**Vehicle Registration for CBIT Parking**

**Submitted to COSC**

**Done by**

G. Pavan Praneeth – CSE - 3rd Year - App Development

Sowmya Bommu – CSE – 3rd Year - App Development

Mohammed Abdul Mateen – IT – 2nd Year – Database

Mohammed Adnan – CSE – 1st Year – API

Rohit Kumar Bandaru – CSE – 2nd Year - Web Development

Kalyan Reddy Gone – IT – 2nd Year – Web Development

Ganga. Bhanu Teja – CSE – 1st Year – Web Development

Koppala Praneetha – CSE – 1st Year – Web Development

**Mentored by**

Preethivardhan Anusri Ega

**Problem Statement**

The project will consist of 2 main components, the mobile application for students to sign up, login to the app, add vehicles(up to 2 per person) and apply for parking sticker, delete vehicles and apply for a new CBIT parking sticker with fine.

The second component is the admin functionalities which will be in a web application. The admin will be able to login, receive all the applications for vehicle registration, verify and approve, receive fine payments physically, update payment status in application and issue parking stickers and dismiss any vehicles registration (in case of misbehavior). It will also consist of a statistics page where registration and payment statistics will be displayed based on the date given.

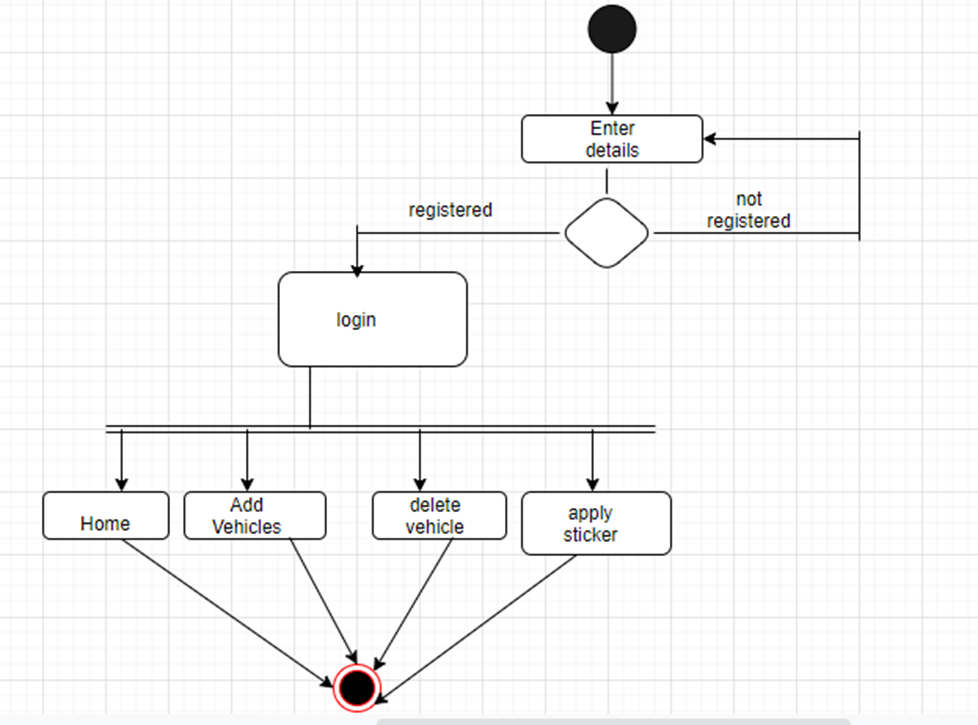
**Design**

**Use case diagram:**

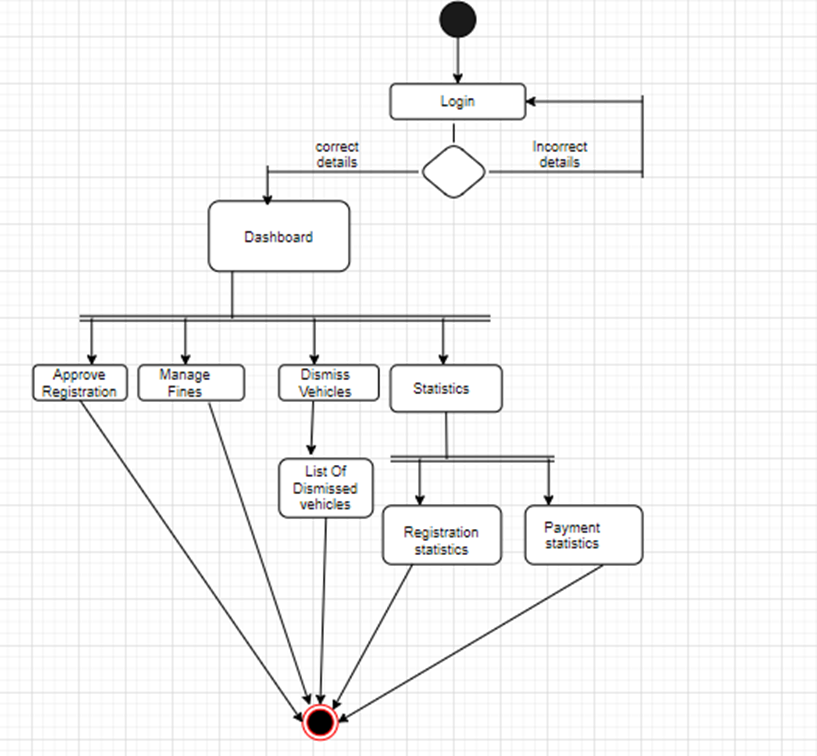
A close up of a map

Description automatically generated

**Activity Diagrams:**

****

Mobile app

****

Web app

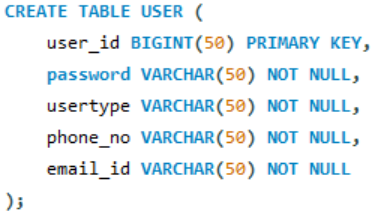
**DATABASE**

Database was created in MySQL Workbench. 3 tables have been created.

**User table**

This table will be holding all the credentials of the user’s and admins. Every user id will be unique.

This data comes in handy when an admin/user tries to login, if that credentials are authentic then they will be redirected to dashboard.



**Registration table**

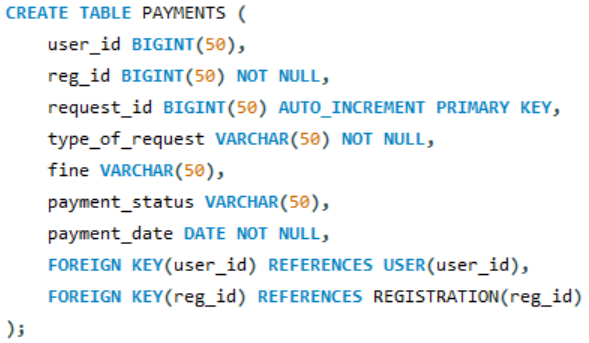
Holds the data of all the registered vehicles. Every registration id is unique for easy search. Necessary information about the person and vehicle is stored. Registration status can be dismissed, deleted, approved or pending. Registration date is stored for statistics.

A screenshot of a cell phone

Description automatically generated

**Payments table**

Holds the data about the payments made physically. There are 3 criteria for payments – adding back deleted, dismissed vehicle and for new sticker. Payment status gets updated to ‘paid’ once done physically. Payment date for statistics.



**API**

Python and Flask were used to develop the REST API’s.

The Rest API has been deployed to HEROKU.

Link: <https://cosc-vehicle.herokuapp.com/>

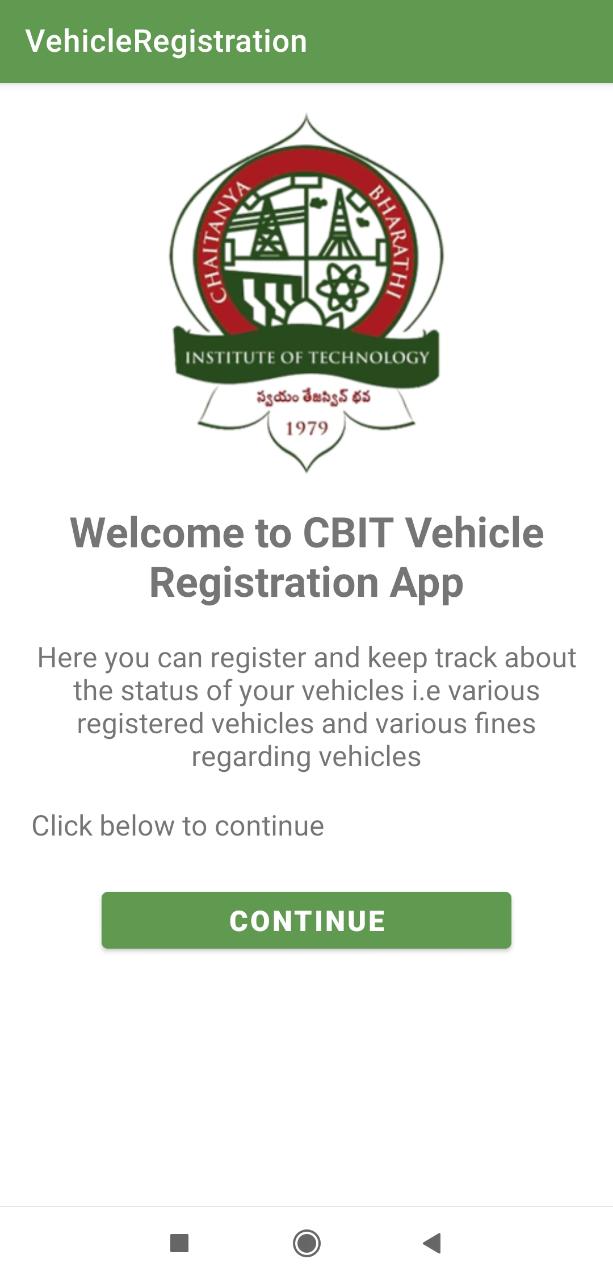
The below endpoints to access the tables in the database.

* */login*: takes a JSON object with 'user\_id', 'password' and 'usertype' gives back JWT if exists in Users table. The JWT shall be used to access all the end points. For all the endpoints an Authorization Header should be included with value 'Bearer '.
* */register* (POST): To post into Users table.
* */vechreg* (POST): To add a vehicle and its details in the Registration table. The reg\_status is by default pending.
* */vechreg* (GET): To get the details of any vehicle based on Registration ID (reg\_id) provided.
* */pending* (GET): fetches details of all the registrations whose Registration Status (reg\_status) is pending.
* */pending* (PUT): updates the reg\_status as ***approved*** based on the Registration ID (reg\_id) provided.
* */dismiss* (PUT): To update the reg\_status as ***dismissed*** for the Registration ID (reg\_id) provided. A vehicle will be dismissed in case of Misbehavior.
* */dismiss* (GET): To fetch List of vehicles that have been dismissed.
* */delete* (PUT): For Updating the Registration Status as ***deleted***.
* */regstats* (GET): To get data of registration table for the date provided by the user.
* */fine* (POST): To post a fine request and get back request\_id.
* */fine* (PUT): To update the Payment Status as ***paid*** after receiving the Payment manually.
* */fine* (GET): To get the details of users whose payment\_status is ***pending***.
* */paymentstats* (GET) : To get the details of fine payments made on a particular date. The date will be provided by the user.
* */regsuser* (GET): To get the details of the vehicles registered based on the user\_id provided.

**Implementation**

**Mobile Application (User)**

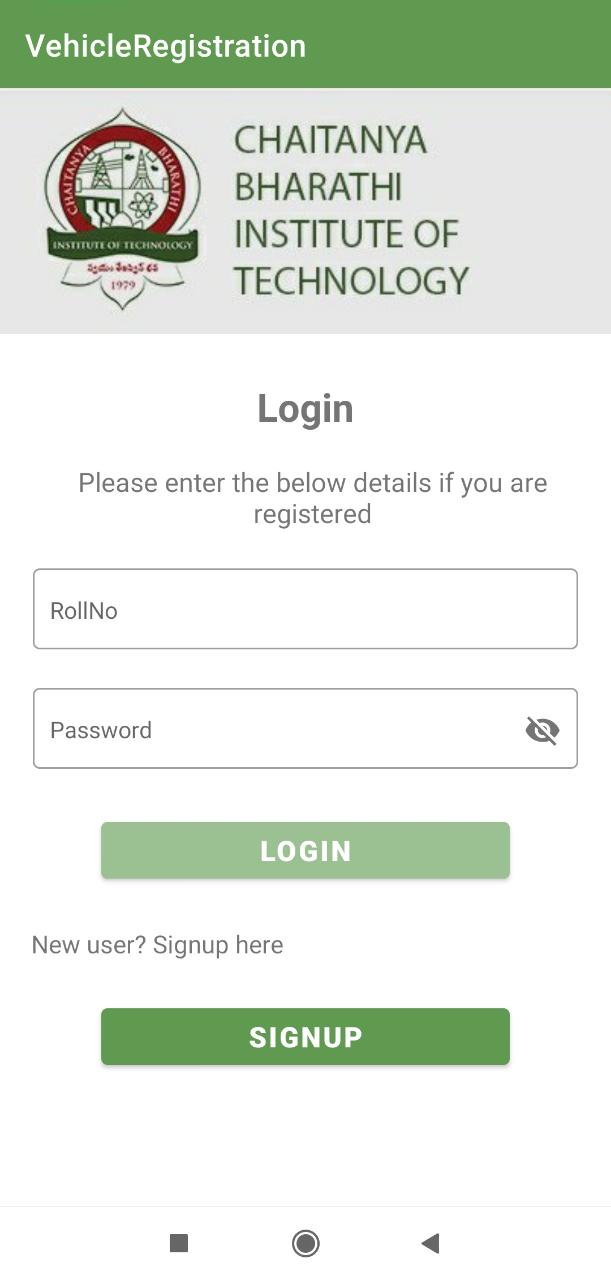
The mobile application was developed in java with the help of Android Studio API calls were made using the Http requests and response module. Json parsing is used for data transfer in the request and response modules.



Welcome page

**Login**

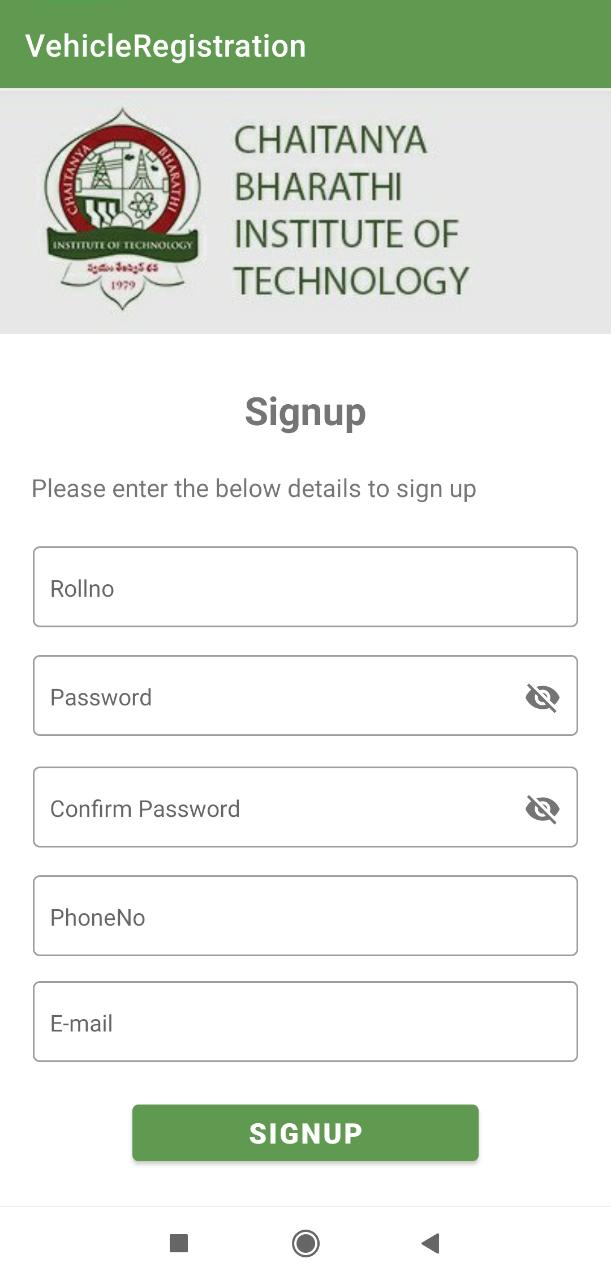
The login page has two fields, user roll number and password. If the user enters the wrong details, toast message “Please check your details” will be displayed. If the correct details are entered, User is redirected to the home page.



Login page

**Sign up**

The signup page has fields roll number, password, phone number, E-mail. The student can register by giving these details. POST API is used to insert these details to the database. On Successful registration, user is redirected to login page. If user roll number already exists it shows toast message “User already exists”.



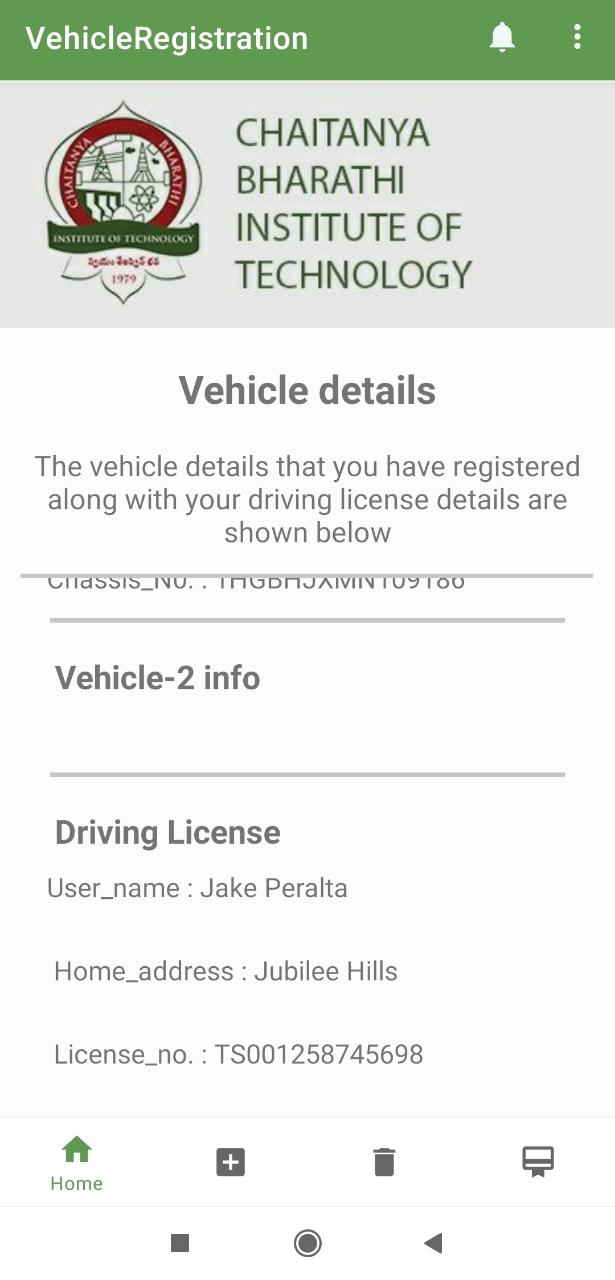
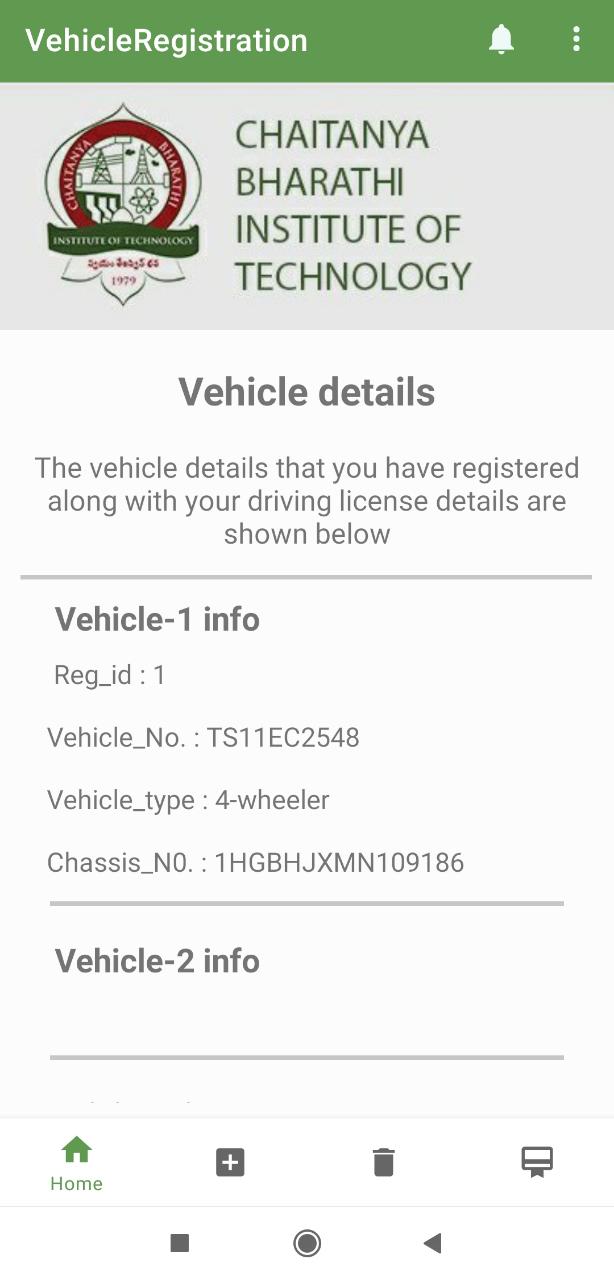
Sign Up

**Bottom Navigation Bar:**

In the bottom nav bar, we categorized the application features into four menu options home, add vehicle, delete vehicle, apply parking sticker.

**Home**

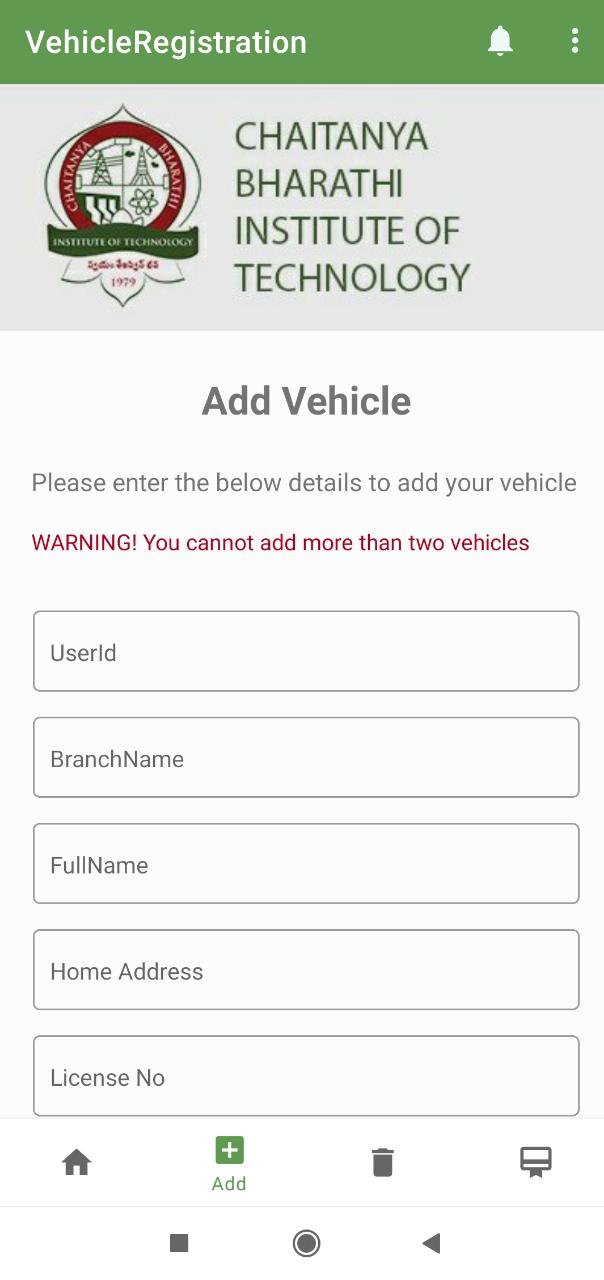
In the home page, registered vehicle details of the user along with his driving license details will be shown. A user can have maximum of two vehicles under his registration. These details are displayed through the GET API call. It shows No vehicle registered if the user did not register any of his vehicles.



Home Page

**Add Vehicle:**

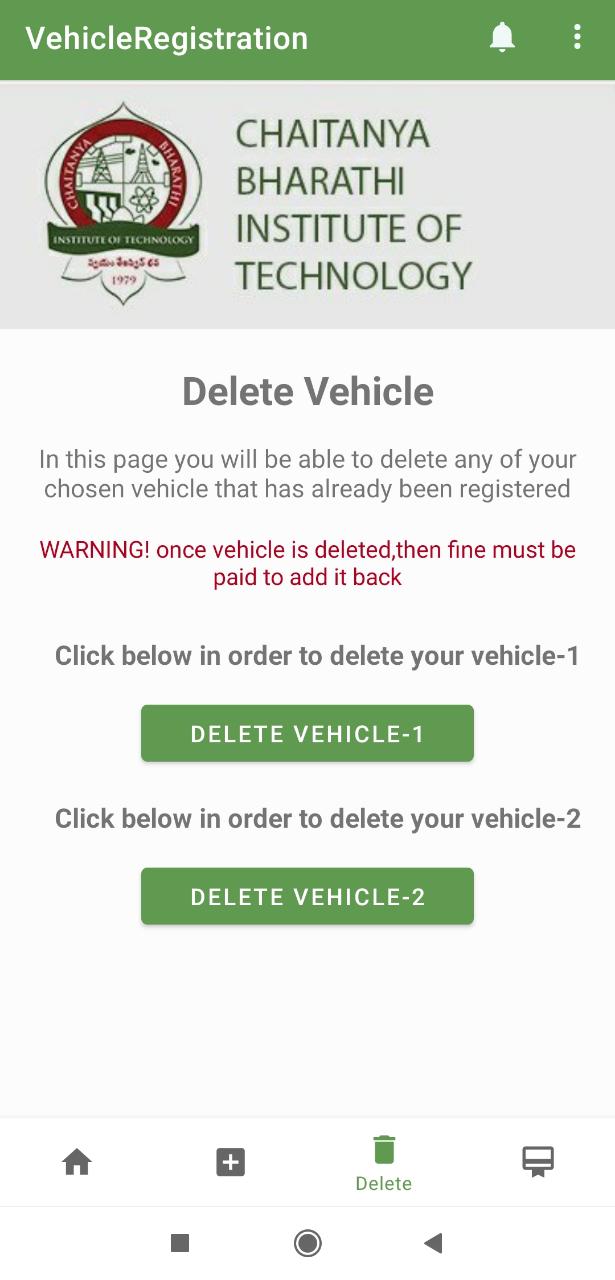
In this user can add his vehicle details by providing details like roll number, full name, branch name, home address, license number, vehicle number, vehicle type, chassis number and registration date. If user tries to add vehicle which is previously deleted a toast message will be displayed saying that vehicle is previously deleted. This adding vehicle details into the database is done through the POST API call.



Add vehicle page

**Delete Vehicle:**

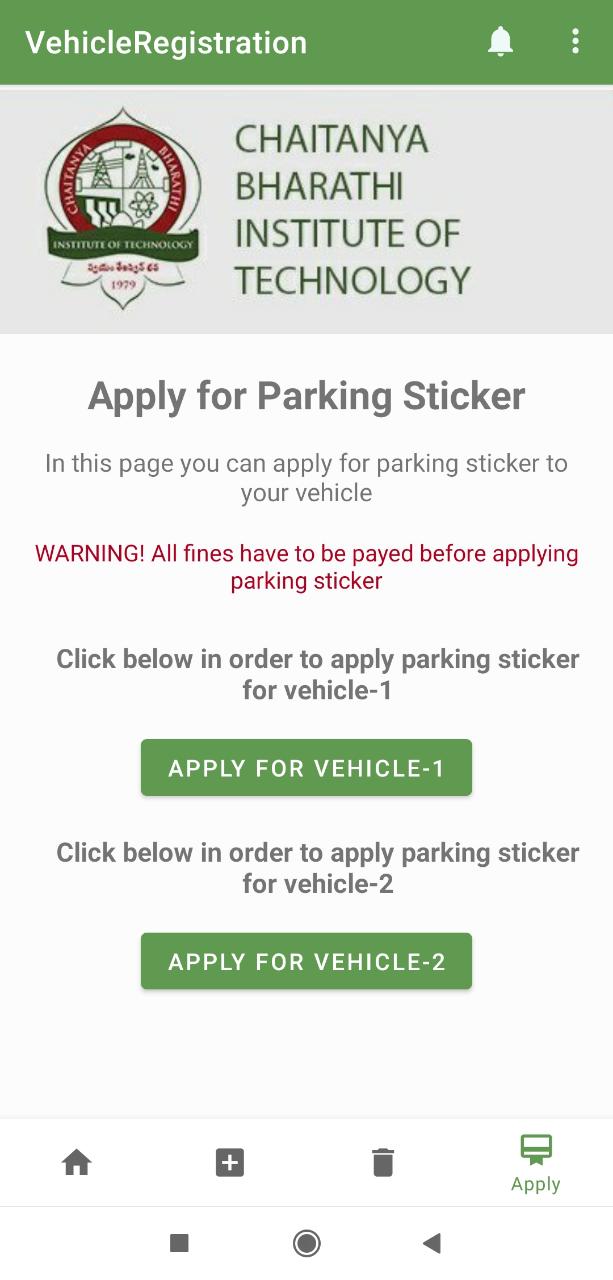
In this user has an option to delete any of his registered vehicle. A button will be displayed corresponding to each vehicle, user can select which vehicle is to be deleted. This is done through PUT API call which updates the status of vehicle as deleted.



Delete Vehicle Page

**Apply Parking Sticker:**

In this user has an option to apply for a new parking sticker for any of his registered vehicles. To apply for a new parking sticker user must pay a fine of 200 rupees. when user applies for the sticker a unique request id is generated and is shown to the user. The fine must be physically paid by the user, on payment of the fine admin issues the new parking sticker. This is done through POST API call.



Apply Parking sticker page

**Web Application (Admin)**

The web application was developed in python with the help of the web framework Django and API calls were made using the requests module. The UI was designed using a combination of CSS and Bootstrap.

**Login**

The login page has two fields, admin ID and password. If the admin enters the wrong details the message “invalid credentials” will be displayed. If the correct details are entered, an access token with which the admin can access all pages will be generated and they will be redirected to registration page.

A sign on the side of a building

Description automatically generated

Login page

**Registrations**

The registrations page consists of a dashboard where all the pending vehicle applications will be displayed. The dashboard consists of a list of cards which contain the user ID and branch name of the student who applied. When each card is clicked, the details such as name, vehicle details and license details of the user will be displayed so that the admin can verify the details. These details are obtained using a GET API call. It will also show an approve button which on clicking will show an alert asking the admin if they are sure they want to approve the vehicle details. If ok is clicked, then the status of the vehicle will be changed from pending to approved based on the reg\_id with the help of a PUT API call. The page will automatically refresh and show the pending applications again.

A screenshot of a social media post

Description automatically generated

Registrations page

**Fines**

It consists of a Dashboard where the User ID along with the date of the pending payments are displayed. On clicking the card all the details regarding the user and fine will be displayed. After receiving the payments manually from the students the admin can update the status of payment to ‘paid’ based on the req\_id.

A screenshot of a cell phone

Description automatically generated

Fines page

**Dismiss Vehicles**

When we click on the 'Dismiss Vehicles' option in the side nav-bar the "Dismiss Vehicles" page is displayed. There is a textbox where we enter the registration id and on pressing enter key, we get the details of the id and there is a button dismiss. If we choose 'dismiss' button the page asks if we are sure to dismiss the vehicle. If we click the ok button or press enter key the vehicle is dismissed and the message that the vehicle is dismissed is shown.

In this page we also have a button which says 'list of dismissed vehicles' and when we click this button it directs us to the page where list of all dismissed vehicles are displayed and we click on them their details are displayed. In this page there is a button 'dismiss vehicles' which takes us back to the 'Dismiss Vehicles' page.

A screenshot of a cell phone

Description automatically generated

Dismiss vehicles page

A screenshot of a cell phone

Description automatically generated

List of dismissed vehicles page

**REGISTRATION STATISTICS**

Here we have date filter dd-mm-yyyy We must select the date of which we require statistics details. After giving the date press submit button. Then it will redirect to same page and details are displayed in form of cards. CARD HEADER contains user id and date. CARD BODY contains details of registered vehicles. Message (“please enter date”) is displayed when admin does not enter date.

A screenshot of a cell phone

Description automatically generated

Registration Statistics page

**PAYMENTS DETAILS**

Here at the top we have payments button. On clicking on that button, it will direct to payment statistics page. Also, here we have date filter. After submitting we will get details of payment statistics. Message (“please enter date”) is displayed when admin does not enter date. If there were no records on that date, then message (“No payments or registrations on that specified date”).

A screenshot of a cell phone

Description automatically generated

Payment Statistics page