

BUS TRACKING APPLICATION

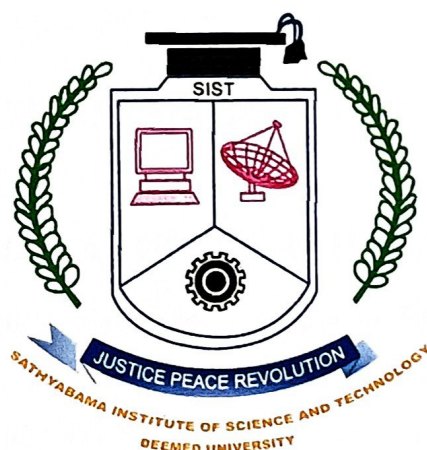
Submitted in partial fulfillment of the required for the award of

Bachelors of Computer Science

by

LAKSHUVASAN V.M

(REGISTER NO: 40290051)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF SCIENCE AND HUMANITIES

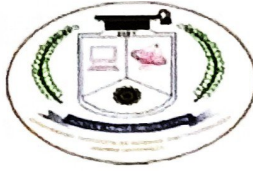
SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC| Approved by AICTE

JEPPIAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600 119

MAY 2023



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **LAKSHUVASAN V.M (40290051)** who carried out the project entitled **"BUS TRACKING APPLICATION"** under our supervision from **JANUARY 2023 TO MAY 2023.**

Shamili .v
Internal Guide

Ms.V.SHAMILI M.Sc.,

Rekha
6/5/23

DEAN AND HEAD OF THE DEPARTMENT

Dr.REKHA CHAKRAVARTHI M.E., Ph.D.,

Submitted for Viva voice examination held on

08.05.2023

6/5/23
Internal Examiner

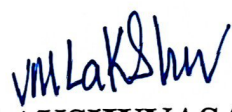
R. Chakravarthi
External Examiner

DECLARATION

I am, **LAKSHUVASAN V.M (40290051)** hereby declare that the Project Report entitled "**BUS TRACKING APPLICATION**" done by us under the guidance of **Ms.V.SHAMILI M.Sc.**, (Assistant Professor) Department of Computer Science at **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**, Jeppiar Nagar, Rajiv Gandhi Salai, Chennai-600 119 is submitted in partial fulfillment of the requirements for the award of Bachelors of Science degree in Computer Science.

DATE: 08.05.2023

PLACE: CHENNAI


LAKSHUVASAN V.M

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for Their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to Dr. Rekha Chakravarthi M.E., Ph.D., Dean of Arts And Science School of Computing, Dr. L. Lakshmanan M.E., Ph.D., Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **Ms. V. SHAMILI M.Sc.**, Assistant Professor **Department of Computer Science** for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff member of the Department of Computer Science who were helpful in many ways for the completion of the project

LAKSHUVASAN V.M

ABSTRACT

The college bus tracking app is a cross-platform mobile application developed using React Native, Firebase, and Tailwind CSS. The app aims to provide a reliable and efficient transportation solution for college students and faculty by utilizing real-time GPS data to track buses and provide accurate arrival times.

The app has several features, including a user-friendly interface that allows users to easily view bus routes, schedules, and real-time locations. The app also allows users to plan their trips by providing estimated arrival times, and push notifications can be sent to alert users when the bus is nearing their location.

The app is powered by Firebase, which provides a robust and secure backend infrastructure for data storage, user authentication, and messaging. The app also utilizes Tailwind CSS, a utility-first CSS framework, to provide a responsive and customizable user interface that can be easily adapted to suit the college's branding and design standards.

Overall, the college bus tracking app is a powerful and convenient solution for college transportation needs, providing reliable and up-to-date information to students and faculty. With its intuitive interface, real-time tracking, and powerful backend infrastructure, the app can improve student attendance, reduce transportation costs, and enhance overall campus safety.

TABLE CONTENT

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	8
1.1	DOMAIN INTRODUCTION	8
1.2	OBJECTIVE OF THE PROJRCT	9
1.3	PROJECT INTRODUCTION	9
2	LITERTURE SURVEY	10
2.1	REVIEW ON LITERATURE WORK	10
2.2	EXISTING SYSTEM	13
2.3	PROPOSED SYSTEM	14
3	METHODOLOGY	15
3.1	REQUIREMENT ANALYSIS	15
3.1.1	HARDWARE ENVIRONMENT	15
3.1.2	SOFTWARE REQUIRMENT	15
3.2	SYSTEM ALGORITHM	15
3.3	SYSTEM DESIGN	17
3.3.1	ARCHITECTURE DIAGRAM	17
3.3.2	USECASE DIAGRAM	18
3.3.3	ACTIVITY DIAGRAM	19
3.4	MODULES DESCRIPTION	20
3.4.1	LOGIN PAGE	20
3.4.2	REGISTER PAGE	20
3.4.3	USER HOME	21

3.4.4	DRIVER HOME	22
3.4.5	SEARCH PAGE	22
3.4.6	MAP PAGE	23
3.4.7	ALERT BOX	24
4	CONCLUSION/FUTURE WORK	25
5	SOURCE CODE	26
5.1	SCREEN SHOTS	41
5.2	REFERENCES	46

CHAPTER 1

INTRODUCTION

1.1 DOMAIN INTRODUCTION

What is meant by full stack development? Full stack development is the end-to-end development of applications. It includes both the front end and back end of an application. The front end is usually accessed by a client, and the back end forms the core of the application where all the business logic is applied.

React Native

React Native is an open-source UI software framework created by Meta Platforms, Inc. It is used to develop applications for Android, Android TV, iOS, macOS, and tvOS, Web, Windows and UWP by enabling developers to use the React framework along with native platform capabilities. It is used to develop the Android and iOS applications at Facebook, Microsoft, and Shopify. It is also being used to develop virtual reality applications at Oculus.

Firebase

Firebase is a set of backend cloud computing services and application development platforms provided by Google. It hosts databases, services, authentication, and integration for a variety of applications, including Android, iOS, JavaScript, Node.js, Java, Unity, PHP, and C++.

Tailwind CSS

Tailwind CSS is an open source CSS framework. The main feature of this library is that, unlike other CSS frameworks like Bootstrap, it does not provide a series of predefined classes for elements such as buttons or tables. Instead, it creates a list of "utility" CSS classes that can be used to style each element by mixing and matching.

GOOGLE MAPS API

These are google API which are used to intergrade google maps and its feature into the application

1.2 OBJECTIVE OF THE PROJECT

The aim of our project is to provide exact live location of the buses on the route and all the busses on the route.

1.3 PROJECT INTRODUCTION

College bus transportation is an essential part of campus life for many students, faculty, and staff. However, the reliability and efficiency of college bus services can be a concern, particularly when it comes to tracking the location and arrival times of buses. A college bus tracking application is a solution that can address these concerns by providing real-time tracking of buses and their expected arrival times.

Our proposed project aims to develop a college bus tracking application that can enhance the user experience of using campus transportation. The application will provide up-to-date information on bus locations, estimated arrival times, and any delays or changes in bus schedules. This information can help students, faculty, and staff plan their journeys more effectively, reducing waiting times and improving the overall transit experience.

The college bus tracking application will also provide a range of features, such as route mapping, bus locations, which can help users navigate the bus system more easily. Additionally, the application will have a user-friendly interface, making it easy to use for people of all ages and backgrounds.

Overall, the college bus tracking application will provide a reliable, convenient, and accessible way for people to access information about campus transportation in real-time. By doing so, the application can help promote the use of college bus transportation and improve the overall transportation experience for students, faculty, and staff.

CHAPTER 2

LITERATURE SURVEY

2.1 REVIEW ON LITERATURE WORK

Paper 1:

Abstract:

The BUS TRACKING AND STUDENT MONITORING SYSTEM is a mobile application that ensure security to the students. The main objective of this application is to build a vehicle monitoring system for school children. Nowadays, parents are perturbed about children going to school because of the increasing number of cases of missing students. On certain occasions, students need to wait a much longer time for arrival of their school bus; our proposed system provides a technical solution for the above scenario. The system consists of an Internet enabled android application which will interact with a server. Authentication, attendance monitoring, vehicle tracking are the other features provided by the system. Driver, Faculty/Teachers, Parents, and Administrators are the end users of the application. The application uses wireless technologies like GPS and GPRS/GSM. With this android application parents can watch all the movements of the bus and monitor their child's presence.

Paper 2:

Abstract:

The use of private vehicles intensifies the existing unbearable traffic jam and majority of the parents consider school buses to be largely unsafe for their kids. However, safe and user friendly school buses can reduce the use of private vehicles and will eventually cut back the traffic jam in cities especially in school hours. All buses can be tracked by the guardians using the proposed intelligent and secured tracking system for school buses. This proposed system includes an Android application that can be employed to send notifications during entry and exit of the students using radio frequency identification (RFID) as well as Fingerprint scanner, ensuring arrival confirmation of the student by dual authentication. The system will also notify parents when the bus is a 15-min distance away from the home. To ensure on-board safety, 'Window Signalling' will also be used. The focus of the research is to display the feasibility of a safe and intelligent school bus using secured tracking system based on dual authentication procedures. It is expected that the offshoot of this research initiative will regain the confidence and reliability of parents in school bus and reduce

the use of individual transport. A smart school bus will keep the student safe, easing the tension for parents and the city will have a smooth traffic system.

Paper 3:

Abstract:

This research is based on Bus Tracking System, The main objective of this work is to make a GPS tracking application which would be able to track school buses more accurately and efficiently than present bus-tracking systems. We got the motivation for our research paper by observing how much precious time of the students studying in our college is wasted by just waiting for the bus on their particular route, and also there is a lack of information for the buses which are traveling on their particular route for the students who have already missed their particular bus. This work consists of three parts. In the first part, the bus location is received from the satellite and then further it is processed through the Kalman filter and then it is sent to dedicated web servers with the help of cellular networks. In the second part, the coordinates received is processed through the Google Maps API. In the third part, the data received from Google Maps API is processed in user's device to show real-time location.

Paper 4:

Abstract:

This paper proposes an Android phone application that gives information about transports, transport numbers similarly as transport services both on the web and Reason behind Android organize Android requires an open source enhancement which is probably the most reachable and a present straightforward methodology. This paper also oversees Location Based Services, which are used to pursue the present territory of the vehicle similarly as give a measure remaining time for the pursued transport to accomplish its objective using the client - server development. Moreover it demonstrate the required maps with the help of GPS.

Paper 5:

Abstract:

The proposed smart school bus tracking system is an easy-to-use software, including both a web-based program and a mobile application, that mainly gives parents, students and school service firms the ability to track accurately the location of their school service vehicles. Such a smart school bus service information system is inevitable within the context of smart city features. The mobile application makes it smarter for kids/students using their school services; also, parents can easily track their kids' buses. In addition, bus service companies can easily register new students and determine the routes of the school buses dynamically. The system provides scalability, flexibility, low cost, security, and reliability. All shared location information and users' private data are stored encrypted; the parents may receive information about their related buses only. The system is based on both web and mobile platforms and it is implemented as an interactive application.

Paper 6

Abstract:

It is very difficult for school bus drivers to monitor and track children when they pick-up or to drop-off children without human errors which may lead to tragedy or even death. We have developed a school bus tracking and students monitoring system which provide a comprehensive and highly reliable integrated services needed for school buses to be tracked, monitored and inquired on Google maps in real time. The system offers a web and SMS services so that a real time data and information such as the location of the bus, boarding and disembarking status of students riding the bus are shared with parents and school administrators in real-time. The possibility of missing children's during school bus transportation or accidentally left them sleeping inside the bus is eliminated.

2.2 EXISTING SYSTEM

The current system used in many colleges to know the location of the bus is often inefficient and unreliable. Typically, students rely on either calling a friend who has already boarded the bus or assuming the bus timing for their stop from the time of departure at the first stop which is given in the pdf circular. Calling a friend who has already boarded the bus can be unreliable, as students may not always be able to get in touch with their friends or may not have accurate information about the location of the bus. This can lead to confusion and frustration, as students may be left waiting at their stop for extended periods of time without any information about when the bus will arrive. Similarly, relying on the time of departure at the first stop can also be unreliable, as traffic and other issues can cause delays or disruptions to the bus schedule. This can leave students waiting at their stop for extended periods of time, without any accurate information about when the bus will arrive.

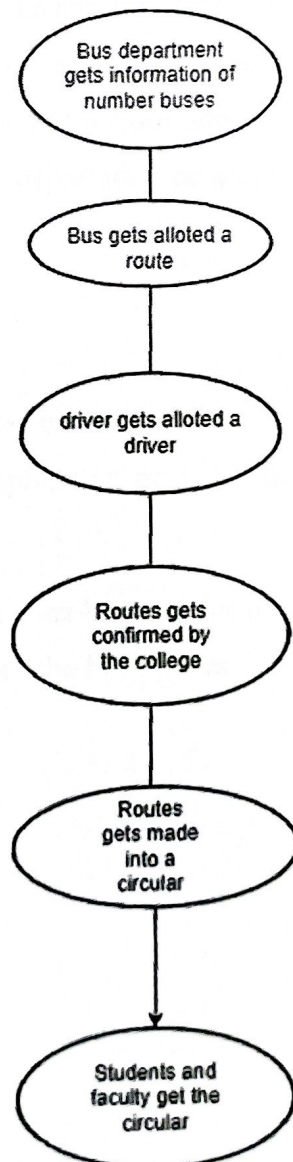


Fig.2.1

DISADVANTAGES:

- Its unreliable
- It cannot be updated frequently
- It does not provide the real time location of bus

2.3 PROPOSED SYSTEM

It is very difficult for school bus drivers to monitor and track children when they pick-up or to drop-off children without human errors which may lead to tragedy or even death. We have developed a school bus tracking and students monitoring system which provide a comprehensive and highly reliable integrated services needed for school buses to be tracked, monitored and inquired on Google maps in real time. The system offers a web and SMS services so that a real time data and information such as the location of the bus, boarding and disembarking status of students riding the bus are shared with parents and school administrators in real-time. The possibility of missing children's during school bus transportation or accidentally left them sleeping inside the bus is eliminated.

ADVANTAGES

- The proposed system **provides live location of the bus.**
- The user provides the area and the application provides the user with all the bus number going through that area.
- Due to this application no one will miss the bus and for new students and staff this will save them from standing in their stop hours before the bus arrives.

CHAPTER 3

METHODOLOGY

3.1 REQUIREMENT ANALYSIS

Requirement analysis determines the requirements of a new system. This project analyses on product and resource requirement, which is required for this successful system. The product requirement includes input and output requirements it gives the wants in term of input to produce the required output. The resource requirements give in brief about the software and hardware that are needed to achieve the required functionality.

3.1.1 Hardware Environment

- ✓ Hard Disk : 150 MB
- ✓ RAM : 4GB and Above
- ✓ Processor : QUALCOMM 6th gen

3.1.2 Software Requirements

- ✓ Operating System : Android 11 or above
- ✓ Software : Google maps
- ✓ Tools : Location service

3.2 System Algorithm

MESSAGE BROKER ALGORITHM

RabbitMQ is the most widely deployed open source message broker with tens of thousands of users, RabbitMQ is one of the most popular open source message brokers. From T-Mobile to Runtastic, RabbitMQ is used worldwide at small startups and large enterprises. RabbitMQ is lightweight and easy to deploy on premises and in the cloud. It supports multiple messaging protocols. RabbitMQ can be deployed in distributed and federated configurations to meet high-scale, high-availability requirements. RabbitMQ runs on many operating systems and cloud environments, and provides a wide range of developer tools for most popular languages.

RABBIT MQ uses AMQP protocol

AMQP 0-9-1 (Advanced Message Queuing Protocol) is a messaging protocol that enables conforming client applications to communicate with conforming messaging middleware brokers. Brokers and Their Role Messaging brokers receive messages from publishers (applications that publish them, also known as producers) and route them to consumers (applications that process them). Since it is a network protocol, the publishers, consumers and the broker can all reside on different machines. AMQP 0-9-1 Model in Brief the AMQP 0-9-1 Model has the following view of the world: messages are published to exchanges, which are often compared to post offices or mailboxes. Exchanges then distribute message copies to queues using rules called bindings. Then the broker either deliver messages to consumers subscribed to queues, or consumers fetch/pull messages from queues on demand. Exchanges are AMQP 0-9-1 entities where messages are sent to. Exchanges take a message and route it into zero or more queues.

Default Exchange

The default exchange is a direct exchange with no name (empty string) pre-declared by the broker. It has one special property that makes it very useful for simple applications: every queue that is created is automatically bound to it with a routing key which is the same as the queue name.

For example, when you declare a queue with the name of "search-indexing-online", the AMQP 0-9-1 broker will bind it to the default exchange using "search-indexing-online" as the routing key (in this context sometimes referred to as the binding key). Therefore, a message published to the default exchange with the routing key "search-indexing-online" will be routed to the queue "search-indexing-online". In other words, the default exchange makes it seem like it is possible to deliver messages directly to queues, even though that is not technically what is happening.

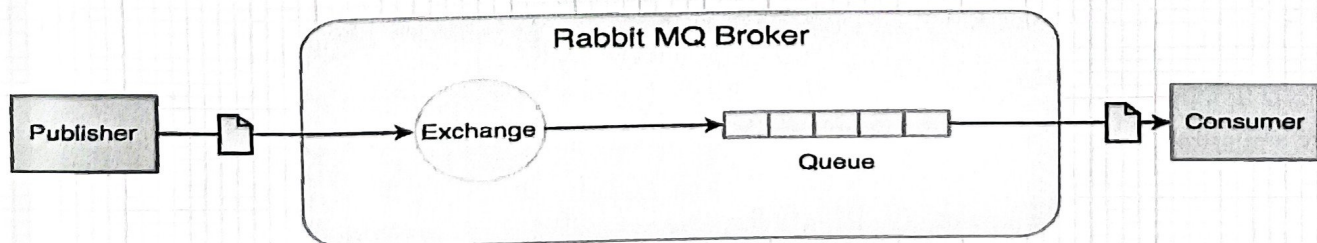


Fig. 3.1

3.3 SYSTEM DESIGN

3.3.1 ARCHITECTURE DIAGRAM

An architecture diagram is a visual representation of a system's structure and components, including their relationships and interactions. It provides a high-level view of the system's design and helps stakeholders understand its various components and how they work together. Architecture diagrams are an important tool for software developers, system architects, and project managers. They provide a clear and concise way to communicate complex systems to technical and non-technical stakeholders alike. They can help identify potential issues or bottlenecks in a system's design, as well as opportunities for optimization or improvement. The introduction to an architecture diagram typically includes an overview of the system and its purpose. It may also describe the key stakeholders and users of the system, as well as any specific requirements or constraints that need to be considered in the design process. Overall, architecture diagrams provide a powerful tool for designing, communicating, and optimizing complex systems. By providing a clear and concise view of a system's design, they can help ensure that all stakeholders have a shared understanding of the system's structure and functionality.

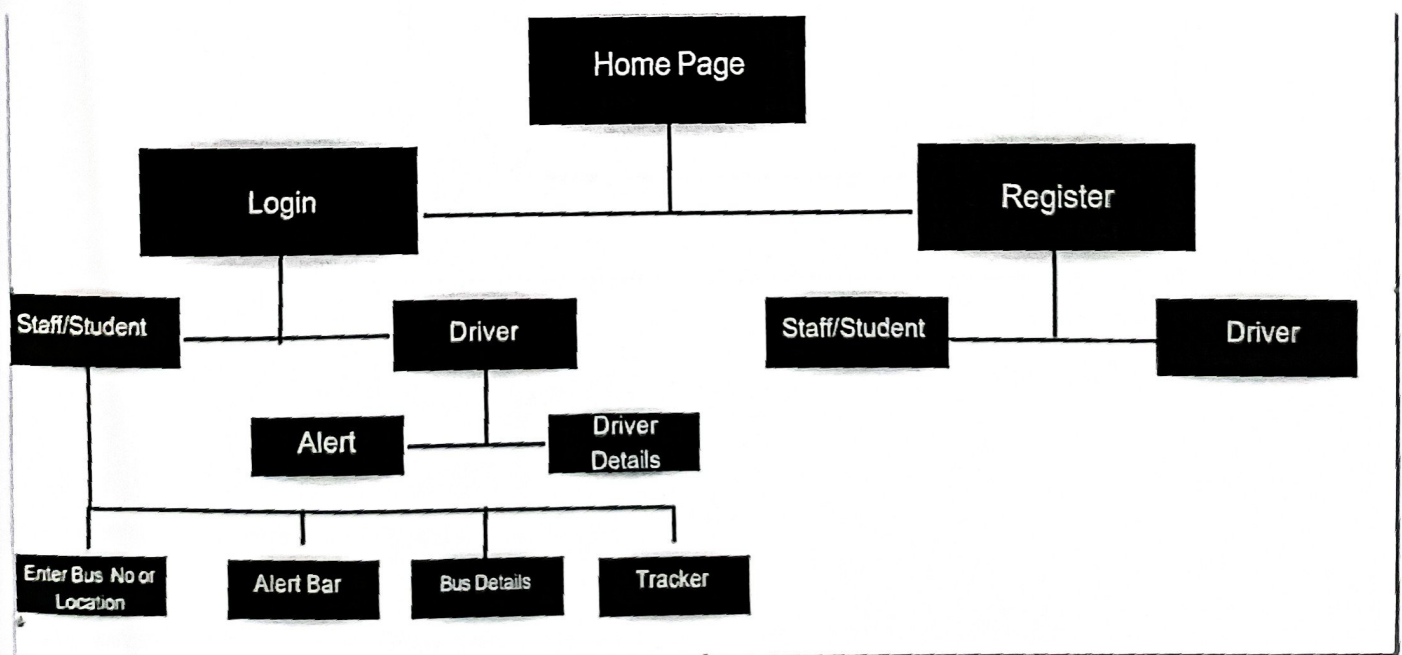


Fig: 3.2

3.3.2 USECASE DIAGRAM

Use case diagrams overview the usage requirement for system. They are useful for presentations to management and/or project prediction, but for actual development you will find that use cases provide significantly more value because they describe “the meant” of the actual requirements. A use case describes a sequence of action that provides something of measurable value to an action and is drawn as a horizontal ellipse.

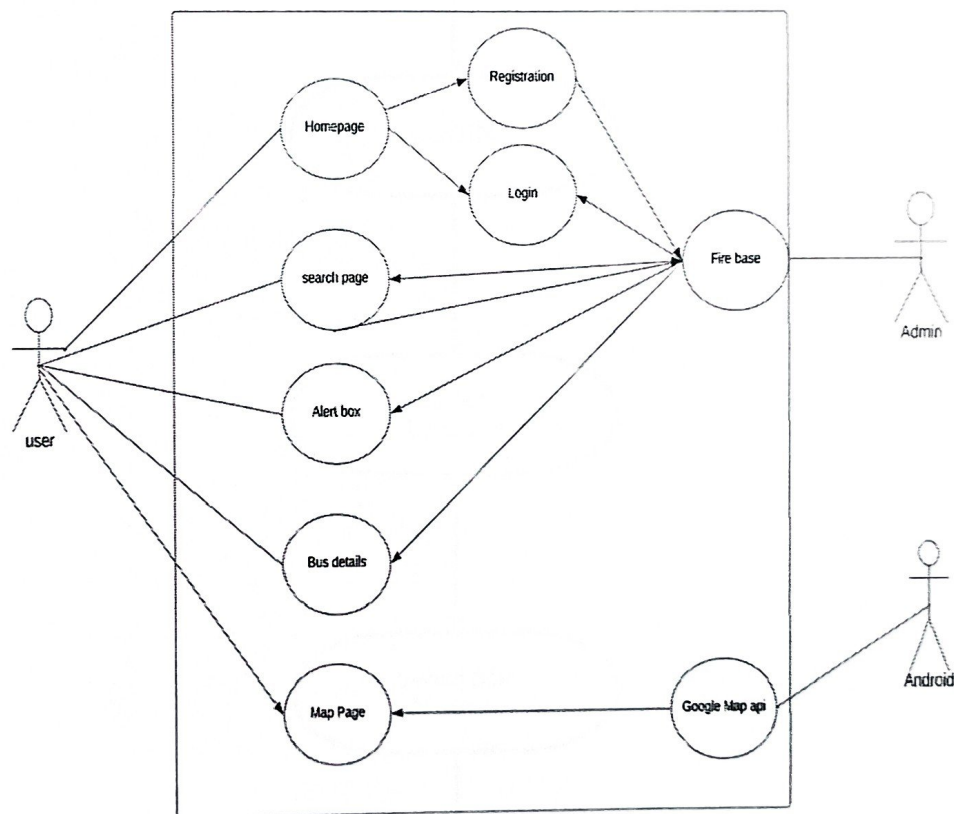


Fig: 3.3

3.3.3 ACTIVITY DIAGRAM

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.

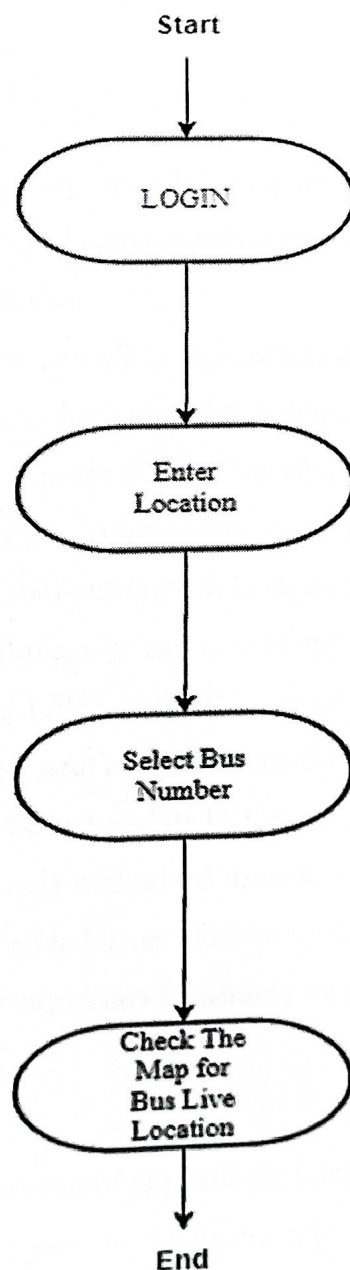


Fig: 3.4

3.3.4 Module Description

- **Login Page**
- **Register Page**
- **User Home**
- **Driver Home**
- **Search Page**
- **Map Page**
- **Alert Box**

3.3.4.1 Login page

The login page is the first page that users see when they open a college bus tracking app. This page is designed to provide a secure and convenient way for users to access the app and access the features and functionality that it provides.

To access the app, users must first enter their login credentials, which typically include a username and password. Once the user has entered their login information, the app verifies their credentials and grants access to the appropriate user interface.

On the login page, a toggle button is also provided to switch between user and driver login. This feature allows users to easily switch between different types of user interfaces depending on their role within the bus system. For example, if a user is a driver, they can easily switch to the driver interface and access the features and functionality that are relevant to their role.

The login page is an important component of any college bus tracking app, providing a secure and convenient way for users to access the app and its features. By requiring users to enter their login credentials, the app can ensure that only authorized users have access to the bus system and its data. The toggle button provides an added layer of convenience and flexibility, allowing users to easily switch between different user interfaces depending on their role within the bus system.

3.3.4.2 Register Page

The registration page is a key component of any college bus tracking app, as it provides a way for new users or drivers to sign up and gain access to the app. This page typically requires users or drivers to enter their personal details, including their name, email, and password, which are then saved to the app's database, typically using a service like Firebase.

To register, users or drivers must first navigate to the registration page and enter their personal details in the designated fields. Once they have entered their details, they can submit the registration form, and their details will be saved to the app's database.

The registration page is an important feature of the app, as it allows new users or drivers to quickly and easily gain access to the app and its features. By requiring users or drivers to register, the app can ensure that only authorized users or drivers have access to the bus system and its data. The use of a service like Firebase to store user data is a common approach for mobile app development. Firebase provides a cloud-based backend service that allows app developers to easily store and manage user data, as well as providing features like authentication and real-time data synchronization.

Overall, the registration page is a key feature of any college bus tracking app, providing a way for new users or drivers to sign up and gain access to the app's features, while also ensuring that the app remains secure and only authorized users or drivers have access to the system and its data.

3.3.4.3 User Home

The user home page is the main interface that users see after logging into the college bus tracking app with their credentials. This page is designed to provide users with easy access to the key features and functionality of the app, including the search page, a map for verifying the user's location, and an alert box.

The search page allows users to search for buses based on route numbers, which is a key feature of any bus tracking app. Once the user has selected their route number, they can see the live location of the bus, the distance between the route the bus is taking, and the estimated time of arrival of the bus to the user. This provides users with real-time information about the location and timing of the bus, which can help them plan their travel more efficiently.

The map on the user home page allows users to verify their current location, which is important for ensuring that they are able to accurately track the bus and plan their travel accordingly. By using the map, users can easily see their current location and compare it to the bus route to determine the best route to take to the bus stop.

Overall, the user home page is a key component of any college bus tracking app, providing users with easy access to the key features and functionality of the app, including search, maps, and alerts. By providing users with real-time information about the location and timing of the bus, the app can help them plan their travel more efficiently and avoid delays or issues along the way.

3.4.4 Driver Home

The driver home page is a crucial component of the college bus tracking app, designed to provide drivers with access to key information about their bus, route, and passengers. This page typically includes several key features, including the bus number, the route of the bus, the alert box, and driver details.

The bus number is a critical piece of information for drivers, as it allows them to easily identify their bus and ensure that they are driving the correct vehicle. This information is typically displayed prominently on the driver home page, along with other key details about the bus and route.

In addition to the bus number, the driver home page typically includes information about the route of the bus, including the starting point, end point, and any stops along the way. This information can help drivers plan their route and ensure that they stay on schedule, even in the event of traffic or other issues.

The alert box on the driver home page is an important feature that allows drivers to update users with any notification/alert messages such as bus breakdown, accident, and roadblock. By updating the alert box with important information, drivers can help keep passengers informed and ensure that they are able to make alternate travel arrangements if necessary.

Finally, the driver home page typically includes information about the driver, including their name, contact information, and any other relevant details. This information can be useful for passengers who need to get in touch with the driver or have any questions or concerns about the route or schedule.

Overall, the driver home page is a critical component of the college bus tracking app, providing drivers with access to key information about their bus, route, and passengers. By providing drivers with real-time information about their route and any potential issues, the app can help ensure that passengers are able to travel safely and efficiently.

3.4.5 Search Page

The current location feature is an essential component of a college bus tracking app that allows users to input their current location and view the buses that are currently operating in their area. This feature provides a convenient and intuitive way for users to plan their trips and navigate the bus system.

When a user inputs their current location, the app retrieves real-time GPS data from the buses

operating in the area and displays their locations on a map. Users can then see the buses that are currently operating in their vicinity, as well as the routes they are taking and the estimated time of arrival (ETA) of each bus to the user's location.

This feature is especially useful for users who are new to the area or are unfamiliar with the bus system. By providing a clear and concise view of the buses operating in the area, users can plan their trips more efficiently and avoid unnecessary delays or confusion.

In addition to providing information about the buses operating in the area, the current location feature can also be used to identify potential issues or delays in the bus system. For example, if there are no buses operating in the user's area or if there are significant delays or detours on the bus routes, users can adjust their plans accordingly and avoid any potential issues.

Overall, the current location feature is a component of college bus tracking app that provides users with real-time information about the buses operating in their area and helping them to plan their trips more efficiently. By providing a clear and concise view of the bus system and its operations, the app can improve the overall user experience and increase the efficiency and reliability of the bus system.

3.4.6 Map page

The user interface of a college bus tracking app typically includes a page where the user can select the route number and view the live location of the bus. This page provides a clear and concise overview of the bus system and allows users to plan their trips more efficiently.

When the user selects a route number, the app retrieves real-time GPS data from the buses on that route and displays the location of each bus on a map. The user can then see the current position of the bus relative to the route it is taking, as well as the estimated time of arrival (ETA) of the bus to the user's location.

The app may also display additional information, such as the distance between the bus and the user's location, the current traffic conditions, and any potential delays or detours on the route. This information can help users to plan their trips more effectively and make informed decisions about their transportation options.

Overall, the "Select Route" page is an essential component of college bus tracking app, providing users with real-time information about the bus system and helping them to plan their trips more efficiently. By providing a clear and concise view of the location and ETA of the buses on each route, the app can improve the overall user experience and increase the efficiency and reliability of the bus system.

3.4.7 Alert Box

The notification or alert box is a feature that is present in both the user and driver home screens of a college bus tracking app. This feature is designed to provide important updates and messages to users about the status of the bus system.

When a driver encounters an issue on the road, such as a breakdown, accident, or roadblock, they can use the app to update the status of the bus and notify the users of the issue. The notification box allows the driver to quickly and easily communicate any updates or changes in the status of the bus system to the users.

When the driver posts a notification, the message will appear in the alert box on the user home screen. This allows users to stay informed about any potential delays or issues with the bus system, allowing them to adjust their plans accordingly.

The alert box can also be used to communicate other important messages to users, such as changes to bus schedules or routes, special events or promotions, or other important announcements. By providing a centralized location for users to access important information about the bus system, the alert box can improve the overall user experience and increase user engagement with the app.

Overall, the notification or alert box is a feature of college bus tracking app, that provides a convenient way for drivers to communicate important updates and messages to users and helping to keep users informed and up-to-date about the status of the bus system.

CHAPTER-4

CONCLUSION / FUTURE WORK

4.1 CONCLUSION

In summary, the college bus tracking app provides a convenient and efficient solution for students and staff to stay up-to-date with changes in the bus route and track the location of their bus. This feature is particularly useful for new students or staff, as well as those who have changed their stopping location. Additionally, the app can help the management team take action more quickly in the event of accidents, breakdown, running out of fuel, or other bus malfunctions. Overall, the app streamlines the bus transportation process, providing a more reliable and convenient experience for all users.

4.2 FUTURE WORK

There are several potential areas for future work in the development of an Android college bus tracking app. Some potential areas for improvement and expansion could include:

1. Integration with Microsoft Windows and IOS Devices: The app could be expanded to integrate with IOS devices, allowing students and faculty to use the application on any mobile device.
2. Personalized trip planning: The app could be improved to provide more personalized trip planning features, such as suggesting alternative bus number based on the user's preferences and past usage patterns.
3. Real-time traffic data: Integrating real-time traffic data into the app could help users to better anticipate delays or congestion on their route, allowing them to adjust their travel plans accordingly.
4. Enhanced security features: The app could be improved to include enhanced security features, such as facial recognition or fingerprint authentication and OTP authentication, to ensure that only authorized users are able to access sensitive information.
5. User feedback and ratings: Incorporating a user feedback and rating system into the app could help to identify areas for improvement and allow users to provide feedback on their experiences with the system.

Overall, the development of an Android college bus tracking app has the potential for significant expansion and improvement, with opportunities to integrate with other systems, provide more personalized trip planning, and enhance security features.

CHAPTER 5

SOURCE CODE

```
import React, {useState} from 'react';
import {Text, View, TouchableOpacity} from 'react-native';
import startStyle from '../screenStyles/loginStyle';
import Background from '../src/Background';
import SmallBtn from '../src/components/SmallBtn';
import {yellow, cream} from '../src/constants';
import WhiteField from '../src/WhiteField';
import {signInWithEmailAndPassword} from 'firebase/auth';
import {auth} from '../firebase/firebase.config';
import {useNavigation} from '@react-navigation/native';
import ToggleSwitch from 'toggle-switch-react-native';

const Start = props => {
  const [email, setEmail] = useState(null);
  const [password, setPassword] = useState(null);
  const [toggle, setToggle] = useState(true);

  // Basically it depends on the toggle
  // if the toggle is student / staff then they are logged in
  // if the toggle is driver then the driver logged in

  const onToggleSwitch = () => {
    if (toggle === true) {
      setToggle(false);
    } else {
      setToggle(true);
    }
  };

  console.log(toggle);
  const navigation = useNavigation();

  // Authentication using google firebase

  const login = () => {
    signInWithEmailAndPassword(auth, email, password)
      .then(userCredential => {
        alert('User Login Successful')
        toggle? navigation.replace('UserHome'): navigation.replace('DriverHome')
      })
      .catch(error => {
        const errorMessage = error.message
        alert(errorMessage)
      });
  };

  // console.log(email);
```

```

// console.log(password);

// styles are done in loginStyle.js

return (
  <Background>
    <View style={startStyle.container}>
      <Text style={startStyle.text}>Bus Tracker</Text>

      <ToggleSwitch
        isOn={toggle}
        onColor={yellow}
        offColor={yellow}
        label={toggle ? 'Student / Staff' : 'Driver'}
        labelStyle={{color: 'white', fontWeight: '900', fontSize: 20}}
        size="small"
        onToggle={onToggleSwitch}
      />

    {
      // Login Screen for Student or a staff

      toggle ? (
        <View style={startStyle.formBg}>
          <WhiteField
            placeholder=" Email"
            onChangeText={text => setEmail(text)}
          />
          <WhiteField
            placeholder=" Password"
            secureTextEntry={true}
            onChangeText={text => setPassword(text)}
          />
          <SmallBtn
            bgColor={yellow}
            textColor={cream}
            btnLabel="Login"
            Press={() => navigation.navigate('UserHome')}
          />
        </View>
      ) : (
        // Login Screen for Student or a staff

        <View style={startStyle.formBg}>
          <WhiteField
            placeholder=" Mobile Number"
            onChangeText={text => setEmail(text)}
          />
          <WhiteField
            placeholder=" Password"
            secureTextEntry={true}

```



```

    onChangeText={text => setPassword(text)}
  />
  <SmallBtn
    bgColor={yellow}
    textColor={cream}
    btnLabel="Login"
    Press={() => navigation.navigate('DriverHome')}
  />
</View>
)
}
</View>
<TouchableOpacity
  onPress={() => props.navigation.navigate('RegisterScreen')}>
  <Text style={startStyle.buttonText}>Don't have an account?</Text>
</TouchableOpacity>
<TouchableOpacity
  onPress={() => props.navigation.navigate('RegisterScreen')}>
  <Text style={startStyle.passwordText}>Forgot your password?</Text>
</TouchableOpacity>
</Background>
);
};

export default Start;

{
  /* <Btn  textColor  =  {yellow}  btnLabel  =  "Register"  Press={  ()  =>
props.navigation.navigate("RegisterScreen")} /> */
} import React, { useState, useEffect } from "react";
import { StyleSheet, SafeAreaView, Text, TextInput, KeyboardAvoidingView } from "react-native";
import ChooseLocation from "../src/components/ChooseLocation";
import SearchBtn from "../src/components/SearchBtn";
import MapView from 'react-native-maps';
import { useNavigation } from '@react-navigation/native';
import SmallBtn from "../src/components/SmallBtn";

function UserHome() {

  // const fetchAddressCords = (lat, lng) => {
  //   // console.log('lat',lat)
  //   // console.log('long', lng)
  //   setState({...state, startingCords: {
  //     latitude: lat,
  //     longitude: lng
  //   }})
  // }

  //navigate to the map page

```

```

const onPressLocation = () => {
  navigation.navigate('MapPage')
}

// console.log('starting coords ==>', startingCords)

const navigation = useNavigation()

return (
  <KeyboardAvoidingView style={styles.container} enabled={false} behavior="height">
    <SafeAreaView style={styles.navbar}>
      <SafeAreaView style={styles.top}>
        <Text style={styles.welcome}>Welcome, Kevin Franklin</Text>
      </SafeAreaView>
      <SafeAreaView style={styles.autocomplete}>
        <ChooseLocation />
      </SafeAreaView>
      <SafeAreaView style={styles.bottom}>
        <TextInput style={styles.input} placeholder="Sathyabama, Chennai"
placeholderTextColor={'grey'} editable={false}/>
        <SearchBtn textColor={'black'} btnLabel={'Search'} Press={onPressLocation}/>
        <SmallBtn textColor={'black'} btnLabel={'Search'} Press={()=>
navigation.navigate('DriverHome')}/>
      </SafeAreaView>
    </SafeAreaView>
    <SafeAreaView style={styles.body}>
      <Text style={styles.subHeading}>Alert Box</Text>
    </SafeAreaView style={styles.alert}></SafeAreaView>
      <Text style={styles.subHeading}>Buses Near You</Text>
    <SafeAreaView style={styles.nearby}>
      <MapView style={StyleSheet.absoluteFill}
initialRegion={{
  latitude: 13.0827,
  longitude: 80.2707,
  latitudeDelta: 0.0922,
  longitudeDelta: 0.0421,
}}
/>
    </SafeAreaView>
  </SafeAreaView>
</KeyboardAvoidingView>
);
}

export default UserHome

const styles = StyleSheet.create({
  container: {
    height: '100%',
  }

```



```

backgroundColor: '#cfedb8',
alignItems: 'center'
},
subHeading: {
  top: 20,
  color: 'black',
  fontWeight: 'bold',
  fontSize: 20
},
navbar: {
  top: 10,
  height: '30%',
  width: '90%',
  backgroundColor: '#f6b756',
  borderRadius: 20,
},
bottom: {
  width: '100%',
  alignItems: 'center',
},
autocomplete: {
  position: 'relative',
  width: '100%',
  alignItems: 'center',
  zIndex: 1
},
welcome: {
  fontSize: 22,
  color: 'white',
  fontWeight: 'bold',
  borderColor: 'black',
  width: '100%',
  height: 60,
  backgroundColor: '#4a686a',
  textAlign: 'left',
  paddingLeft: 15,
  paddingTop: 15,
  borderTopLeftRadius: 20,
  borderTopRightRadius: 20
},
alert: {
  top: 30,
  height: '30%',
  width: '90%',
  backgroundColor: '#f6f8e1',
  borderRadius: 20,
  marginBottom: 30
},
nearby: {
  top: 40,
  height: '40%',

```

```

width: '90%',
backgroundColor: 'blue',
borderRadius: 20,
},
body: {
  position: 'relative',
  zIndex: -1,
  top: 30,
  height: '65%',
  width: '88%',
  backgroundColor: '#fbbc57',
  alignItems: 'center',
  borderRadius: 20,
},
input: {
  width: '80%',
  top: 15,
  backgroundColor: '#f6f8e1',
  borderRadius: 25,
  height: '24%',
  fontWeight: "bold",
  fontSize: 18,
  paddingLeft: 20,
  paddingBottom: 8,
  marginBottom: 22,
},
search: {
  top: 20,
},
})

```

```

import { StyleSheet, Text, SafeAreaView, TextInput } from 'react-native'
import React, { useState } from 'react'
import SmallBtn from '../src/components/SmallBtn'
//import { collection, addDoc } from 'firebase/firestore'
//import { auth, db } from '../firebase/firebase.config'
import tw from 'tailwind-react-native-classnames'

```

```

const DriverHome = () => {

  const [message, setMessage] = useState(null)

  // const saveData = async () => {
  //   console.log('running saveData')
  //   const docRef = await addDoc(collection(db, 'Message'), {
  //     Message: message
  //   })
  //   console.log("Document with id : ", docRef.id)
  // }

  return (

```



```

<SafeAreaView style={[tw`flex-auto`,styles.container]}>
  <SafeAreaView style={[tw`flex-auto`,styles.navbar]}>
    <SafeAreaView style={[tw`flex-auto`,styles.top]}>
      <Text style={[tw`flex-auto`,styles.welcome]}>Welcome, Kevin Franklin</Text>
    </SafeAreaView>
  </SafeAreaView>
  <Text style={[tw`flex-auto`,styles.subHeading]}>Your Details:</Text>
  </SafeAreaView>
  <SafeAreaView style={[tw`flex-auto`,styles.bottom]}>
    <Text style={[tw`flex-auto`,styles.details]}>Name: Lakshu Vasan</Text>
    <Text style={[tw`flex-auto`,styles.details]}>Mobile: 825718452</Text>
    <Text style={[tw`flex-auto`,styles.details]}>Bus Route No: 47</Text>
    <Text style={[tw`flex-auto`,styles.details]}>Age: 20</Text>
  </SafeAreaView>
</SafeAreaView>
<SafeAreaView style={[tw`flex-auto`,styles.message]}>
  <Text style={[tw`flex-auto`,styles.subHeading]}>Send Message :</Text>
  <SafeAreaView style={[tw`flex-auto`,styles.alert]}>
    <TextInput style={styles.input}
      multiline={true}
      placeholder="Start typing..."
      //onChangeText={(text)=> setMessage(text)}
    />
  </SafeAreaView>
  <SmallBtn bgColor={'white'} textColor={'black'} btnLabel={'Send'} />
  <Text style={styles.details}>
    * While Sending Message, Please Specify the Bus Number.
  </Text>
  <Text style={styles.details}>
    * If you face any issue with the app, Please contact the admin.
  </Text>
</SafeAreaView>
</SafeAreaView>
)
}

```

```
export default DriverHome
```

```

const styles = StyleSheet.create({
  container: {
    height: '100%',
    backgroundColor: '#fcffd',
    alignItems: 'center'
  },
  navbar: {
    top: 10,
    height: '32%',
    width: '90%',
    backgroundColor: '#f6b756',
    borderRadius: 20,
  },

```

```

welcome: {
  fontSize: 25,
  color: 'white',
  fontWeight: 'bold',
  width: '100%',
  height: 60,
  backgroundColor: '#4a686a',
  textAlign: 'left',
  paddingLeft: 15,
  paddingTop: 15,
  borderTopLeftRadius: 20,
  borderTopRightRadius: 20
},
subHeading: {
  fontSize: 22,
  color: '#4a686a',
  fontWeight: '700',
  width: '100%',
  height: 50,
  textAlign: 'left',
  paddingLeft: 15,
  paddingTop: 15,
},
details: {
  fontSize: 20,
  color: 'white',
  width: '100%',
  textAlign: 'left',
  paddingLeft: 15,
  paddingBottom: 6
},
message: {
  alignItems: 'center',
  top: 20,
  height: '62%',
  width: '90%',
  backgroundColor: '#f6b756',
  borderRadius: 20,
},
alert: {
  top: 10,
  height: '50%',
  width: '90%',
  backgroundColor: '#f6f8e1',
  borderRadius: 20,
  marginBottom: 30
},
input: {
  fontSize: 20,
  padding: 20
}

```



```

})
import * as React from "react";
import { useState } from "react";
import { StyleSheet, Text, View, TextInput, TouchableOpacity } from "react-native";
import registerStyle from "../../screenStyles/registerStyle";
import Background from '../../src/Background';
import { yellow } from "../../src/constants";
import ToggleSwitch from "toggle-switch-react-native";
import studentRegisterStyle from '../../screenStyles/studentRegisterStyle';
import driverRegisterStyle from "../../screenStyles/driverRegisterStyle";
import WhiteField from "../../src/WhiteField";
import SmallBtn from "../../src/components/SmallBtn";
import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-view'

```

```

const RegisterScreen = (props) => {

```

```

  const [fullName, setFullName] = useState("")
  const [email, setEmail] = useState(null)
  const [password, setPassword] = useState(null)
  const [number, setNumber] = useState("")
  const [toggle, setToggle] = useState(true)

```

```

  // Basically it depends on the toggle
  // if the toggle is student / staff then their details are collected
  // if the toggle is driver then the driver details are collected

```

```

  const onToggleSwitch = () => {
    if (toggle === true) {
      setToggle(false)
    } else {
      setToggle(true)
    }
  };

```

```

  if (toggle === true) {
    console.log('maame')
  }

```

```

  return (
    <Background>
      <KeyboardAwareScrollView>
        <View style={registerStyle.container}>
          <ToggleSwitch
            isOn={toggle}
            onColor={yellow}
            offColor={yellow}
            label={toggle ? 'Driver ?' : 'Student / Staff ?'}
            labelStyle={{ color: "white", fontWeight: "900", fontSize: 20 }}
            size="small"
            onToggle={onToggleSwitch}

```

```

/>

{
// STUDENT REGISTRATION

toggle ? (
<View style={studentRegisterStyle.container}>
  <Text style={studentRegisterStyle.title}>
    Student / Staff Register
  </Text>
  <View style={studentRegisterStyle.formBg}>
    <Text style={studentRegisterStyle.formTitle}>
      Create an Account
    </Text>
    <WhiteField
      placeholder=" Full Name"
      onChangeText={(text) => setFullName(text)}
    />
    <WhiteField
      placeholder=" Email"
      keyboardType={"email-address"}
      onChangeText={(text) => setEmail(text)}
    />
    <WhiteField
      placeholder=" Password"
      secureTextEntry={true}
      onChangeText={(text) => setPassword(text)}
    />
    <WhiteField
      placeholder=" Re-enter Password"
      secureTextEntry={true}
    />
    <View style={studentRegisterStyle.otp}>
      <TextInput
        style={{
          borderRadius: 100,
          color: "black",
          paddingHorizontal: 10,
          width: "40%",
          height: 50,
          backgroundColor: "white",
          fontSize: 19,
          marginVertical: 15,
        }}
        placeholderTextColor="grey"
        placeholder=" Enter OTP"
      ></TextInput>
      <TouchableOpacity>
        <Text style={studentRegisterStyle.otpText}>Resend OTP</Text>
      </TouchableOpacity>
    </View>

```



```

<SmallBtn
  bgColor={yellow}
  textColor="white"
  btnLabel={"Register"}
  Press={() => {
    signup();
    create();
  }}
/>
</View>
</View>

```

```
);
```

```
// DRIVER REGISTRATION
```

```

(
  <View style={driverRegisterStyle.container}>
    <Text style={driverRegisterStyle.title}>Driver Register</Text>
    <View style={driverRegisterStyle.formBg}>
      <Text style={driverRegisterStyle.formTitle}>
        Create an Account
      </Text>
      <WhiteField
        placeholder=" Full Name"
        onChangeText={(text) => setFullName(text)}
      />
      <WhiteField
        placeholder=" Mobile no"
        keyboardType={"number-pad"}
        onChangeText={(num) => setNumber(num)}
      />
      <WhiteField
        placeholder=" Email"
        keyboardType={"email-address"}
        onChangeText={(text) => setEmail(text)}
      />
      <WhiteField
        placeholder=" Password"
        secureTextEntry={true}
        onChangeText={(text) => setPassword(text)}
      />
      <WhiteField
        placeholder=" Re-enter Password"
        secureTextEntry={true}
      />
    <View style={driverRegisterStyle.otp}>
      <TextInput
        style={{
          borderRadius: 100,
          color: "black",

```

```

paddingHorizontal: 10,
width: "40%",
height: 50,
backgroundColor: "white",
fontSize: 19,
marginVertical: 15,
}}
placeholderTextColor="grey"
placeholder=" Enter OTP"
></TextInput>
<TouchableOpacity>
  <Text style={driverRegisterStyle.otpText}>Resend OTP</Text>
</TouchableOpacity>
</View>
<SmallBtn
  bgColor={yellow}
  textColor="white"
  btnLabel={"Register"}
  Press={() => {
    signup();
    create();
  }}
/>
</View>
</View>
))
</View>
</KeyboardAwareScrollView>
</Background>

);
};

export default RegisterScreen;

const styles = StyleSheet.create({});
import {StyleSheet, Text, View, TouchableOpacity, Image} from 'react-native';
import React, {useEffect, useRef, useState} from 'react';
import SmallBtn from '../src/components/SmallBtn';
import {useNavigation} from '@react-navigation/native';
import BusBtn from '../src/components/busNumber';
import MapView, {Marker, AnimatedRegion, MarkerAnimated} from 'react-native-maps';
import MapViewDirections from 'react-native-maps-directions';
import {GOOGLE_MAP_KEY} from '../src/constants/googleMapKey';
import imagePath from '../src/constants/imagePath';
import {useSelector} from 'react-redux';
import {selectOrigin} from '../slices/navSlice';
//import database from '@react-native-firebase/database';
import {
  locationPermission,

```



```

getCurrentLocation,
} from '../src/helper/helperFunction';
import Loader from '../src/components/Loader';

const MapPage = () => {
  //create reference for map and marker
  const mapRef = useRef();
  const markerRef = useRef();

  const origin = useSelector(selectOrigin);
  const navigation = useNavigation();

  // fetch lat and long from user home page with the help of slice
  // Use Animated region for smooth tracking

  const [state, setState] = useState({
    curLoc: {
      latitude: origin.location.lat,
      longitude: origin.location.lng,
      latitudeDelta: 0.005,
      longitudeDelta: 0.0,
    },
    isLoading: false,
    coordinate: new AnimatedRegion({
      latitude: origin.location.lat,
      longitude: origin.location.lng,
      latitudeDelta: 0.005,
      longitudeDelta: 0.0,
    }),
    time: 0,
    distance: 0,
    heading: 0
  });

  const {curLoc, isLoading, coordinate, time, distance, heading} = state;
  const updateState = (data) => setState((state) => ({ ...state, ...data }));

  // create smooth animated tracking based on the OS

  const animate = (latitude, longitude) => {
    const newCoordinate = {latitude, longitude};
    if (Platform.OS === 'android') {
      if (markerRef.current) {
        markerRef.current.animateMarkerToCoordinate(newCoordinate, 7000);
      }
    } else {
      coordinate.timing(newCoordinate).start();
    }
  };
};

```

// Note !! the destination is always fixed, It is Sathyabama.

```

const [dropCords, setDropCords] = useState({
  latitude: origin.location.lat,
  longitude: origin.location.lng,
  latitudeDelta: 0.0922,
  longitudeDelta: 0.0421,
})
// console.log('drop', dropCords)
// console.log('current', curLoc)
if (dropCords.latitude && dropCords.longitude === curLoc.latitude && curLoc.longitude) {
  console.log('they crossed')
}
// get location permission from user
const getLiveLocation = async () => {
  const locPermissionDenied = await locationPermission();

  // if location granted

  if (locPermissionDenied) {
    const {latitude, longitude} = await getCurrentLocation();
    animate(latitude, longitude);
    updateState({
      heading: heading,
      curLoc: {
        latitude,
        longitude,
        latitudeDelta: 0.0922,
        longitudeDelta: 0.0421,
      },
      coordinate: new AnimatedRegion({
        latitude: latitude,
        longitude: longitude,
        latitudeDelta: 0.005,
        longitudeDelta: 0.0,
      })),
    });
  }
};

// track live location every 6 seconds

useEffect(() => {
  const interval = setInterval(() => {
    getLiveLocation();
  }, 6000);
  return () => clearInterval(interval);
})

// makes the moving marker to display in the center

const onCenter = () => {
  mapRef.current.animateToRegion({

```



```

latitude: curLoc.latitude,
longitude: curLoc.longitude,
latitudeDelta: 0.005,
longitudeDelta: 0.0,
}))
}

const fetchTime = (d, t) => {
  updateState({
    distance: d,
    time: t,
  })
}
console.log(time)
const [arrived, setArrived] = useState(false)

const isArrived = () => {
  if(time < 1) {
    setArrived(true)
  } else {
    setArrived(false)
  }
}

return (
  <View>
    <View style={styles.navbar}>
      <SmallBtn
        bgColor={'#4a686a'}
        btnLabel={'Back'}
        textColor={'white'}
        Press={() => navigation.navigate('UserHome')}
      />
    </View>
    <View style={styles.container}>
      <View style={styles.busNumber}>
        <View>
          <Text style={styles.subHeading}>Bus Route Number</Text>
        </View>
        <View style={styles.busRoute}>
          <BusBtn btnLabel={'67'} />
          <BusBtn btnLabel={'67'} />
          <BusBtn btnLabel={'67'} />
          <BusBtn btnLabel={'67'} />
          {/* <BusBtn btnLabel={'67'} />
            <BusBtn btnLabel={'67'} />
            <BusBtn btnLabel={'67'} />
            <BusBtn btnLabel={'67'} /> */}
        </View>
      </View>
    </View>
  )

```

B.Screen Shots

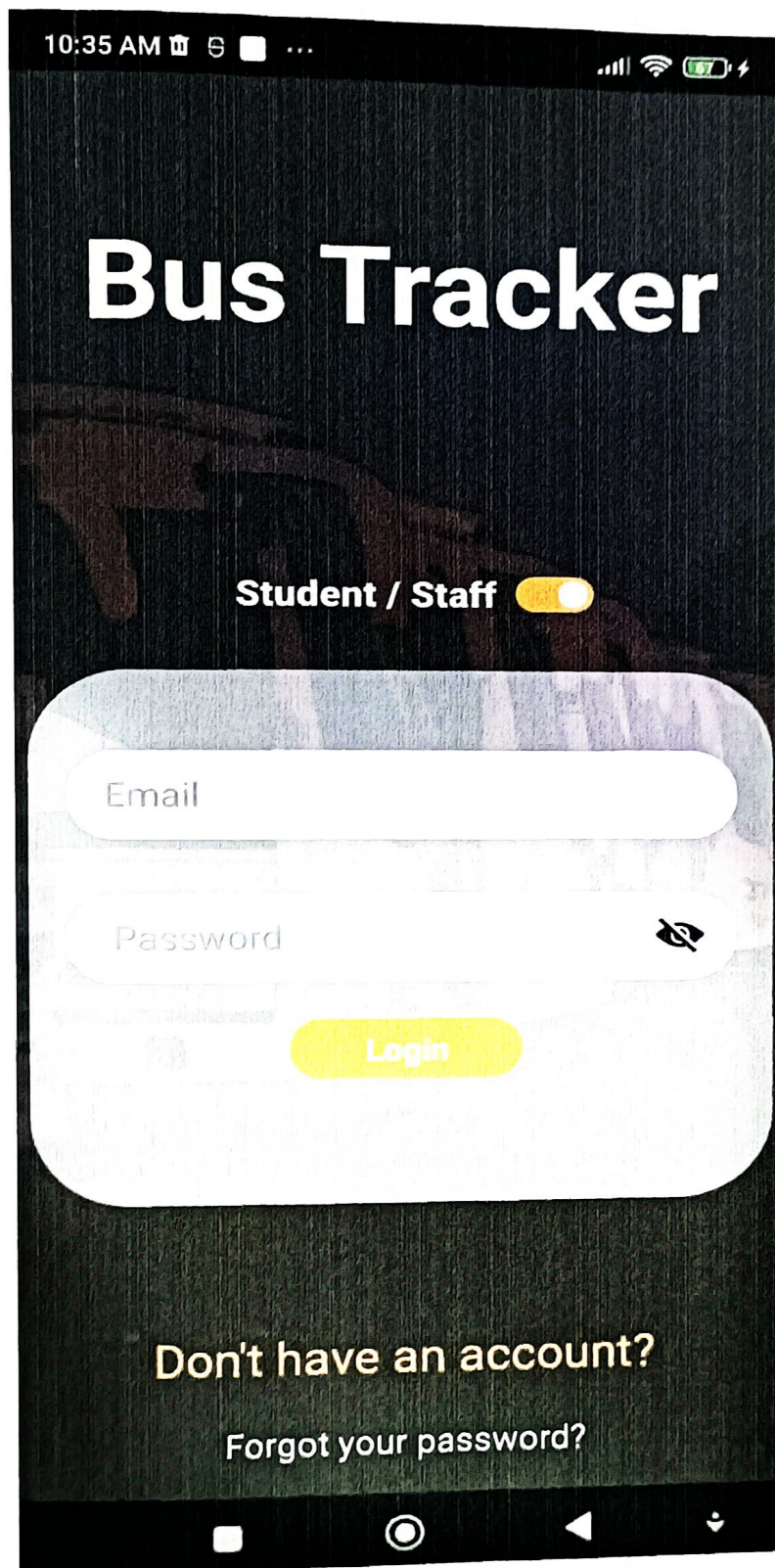


Fig: 1 Login Page

10:35 AM    ...

Driver ? ☒

Student / Staff Register

Create an Account

Full Name

Email

Password 

Re-enter Password

Register

Fig: 2 User Registration Page



Fig: 3 Driver Home Page

10:39 AM



Welcome, Kevin Franklin

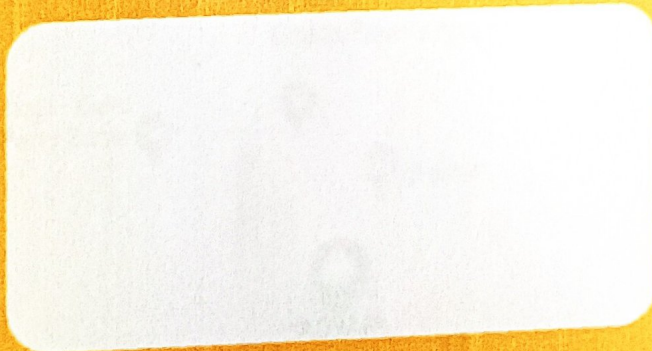


Where are you ?

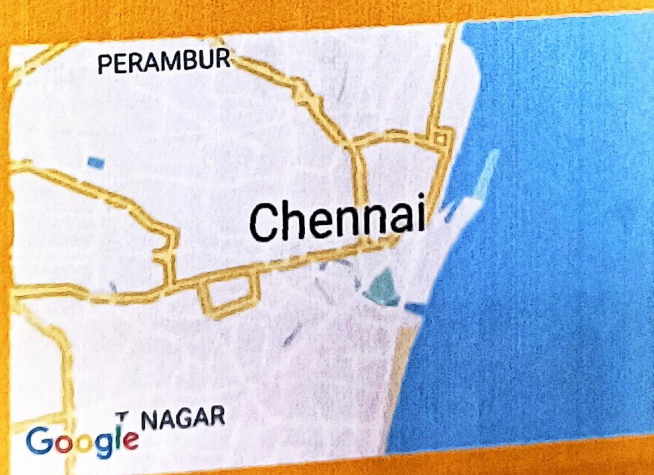
Sathyabama, Chennai

Search

Alert Box



Buses Near You



User Home Page

10:39 AM



Back

Bus Route Number

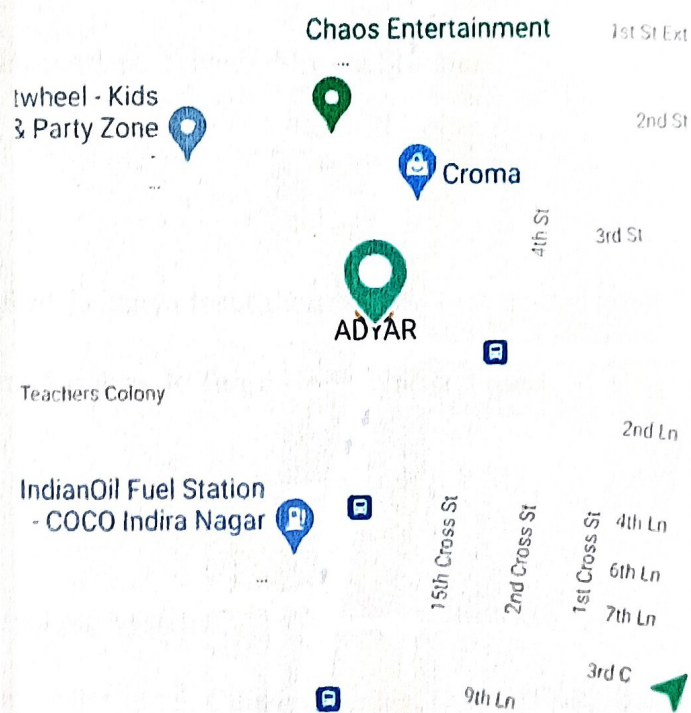
67

67

67

67

Bus Arrived ?



Estimated Time: 0 mins

Estimated Distance: 0 Kms

Map Page

REFERENCES

- [1] BUS TRACKING AND STUDENT MONITORING. Author Name: RC Jisha,, Mathews P Mathews, Sidharth P Kini, Vineeth Kumar, UV Harisankar, M Shilpa. Year of Publish: 2018
- [2] An Intelligent and Secured Tracking System for Monitoring School Bus
Author Name : Asif Ahmed, M MRayhanParvez , MdHridoyHasan, FernezNarinNur, NazmunNessa Moon, AsifKarim, Sami Azam, BharanidaranShanmugam, MirjamJonkman Year of Publish: 2018
- [3] Application Based Bus Tracking System
Author Name: Subham Jain, AdarshTrivedi, Shweta Sharma
Year of Publish: 2017
- [4] Bus Tracking System for Chaitanya Institution
Author Name: P Siva Rama Krishna, R Tamil Kodi, P Siva Prasad
Year of Publish: 2019
- [5] A Smart School Bus Tracking System
Author Name: IkerKorkmaz, Alp Camci, CihangirCengiz, DoganDirik, EmreCekci, Faith MemhmetAkbab
Year of Publish: 2019
- [6] A Real-time School Bus Tracking and Monitoring System
Author Name: Khalid Ammar, Mohammad Jalmoud, AbdiulrasullBoushehri, Khalid Fakhro
Year of Publish: 2019