be inferred as a table in a database. This document contains our required fields such as latitude, longitude, last update time etc. The data from the GPS sensor is received in NMEA format, which is decoded and then pushed onto firebase.

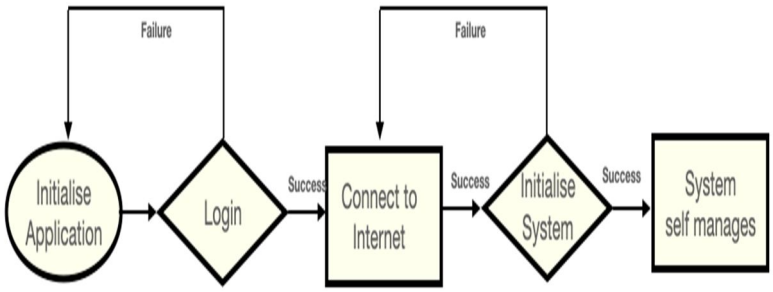

Fig 5. Driver Flow

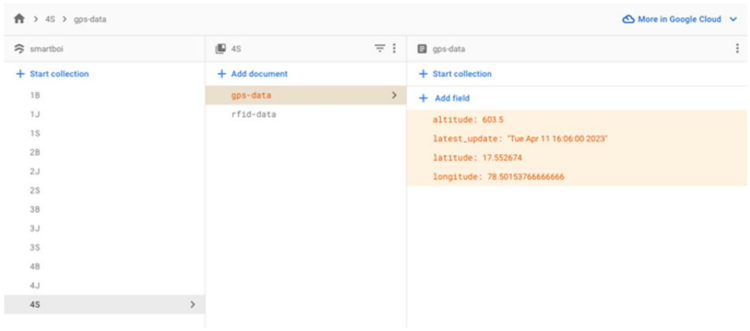

Fig 6. Firestore Databaser Live Update

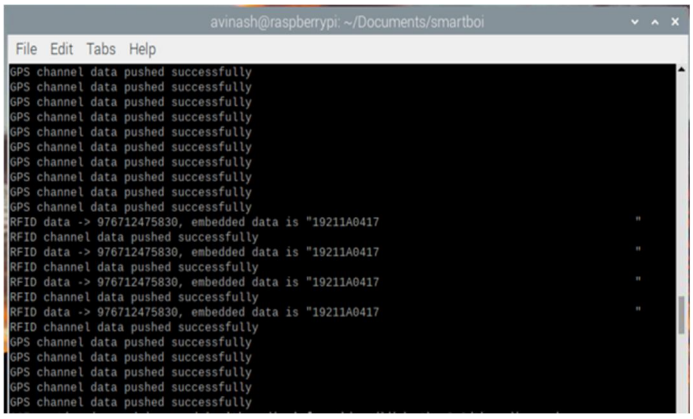

Fig 7. Hardware Console

In Figure 6., the columns inside document gps-data are colored "orange". This infers to the live updation of data. Similarly, the system backend can concurrently manage hundreds of instances such as these. The hardware has its own console which can be viewed by a VNC software. This console is helpful when debugging the system for any probable malfunctions.

The mobile application serves as the face of the project. As it can be inferred from Figure 8., there is a login screen present.[4] Any user who wishes to interact with the system, must use this common login screen , based on the entered credentials, the system will automatically take to the desired flow, either passenger , admin or driver.

The Figures 8.and 10.represent the immediate screen which is displayed after logging in as a passenger. Here all the available transport vehicles are listed on order, and when clicked on any one of them, the corresponding options are displayed.

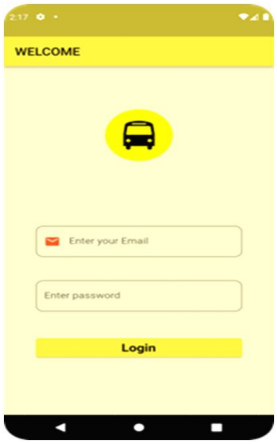

Fig 8. Common Login Screen

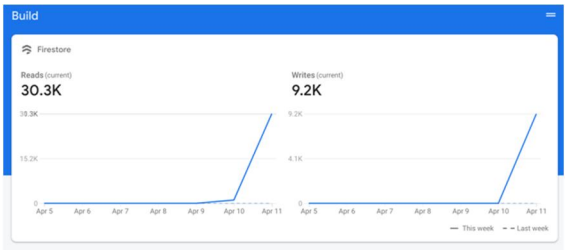

Fig 9. Firestore Analytics

Although a limited number of sensors such as the NEO-6M GPS sensor and a RC522 RFID sensor have been used to show the functioning ability of the system's backend architecture, depending on the hardware used, the system can accommodate even more sensors which can be used in other varying applications.[8]This system's novelty lies in its stability even when the user count is high. The pilot version of this application was released to a batch of 300 students over a period of 180 minutes. Unlike the existing systems [2][3][5][7][9] which were proposed earlier, this system observed no performance drops or app outages.

The above result clearly shows the bus location with the last updated time and the passenger count inside it. In some cases the hardware may pose limitations due to the number of sensors or the weather conditions it needs to be operated in, but the MVP of the system which is the backend code which runs on the hardware can handle humongous data whichever it has access to, and the non relational database on which this project lies on can scale as per requirement.

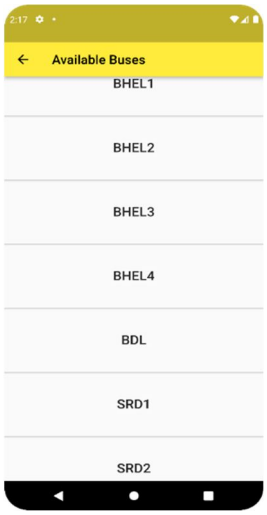

Fig 10. Available Buses Screen

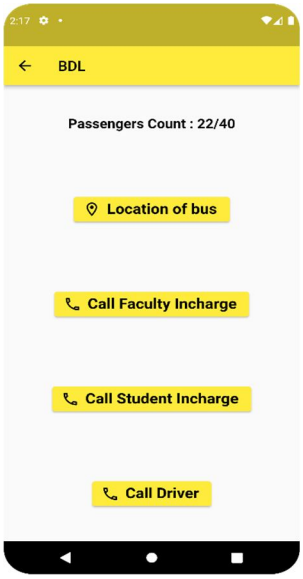

Fig 11. Bus main screen

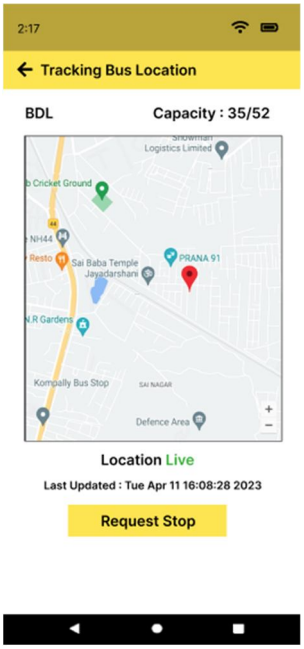

Fig 12. Bus Tracking Page

Figure 12. shows the hardware pilot product which was designed to carry out the design and development of this system.

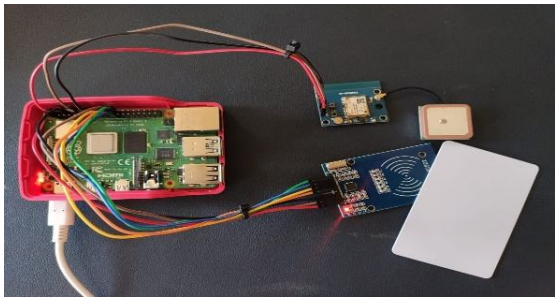

Fig 13. Hardware Prototype

TABLE I. PERFORMANCE COMPARISON

Comparison Metrics	**System Comparison**	
	Existing Systems	***Proposed System***
Application Platform	Android	Android, iOS
App Startup Latency	2-3 secs	300 ms
Location query response time	30-45 secs	~1 sec
No of simultaneous users	300+	100-130

TABLE II. PERFORMANCE METRICS

Proposed System Metrics			
Total time	***Read Cycles***	***Write Cycles***	***DB response time***
180 min	~30,300	~9,200	<100 ms

From Table 1. It can be clearly inferred that, the existing systems which were built on native SDKs were clearly lacking in speed due to the usage of novice unoptimized techniques, and the proposed system here, takes clear advantage of modern development techniques to provide faster information access to the users. The cloud based database served over 30K read cycles over a span of 180 minutes with no outages. This proves the clear capabilities of the system.
From Table 2. It can be clearly inferred that, the database concurrently performs closer to 30,300 read and 9,200 write cycles without any crashes, and the db response time appropriates to less than 100ms.

V. FUTURE SCOPE

The Future scope of this generic approach system lies with the further development and the ease it offers in achieving the same. Currently the database used for the backend is a Web2 hosted service, which particularly resides in a data center. The project can be expanded to operate on a open sourced Web3 database which uses Ethereum or any popular blockchain, this way the project becomes future proof and the costs for operating on a mass scale can be easily reduced. The system is currently optimized for private usage, i.e with restricted access. One of the project's future.
The project's admin portal is only present on a mobile platform in this build. This can be further extended to a web application, where the entire transport management activity of an institution or an enterprise or an organization can be easily managed. The proposed system currently relies on the usage of internet to process data. The project can be further improved by using decentralized tracking system, where nodes among the network communicate to give the user an accurate location.

V. CONCLUSION

A mobile application was proposed to be built on Flutter SDK. The built application supported near native application speeds, which were higher than the previously proposed system's mobile application speeds. The proposed backend architecture which managed the hardware has been implemented using microservices, which help in the easy management of sensor APIs. The database supporting the system has been rigorously tested with over 300 concurrent users without any usage issues, database outages or app crashes. This proves the reliability of the proposed system over the other present alternatives.
The backend from Figure 2. can be implemented on any microcomputer board which can run Python natively without any extra computational requirements. This makes the system adversely cost efficient.

REFERENCES

[1] Deebika Shree, J. Anusuya and S. Malathy, "Real Time Bus Tracking and Location Updation System," 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), Coimbatore, India, 2019 .
[2] O. M. Moushi, M. Kamal, M. Haque and M. S. Ahsan, "Design and Development of an Online Bus Monitoring System," 2018 10th International Conference on Electrical and Computer Engineering (ICECE), Dhaka, Bangladesh, 2018.

[3] A. Ahmed et al., "An Intelligent and Secured Tracking System for Monitoring School Bus," 2019 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2019.
[4] S. S. Chakole, N. A. Ukani and S. Sheikh, "GPS and GSM Enable Tracking, Monitoring and Control system for Multiple Application," 2019 International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2019.
[5] M. J. Shah, R. P. Prasad and A. S. Singh, "IOT Based Smart Bus System," 2020 3rd International Conference on Communication System, Computing and IT Applications (CSCITA), Mumbai, India, 2020.
[6] K. G. Subadra, J. M. Begum and H. Dhivya, "Analysis of an automated bus tracking system for metropolitan using IoT," 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, India, 2017.
[7] K. Premkumar, P. K., P. J., P. D. and P. P., "College Bus Tracking and Notification System," 2020 International Conference on System, Computation, Automation and Networking (ICSCAN), Pondicherry, India, 2020.
[8] Ricardo Salazar-Cabrera, Álvaro Pachón de la Cruz, Juan Manuel Madrid Molina, Sustainable transit vehicle tracking service, using intelligent transportation system services and emerging communication technologies: A review,Journal of Traffic and Transportation Engineering (English Edition),Volume 7, Issue 6,2020.
[9] S. Akter, T. Islam, R. F. Olanrewaju and A. A. Binyamin, "A Cloud-Based Bus Tracking System Based on Internet-of-Things Technology," 2019 7th International Conference on Mechatronics Engineering (ICOM), Putrajaya, Malaysia, 2019.
[10] M. A. Hafiizh Nur, S. Hadiyoso, F. B. Belladina, D. N. Ramadan and I. Wijayanto, "Tracking, Arrival Time Estimator, and Passenger Information System on Bus Rapid Transit (BRT)," 2020 8th International Conference on Information and Communication Technology (ICoICT), Yogyakarta, Indonesia, 2020.