|  |  |  |
| --- | --- | --- |
| |  | | --- | |  | | **PES University, Bengaluru** |
|  | (Established under Karnataka Act 16 of 2013) |
| **Department of Computer Science & Engineering** | |
| **Session: Jan - May 2022** | |

**UE19CS353 – Object-Orie Analysis and Design with Java**

**Theory ISA (Mini Project)**

Report on

**COLLEGE BUS TRACKING SYSTEM**



**By:**

**KIRAN B S – PES2UG19CS186**

**KOUSHIK P L – PES2UG19CS193**

**KUNTAL GORAI – PES2UG19CS198**

**6th Semester, C Section**

1. **Project Description**

Github link:-[**https://github.com/bskiranchandru/Bus-Tracking-System**](https://github.com/bskiranchandru/Bus-Tracking-System)

## College Bus Tracking System

The college bus tracking system is a system that helps the college students to keep track of their bus location so that the students donated get delayed or arrive too early at their bus stop.

To overcome this increased waiting time and uncertainty of its arrival, we have come up with this system for our project.

Our proposed system is to find the location of the bus and let the users know about the location so that they can manage their time effectively and reach their stop just before the bus arrives or plan for alternate means of transport if they miss the bus or if they are running late.

**Requirements/Features:**

Our system will mainly consist of three entities:

### 1. User

It is a student who goes to their college via college bus. They have a source destination from where the driver of the respective bus picks him up and drops him off to and from college. The user can only keep track of its respective bus.

### 2. Admin

He has some admin privileges of adding a college student to his respective bus, adding a driver to his respective bus, tracking bus details, viewing route details, and adding a new bus route.

### 3. Driver

He is the driver of the college bus. He can view his bus d some tracking detailed ls, keeptrackacks of students for that bus.

A detailed description of features for each entity:

**Admin:**

### a. Login

A simple login page asking the admin to enter his credentials i.e., username and password. If the user has entered the correct credentials, he will be tracked to other features that an admin can have. In case the user enters the wrong credentials, then an error message will be displayed telling the admin that he has entered the wrong credentials.

**b. Registration activity: -** Admetails, addin creates the user account with help of the details provided by the students, once the account has been successfully created, the login credentials will be mailed to the respective student’s mail id. In the same way, the admin also creates the driver account in which he adds all the driver details.

### c. Add Bus route

This functionality allows the admin to add a new bus route as and when it is required. The admin will be asked to enter various details about his new bus route like bus stops, driver, estimated time at each bus stop, location, and bus number. Once the admin has entered all the details correctly then the new bus route will be added successfully.

### d. Track bus details

This functionality helps in the track of the bus details that will be occasionally updated by the driver of the respective bus. He will verify all the details thave has been updated and make sure that all details are correct and can be viewed by the user. He can also add some necessary changes to the backend database.

### e. Logout

This functionality is to log out of the website once the admin has performed the necessary changes on the website. Once the user logs out, he won’t have admin functionalities. He has to log in again if we want to perform the functionalities described above.

**User:**

* **User Login:** The user is allowed to login into the system with the user id and password sent to his/her email Id.

* **User profile:**- This module has the details of the user.

* **Track:** The user can locate and track the bus movements.

* **Pick up status:** Through this module students can update regarding the change in the pickup point or if they couldn’t take up the bus on that day.

* **Feedback**: The user can give feedback regarding the service or any complaints or suggestions.

* **Change Password**: The user can change the password as the password is initially given by the system which is difficult to memorize.

* **Forgot Password**: The user can provide the email address and reset the password if the user forgets it.

* **Logout**: after the student arrived at his college/home he need to log out from the application for security purpose.

**Driver:**

1. **Login:** The driver has to enter the login details/credentials like user id and password to enter the application given by the college authority if enter the wrong details the error message needs to display on the screen.

1. **App Live**: The Driver has to login into the application and with the help of the mobile GPS the driver, its location is updated at every interval.

1. **View my route details:** This module is used to help the driver by providing the student’s residential address or pick-up address route details.

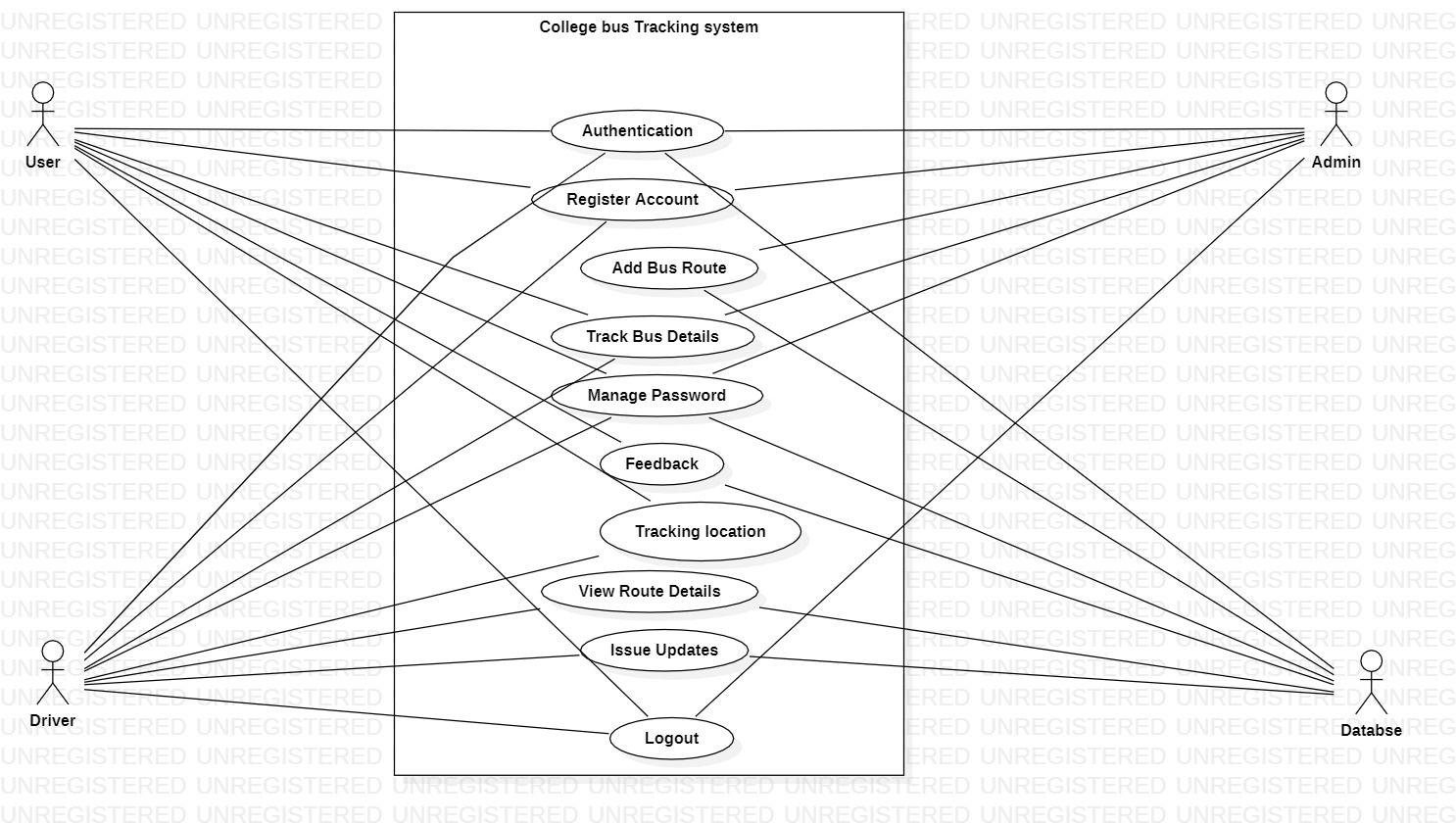
1. **Pick up notification:-**  this module helps the driver know about the student’s status(regarding the change in the pickup point or the student couldn’t catch the bus ), so that driver can make the decision accordingly.

1. **Issues updates: The driver can notify the admin or college authority regarding the breakdown of the bus or** the improper function of hardware components.

1. **Forgot password/reset password:** If the driver enters the wrong password or if forgot the password then needs to provide the option to reset the password.

1. **Logout: -** After his work driver needs to log out from the application to maintenance of the application and security point.
2. **Analysis and Design Models**

**Use Case Diagram:**

****

The main actors are the user(student), driver, Admin, and the backend database to store necessary details. First, the user and driver need to perform a sign-up or sign in to the app. Next, the user can track his location through the app and get to know how far the bus from his location is and what is the estimated time. The driver can any updates based on the location or new routes in the app. Admin performs the functionalities of adding bus routes, verifying users and drivers who have registered, and performing the software updates.

**Diagram

Description automatically generatedClass Diagram:**

The main classes are the student, driver, bus, route, and admin.

Student class can perform the functionalities such as login, view profile, view bus details, track bus, and logout.

Admin class can perform the functionalities such as login, register student, register driver, add us the route, get bus details, and log out.

Driver class can perform the functionalities such as login, view bus details, add tracking details, logout, view profile, and adan d issue.

Route class performs the functionalities such as show route, start route, and destination route.

Bus class performs the functionalities such as getting pickup points and adding pickup points.

**Activity Diagram:**

Diagram

Description automatically generated

Diagram, engineering drawing

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

**State Diagram:**

Diagram

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

Diagram

Description automatically generated

1. **Tools and Frameworks Used**
   1. java14.02
   2. Android Studio
   3. Android framework
   4. Android plugin and Gradle version
   5. Google map API’s
   6. Firebase
   7. Android SDK-(can work with 4-7version)
2. **Design Principles and Design Patterns Applied**

**Design Principle: Single Responsibility principle**

The given uses the single responsibility principle these can be seen by the various functionalities class that perform only a single responsibility like gpstracker.java to perform tracking using GPS, signup.java to add username and password to firebase database, tracking.java to perform tracking of the app using the GPS location.

**Design pattern: - facade pattern**

The given app uses a façade design pattern as the mainactivity.java module act as a facade class and the gpstracker.java, signin.java, signup.java, tracking.java,

profileactivity.java act as the subsystem (sub functionalities)

so the client interacts with the main activity which indirectly calls the sub functionalities and performs those functionalities.

**Architecture: MVC(Model-View-Controller Architecture)**

Here in our app, the view is the UI of our app, the mainactivity.java is the controller, and the GPS tracker, login, signup, tracker, and alarm are the model.

1. **Application Screenshots (3-4 important pages)**

Text

Description automatically generated with low confidenceGraphical user interface, application

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

**1a 1b 1c**

Timeline

Description automatically generated with medium confidenceA picture containing text, screenshot, businesscard

Description automatically generatedGraphical user interface, application

Description automatically generated

**2a 2b 2c**

A screenshot of a phone

Description automatically generated with medium confidenceA picture containing background pattern

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

**3a 3b 3c**

**Description of Screenshots:**

**Image 1a:**

This is the Splash screen of our app. It provides sign-in and sign-up options for users.

**Image 1b:**

This is the signup page of our app, which takes all the details from the user and stores them in the database.

**Image 1c:**

This is the sign-in page, where the user enters login credentials to log in to the app. The entered login credentials are matched with the details stored in the database.

**Image 2a:**

This page is showing the option of saving login credentials for future use.

**Image 2b:**

This is the home page of our app, from where users can fetch the location of the bus and also redirect to the map page.

**Image 2c:**

This is the Dashboard of our app. It contains various activities users can perform.

**Image 3a:**

This page notifies the user that GPS is not enabled, and asks him to enable it.

**Image 3b:**

This page is to display important messages sent by the driver in case of a change in route/bus timings etc. Users can also send messages to the drivers to notify them about a change in the pick-up point or also if he/she is not participating in a ride on that particular day.

**Image 3c:**

This page provides users with an emergency call option, which can be used when there are some emergencies.

**Details stored in the database:**

Graphical user interface, text, application

Description automatically generated

All the user details are stored in the database, we make use of firebase as our database to store the details.

1. **Team member contributions**

|  |  |
| --- | --- |
| **NAME** | **WORK** |
| **KUNTAL GORAI** | **Admin Module, Design, and implementation.** |
| **KIRAN BS** | **Driver modules, design, and implementation.** |
| **Koushik P L** | **User modules, design, and implementation** |