



PROJECT REPORT

ON

“GPS Toll based System simulation using Python”



is submitted to

Intel Unnati Industrial Training 2024

Prof. Ram Meghe Institute of Technology and Research, Badnera, Amravati.

By

Mr. Shriyansh R. Fasate

Ms. Shravani S. Shegokar

Under the Guidance of:

Dr. R. R. Karwa

Prof A. U. Chaudhari

Intel Industry Mentor:

Mr. Amit Aggrawal

Mr. Sriharsha Gajavalli



Prof. Ram Meghe Institute of Technology and Research, Badnera,

Amravati.

(An Autonomous Institute & NAAC Accredited)

Sant Gadge Baba Amravati University, Amravati

2024-2025

**PROF. RAM MEGHE INSTITUTE OF TECHNOLOGY AND RESEARCH,
BADNERA, AMRAVATI.**



CERTIFICATE

This is to certify that

Mr. Shriyansh R. Fasate

Ms. Shravani S. Shegokar

has satisfactorily completed the project work towards the **Intel Unnati Industrial Training 2024** in discipline on the topic entitled “**GPS Toll based System simulation using Python**”, during the academic year 2024-2025 under my supervision. The student was trained by Intel experts.

Date:

Dr. R. R. Karwa

Mentor

Prof A. U. Chaudhari

Mentor

ACKNOWLEDGEMENT

We wish to express our sincere thanks to **Intel** for providing us with the opportunity to participate in their internship program. Our heartfelt gratitude goes out to the **Intel Unnati Industrial Training Team** for giving us the chance to take part in their internship program. We would like to thank Intel Industry Expert **Mr. Amit Aggrawal and Mr. Sriharsha Gajavalli.** for their mentorship and guidance, throughout the project. We appreciate their interest to share their knowledge and perspective on the project which enriched our understanding and project direction.

We extend our gratitude to our college “**Prof. Ram Meghe Institute of technology and Research, Badnera, Amravati**” for allowing us to be a part of this training providing academic environment that encouraged us to explore innovative solutions. Through our work on “**GPS Toll based System simulation using Python**”, the internship has been a great way for us to develop our skills, abilities and acquire useful information. We appreciate the chance to improve our Python programming skills and learn more about GPS toll systems.

We are grateful to our supervisors **Dr. Roshan Karwa** and **Prof. Arpit Chaudhari** for their advice and assistance during the course of the program. We express our special thanks to them for providing significant academic help, technical experience, and constant support throughout the project. We express our sincere thanks to everyone who helped us accomplish this project and to convey our sincere gratitude.

Name of Students:



Ms. Shravani S. Shegokar



Mr. Shriyansh R. Fasate

ABSTRACT

At present, FASTags, an electronic toll collection system, uses RFID technology to make toll payments. This system's drawbacks include a set toll collection rate and a lengthy scanning time. The lengthy processing times at toll plazas are caused by operator involvement. This lengthens the wait time for each car and causes gridlock at the toll booth. Congestion on the road can be caused by traditional toll booths, particularly during rush hours. In addition to wasting time, the stop-and-go style of driving increases fuel consumption and air pollution. Our group is committed to addressing this problem by looking at other toll-collection strategies.

The goal of this project is to create a system that can recognize a vehicle moving toward the highway's starting point automatically and start processing the distance traveled along it. The GPS module will trigger an end point on the system where the car diverts from the highway, and a predefined sum is then immediately deducted from the user account if the vehicle chooses not to use the entire stretch of highway. In this case, the GPS-based highway toll collecting system will continuously gather GPS coordinates to determine the location of moving cars.

Keywords: *Automated-toll, Congestion, Coordinates, FASTags, Geo-fences, GPS, Geo coordinates, RFID, Toll-collection, Simulation*

TABLE OF CONTENTS

Sr no.		Contents	Page no.
1		Introduction	1
	1.1	Problem Statement	2
	1.2	Objective	2
	1.3	Motivation	2
2		Literature Review	3
3		Proposed Methodology	6
	3.1	Software and hardware	8
4		Simulation	10
	4.1	Components	10
	4.2	Working	10
	4.3	Special Features	15
5		Result and Discussion	18
6		Conclusion and Future scope	22
7		Individual Contribution	24

1. Introduction

Transportation has emerged as a dominant part of India. Toll plazas play a crucial role in maintaining road transportation. At present, manual toll collection is the most widely used collection method in India. It significantly requires a toll collector or attendant. Due to manual intervention, the processing time at toll plazas is highest. To address these challenges, the project has been designed for the automation of toll tax payments using GPS Technology, which has been experimented using a combination of Geo coordinates and the calculation of distance from initial to the end point of the vehicle traveled route. The purpose of this simulation is to initiate the operation of an automated toll system in the real world.



Figure 1.1: GPS toll system with a car interacting with virtual toll points.

This project aims at designing a system, which automatically identifies the vehicle that advances towards the Starting point of the highway and initiates the system to process the distance traveled through the highway . If a vehicle diverts from the highway rather using the full stretch of highway the GPS module will initiate a end point on the system where the vehicle diverts from the highway and then predetermined amount is automatically taken from the user account Here the GPS based highway toll collection system will acquire GPS coordinates constantly to pinpoint the position of traveling vehicles. The system features an admin portal for remote toll data management, allowing administrators to monitor operations from anywhere, ensuring flexibility and responsiveness to operational demands, eliminating the need for physical presence at specific locations.

1.1 Problem Statement

GPS Toll based System simulation using Python

India's dominant transportation sector relies heavily on toll plazas, but overcharging drivers is a major issue. Our project aims to collect toll tax for precise vehicle distances with the help of GPS coordinates providing accuracy and Convenience. GPS-based toll calculation is a system that uses Global Positioning System (GPS) technology to track a vehicle's location and calculate tolls in real-time.

1.2 Objectives

The application aims to analyze and improve various aspects of the toll collection system. Objectives are as follows:

- To investigate the feasibility and effectiveness of using extracted GPS coordinates stored in a CSV file as a unique identifier for a vehicle's location. The focus will be on leveraging latitude and longitude values to pinpoint the vehicle's position
- To reduce traffic congestion, which would reduce fuel and time consumption
- To Optimize toll price by comparing various factors according to the type of vehicle and distance traveled.
- To design toll plazas by simulating toll booth configurations and workforce levels which maximize efficiency and throughput.

1.3 Motivation

The creation of a GPS-based tolling system was prompted by the need to address the deficiencies of the existing toll collection systems. This cutting-edge approach has several benefits and enhances the toll tax collecting procedure. A GPS-based system allows for the precise and instantaneous tracking of vehicles. By using GPS, toll authorities may eliminate the need for predefined toll rates or toll booths at certain locations by properly calculating toll fees based on the actual distance traveled. The motivation behind implementing a GPS-based toll tax collection system is its ability to provide accurate toll calculations, efficient traffic management, automated toll collection processes, and valuable data insights.

2. Literature Review

This section describes the past works done in the field of “**GPS Toll based System simulation using Python**” by researchers.

An automated toll gate system using vanet by **Senapati**.et al. proposed an automated toll system using vehicular ad hoc networks (VANETs) to reduce total service time (TST) and average waiting time per vehicle on national highways. The proposed routing scheme is compared to the existing manual toll gate system, aiming to save time and reduce traffic congestion.[1]

Automating the Payment of Toll Tax at Toll Plazas by **Kruti Sanghvi**.et al. proposed an RFID approach that transmits a particular ID code as soon as it reaches near the toll station. On receiving the code, the processor checks the received code and compares it with the stored code, if the code matches the gates open else they remain closed disallowing the vehicle to pass.[2]

Vehicle Number Recognition system for Automatic toll tax collection by **Soomro**.et al. proposed an image processing technique to recognize and extract the vehicle number from an image captured at the toll plaza. Vehicle Number Recognition (VNR) is an image processing technology that uses efficient algorithms to detect the vehicle number from real-time images and compare it with the available database to charge toll tax.[3]

Dynamic Approach towards Toll Tax Collection and Vehicle Tracking With the Help of RFID by **R. M. Hushangabade**.et al proposed a dynamic approach toward toll tax collection and vehicle tracking using RFID technology. The system uses RFID tags mounted on vehicles and RFID readers at toll plazas to automatically deduct toll tax and track vehicle movements. Data information exchanged between the owner of the vehicle and toll authorities is done efficiently to increase efficiency and reduce congestion at toll plazas.[4]

Automated toll collection system using GPS and GPRS by **S. K. Nagothu**.et al. proposed that India's infrastructure has to be greatly improved, but the government cannot afford to pay for the work since it is expensive. To recuperate the costs, public-private partnerships are employed; nonetheless, the toll collection system encounters problems such as traffic congestion and lengthy

lines. In order to solve this, GPS-based geo-fences that charge car owners according to where their vehicles are at the toll plaza can be established.[5]

Development of a GPS-based highway toll collection system by **J. Y. Tan**, et al. proposed that Using a Raspberry Pi 2 microcontroller, a GPS-based highway toll collecting system was created that allows drivers to pay tolls without halting or slowing down. The technology logs trip information and detects cars going through designated areas using GPS coordinates. The Raspberry Pi 2 microcontroller serves as a personal cloud server that enables trip logs to be accessed online. This creative solution lowers toll booth expenses and traffic congestion.[6]

Geolocation Process to Perform the Electronic Toll Collection Using the ITS-G5 Technology by **M. Randriamasy**.et al. proposed that The usage of cooperative intelligent transport system (C-ITS) equipment for tolling applications is covered in this study. It emphasizes the necessity of safe communication during transactions and accurate vehicle geolocation. To overcome these problems, the suggested technique combines GPS with extended Kalman filtering and Kalman filtering blended together. The geolocation procedure during tolling services and the outcomes of using a dynamic model based on actual vehicle behavior are also covered in the study.[7]

“A Python Software Platform for Cooperatively Tracking Multiple GPS Receivers” by **Wycoff**.et al. proposed that The goal of this project is to create a software framework that uses the object-oriented design philosophy to analyze data from many GNSS receivers at once. By performing object-specific actions and storing pertinent data, the framework allows for collaborative placement experiments using shared data. This method is critical for data flow between networked receivers.[8]

"PyTrack: A Map-Matching-Based Python Toolbox for Vehicle Trajectory Reconstruction," by **M. Tortora**.et al. proposed that The increasing use of GPS in IoT devices has led to the development of location-based services for Intelligent Transportation System applications. However, due to their inaccuracy, these technologies require map-matching to match trajectories to the real road network. PyTrack, an open-source Python tool, addresses this issue.[9]

“AUTOMATIC TOLL MANAGEMENT SYSTEM USING NFC AND MOBILE COMPUTING.” by Talele et al. proposed that Toll collecting systems, especially contactless payment systems, are using NFC technology more and more. Due to its affordability, dependability, and enhanced security, it is a good choice for web-based toll payment systems for vehicles. NFC tags with distinct UINs and user information are placed by toll authority; these tags are then scanned by NFC readers and validated by a server.[10]

Author	Methodology	Challenges
Senapati et al [1]	Automated toll system using Vehicular Ad hoc Networks (VANETs)	A real test on VANET platform by using the transceiver module in the vehicle. To add some security mechanism in transaction.
Kruti Sanghvi et al. [2]	RFID approach with ID code transmission near toll stations	Vehicle number plate recognition and Aadhar card Linked with Vehicle Number and driving license.
Soomro et al. [3]	Vehicle Number Recognition (VNR) with image processing	Accuracy of license plate recognition in various lighting conditions, also to maintain calibration of cameras with recognition software.
R. M. Hushangbade et al. [4]	Dynamic RFID for toll collection and vehicle tracking, eliminating manual ticket payments and toll fee collections.	Data security and privacy during information exchange and scanning efficiency.
S. K. Nagothu et al. [5]	GPS-based geo-fences for toll charging	Relying solely on GPS accuracy for toll calculation. GPS signal strength in improper weather conditions.
J. Y. Tan et al. [6]	GPS-based highway toll collection with Raspberry Pi	Mitigating infrastructure costs and traffic congestion. Accuracy of the GPS coordinates.
M. Randriamasy et al. [7]	Geolocation with ITS-G5 technology for electronic toll collection	Ensuring secure communication and accurate vehicle positioning. Integrating ITS-G5 technology with existing infrastructure.

3. Proposed Methodology

The workflow for a system simulation involves defining toll zones, simulating vehicle movement, detecting tolls, and calculating toll amounts based on factors like distance traveled and defined vehicle rates. By selecting the starting and destination points of the simulated journey, the user interacts with the simulation. By comparing the coordinates of that route with the coordinates of the zone collected from a.csv file, the system makes sure the vehicle is in the zone.

Once the entered zone gets defined, the system starts calculating the distance being traced by the vehicle until it crosses the end point or becomes out of the zone. When the endpoint is occurred or the vehicle diverts from the zone the point is considered as the destination and the toll-tax respective to the distance traveled up to that point gets automatically deducted from the user account resulting into a user report. This report serves as a digital receipt accessible by the user for various purposes. Users can print, or save the report for future reference or personal record-keeping. Every completed trip is automatically uploaded to the database along with the associated report ID and user details. It serves as an administrator's central repository for tracking and analyzing system utilization in general.

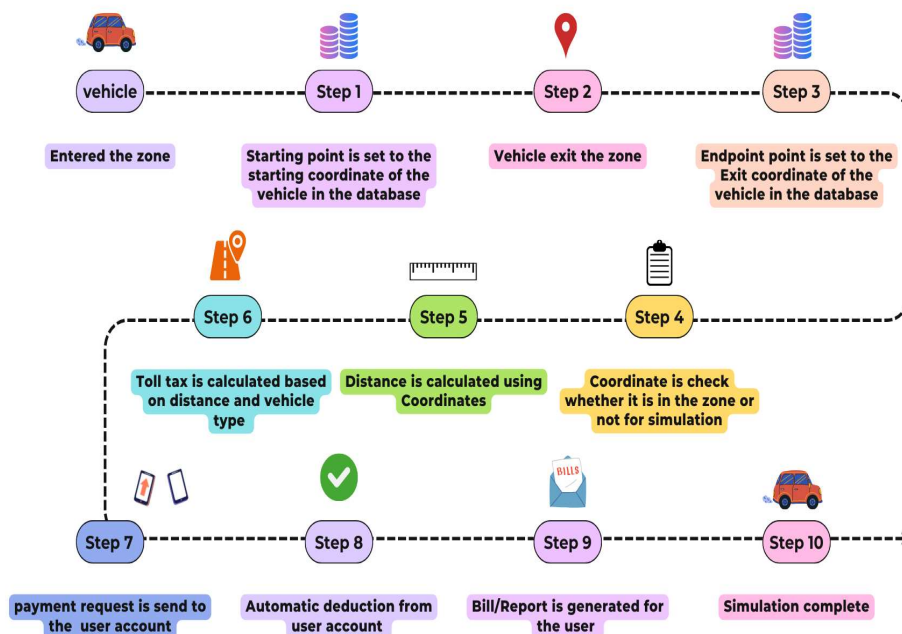


Figure 3.1: flow chart

When the vehicle enters the zone,

- The starting point of the vehicle is set as the starting coordinate in the database.
- The coordinate is checked whether it is in the defined zone or not, if the vehicle is in the zone the distance and toll calculation for the vehicle gets initiated.
- Once the vehicle crosses the end point or escapes through other routes out of the defined zone the point is considered as exit point and is set as the ending coordinate in the database.
- As soon as the vehicle crosses the end point there is automated deduction of toll-tax based on the distance traveled and the type of the vehicle.
- After the deduction from the user's account, a report is generated for the user with the necessary details.

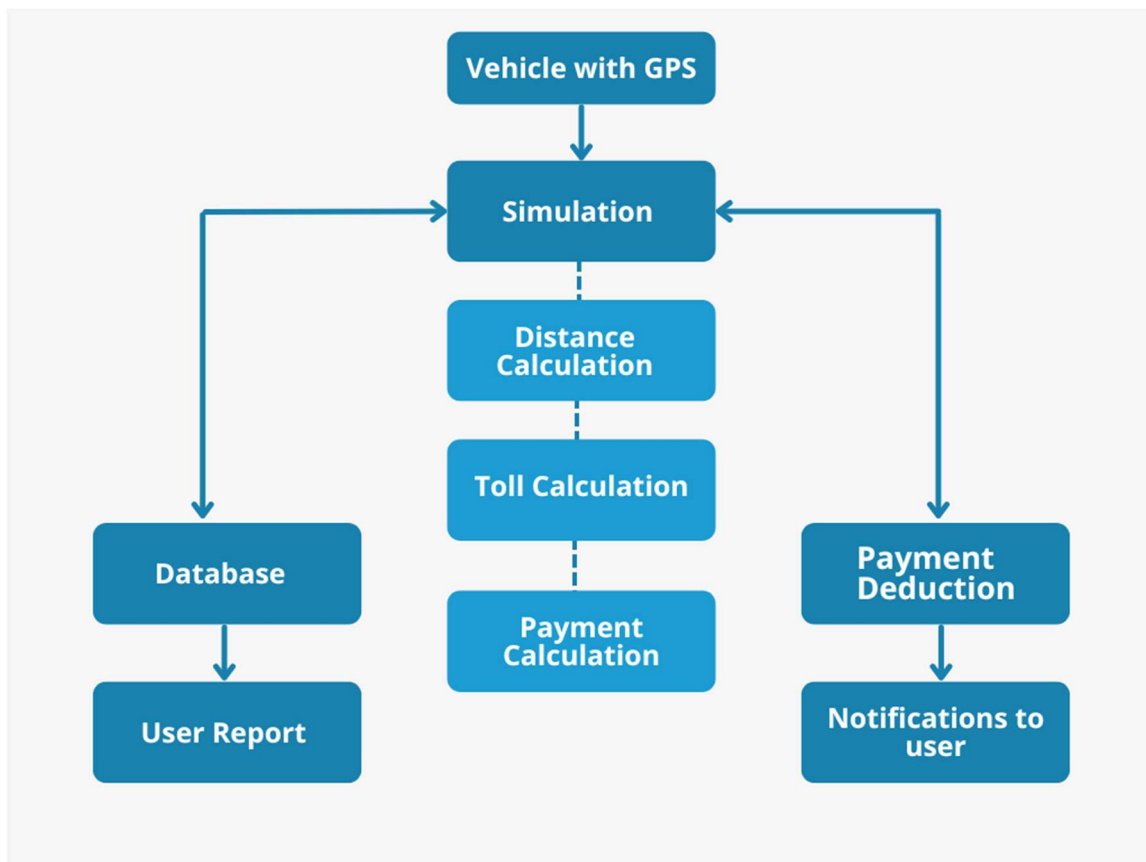


Figure 3.2: Architecture diagram

The architecture diagram depicts the simulation flow of the system from the initial point of the vehicle to the user's report of the entire journey. The bidirectional arrow between the simulation and database indicates the interchange of data values between them. Following are the points which describes the flow of the diagram:

- The process start with fetching the Gps coordinate of the vehicle (here it's in csv.file)
- In the process of simulation the end point of the vehicle is determined and the distance up to that point gets calculated.
- Once the distance is calculated the toll is calculated and there is automated payment deduction.
- The user receives a notification about the toll payment and the user's report is displayed.
- The database is updated from the simulation information.

3.1 Software and Hardware Requirements

The software and hardware requirements for the simulation are enlisted below . However, for a basic simulation, hardware components are not necessary, it can simply define virtual locations and distances within a Python code.

Python is used in the system as it provides various libraries including geopandas, folium, matplotlib, and others for data visualization and simulation. Sqlite3 is used for database management using sqlalchemy. A Flask framework is used to construct an interface for interaction with a database. The application doesn't require a lot of hardware and will work on a range of CPUs, including Intel i3 and i5 models. likewise, several kinds of Intel Arc GPUs will work together, offering a wide range of hardware options.

Requirements	Description
Software	<p>Python: We have used python version 3.11.8 . Python boasts a rich ecosystem of libraries that can enhance our simulation. It also provides Readability , Flexibility, Versatility etc.</p> <p>Libraries: geopandas, folium, matplotlib, geopy ,flask,Sqlalchemy,Werkzeug.security</p> <p>Mapping: Libraries like matplotlib or folium are used for visualizing vehicle movement on a map .</p> <p>Framework : Flask framework is used for user interface database interaction.</p> <p>Database : sqlite3 is used for database management</p> <p>Integrated Development Environment (IDE): Visual Studio Code.</p>
Hardware	<p>Any of the following hardware can be used</p> <p>CPU</p> <ul style="list-style-type: none"> ● intel i5 ● intel i3 <p>GPU</p> <ul style="list-style-type: none"> ● Intel Arc A380 ● Intel Arc A580 ● Intel Arc A750 ● Intel Arc A770

4. Simulation

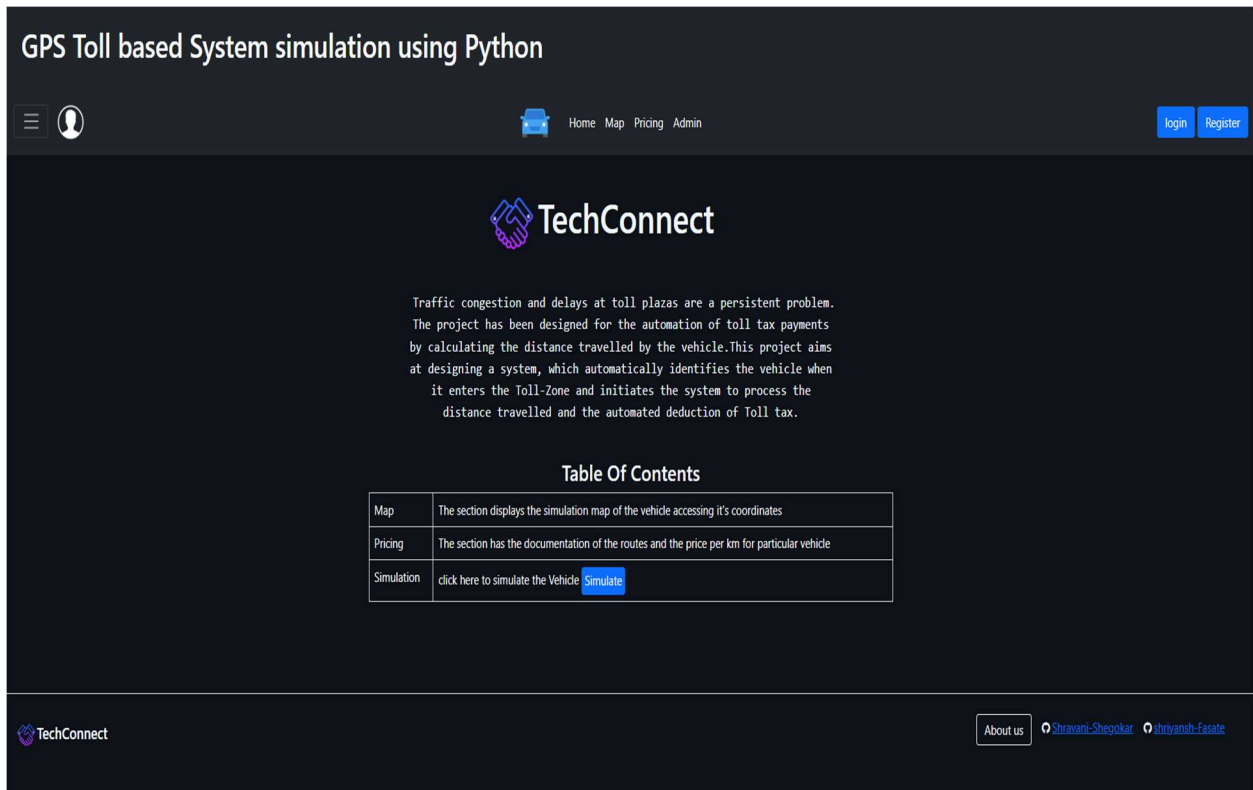
A GPS toll-based system simulation tracks a virtual vehicle's journey and calculates tolls based on distance traveled. Here's a simplified explanation of the components and working of simulation process:

4.1 Components

- **Vehicle Movement Simulation:** Used GPS coordinates Datasets to simulate the movement of vehicles along pre-planned routes.
- **Toll Zone Definition:** Used predefined dataset of coordinates to define toll zones or locations.
- **Distance Calculation:** Calculation of distance traveled by vehicle from starting point to the ending point in the toll zones.
- **Toll Calculation:** Calculate tolls by multiplying the distance traveled or the number of zones crossed . (depending on types of vehicle)
- **Payment Simulation:** Simulating the Automatic deduction process of toll charges (including other,fine,overspeeding) from user accounts.

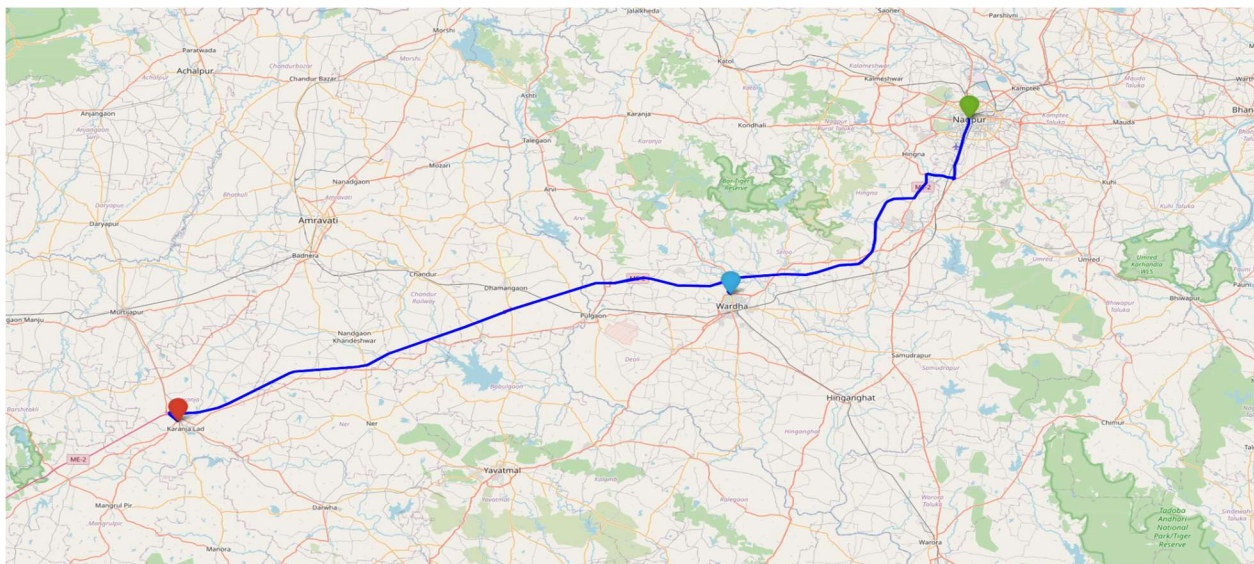
4.2 Working

The **Flask**, a python micro web framework, is used for the simulation of vehicles, toll zones and toll calculations.



Screenshot 4.1: image of application home page

A non-toll road is considered as a toll-zone and has marked the toll points in the defined area.



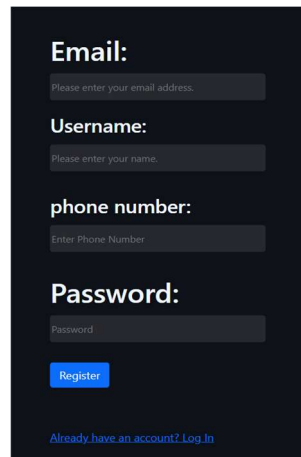
Screenshot 4.2: image of defined area

A **dataset** containing the coordinates (latitude and longitude) of a predetermined path is created. For the testing phase, the considered defined path is the route from Nagpur (Maharashtra, India) to Karanja (Maharashtra, India). In this defined route Nagpur, Wardha and Karanja are marked as the toll points. Even though the starting and ending location are given a name as Nagpur, Wardha and Karanja, but they are the points on the road and are not the exact point of a real map instead they are the points considered on for better distance calculation and simulation. For this, we have used Google My Maps to visualize the path and exported the entire route in **.kml** file format. Further to take out only the coordinates of the entire route as Latitude and Longitude we convert the **.kml** (Keyhole Markup Language) to **.csv** (Comma-Separated Values) using an Online converter. We also got the Coordinates of the endpoints separately.

1	longitude,latitude,
2	77.75023,20.85448,
3	77.75025,20.85472,
4	77.75026,20.8547,
5	77.75029,20.85464,
6	77.75037,20.85451,
7	77.75038,20.8545,
8	77.75043,20.85442,
9	77.75059,20.85424,
10	77.75065,20.85416,
11	77.75069,20.85413,
12	77.75076,20.85406,
13	77.75083,20.85399,
14	77.75109,20.85372,
15	77.7513,20.85355,
16	77.7516,20.85328,
17	77.75216,20.85281,
18	77.75271,20.85231,
19	77.75272,20.8523,
20	77.75355,20.85157,
21	77.75375,20.85141,
22	77.75481,20.85038,
23	77.75586,20.84939,
24	
25	77.75716,20.84816,
26	77.75747,20.84787,
27	77.75766,20.8477,
28	77.7581,20.84729,
29	77.75817,20.84723,
30	77.75982,20.84571,
31	77.75998,20.84557,
32	77.76132,20.84438,
33	77.76202,20.84376,
34	77.76262,20.84322,
35	77.76337,20.84255,
36	77.76335,20.84251
37	

Screenshot 4.3: image of demo coordinates

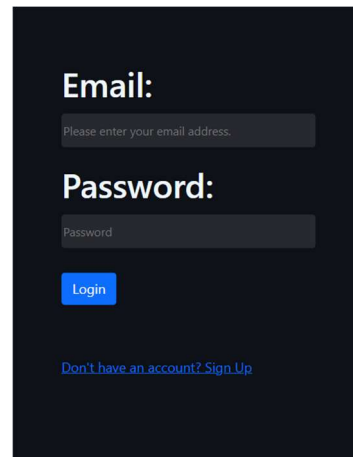
Login and Registration page interface is created to ensure the security of every user's data to maintain Transparency. Users must first log in to utilize the simulation; each user has a profile with their trip history and balance information.



Registration form interface with the following fields and buttons:

- Email:** Input field with placeholder "Please enter your email address."
- Username:** Input field with placeholder "Please enter your name."
- phone number:** Input field with placeholder "Enter Phone Number"
- Password:** Input field with placeholder "Password"
- Register** button (blue)
- [Already have an account? Log In](#) (blue text)

Register



Login form interface with the following fields and buttons:

- Email:** Input field with placeholder "Please enter your email address."
- Password:** Input field with placeholder "Password"
- Login** button (blue)
- [Don't have an account? Sign Up](#) (blue text)

login



User profile interface for user "jhon wick" with phone number "9897100007". It includes a "check Balance" button, a "Recharge" button, and a balance display of "9723.96Rs.". Below this is an "Account Invoice" table showing trip details.

Sr.No	From	To	Vehicle	Distance	fine	Toll-Tax	Total
1	nagpur	wardha	Truck	55.207	0.0	276.04	276.04

Profile

Screenshot 4.4: image of login , registration and profile

Further, whenever the user encounters the **simulate button** in the app an interface opens up resulting in a simulation form, in which the user has to fill up the starting and ending location where he/she wants to go and has to select their type of vehicle. Once the user fill up all the blocks

and clicks on the simulate the simulation of the vehicle starts calculating the distance traveled and toll-tax according to the vehicle type.



Screenshot 4.5: image of simulation button

A dark-themed form titled 'Simulation Details' with a map icon. It contains three sections: 'Coordinates:' with two dropdown menus both set to 'nagpur' separated by 'to'; 'Type Of Vehicle:' with a dropdown menu set to 'Car'; and 'Speed Of Vehicle:' with a dropdown menu set to 'Avg. Speed generated by'. A blue 'simulate' button is at the bottom left.

Screenshot 4.6: simulation details

Here the user has to choose the option for simulation.

The simulation might take some time to execute (nearly 1 min) as throughout this time what actually happens is the starting and ending location given by the user are checked whether they are in the **zone or not**, if the coordinates of location matches with the defined zone there is calculation of the distance traveled by the vehicle from its starting point to the point it exits the zone. whenever the vehicle crosses the endpoint or diverts to other route there comes a flash

message as “**Amount is deducted**” along with the report of the user containing all the necessary details as the tax rate based on type of vehicle, fine on the speed limit (if the speed exceeds limit) which user can print and save in their device.



Screenshot 4.7: Amount deducted message

The payment amount or toll-tax is calculated using the calculated distance and multiplying it with the defined rate of the vehicle chosen by the user in the simulation form. A **wallet** is created for user and multiple payment vendor's are added to recharge the wallet. As the actual bank account is not used, we have already credited an amount of **10,000/-** for every user in their wallet. The tax calculated is deducted from the amount of wallet of the user. This is only a mock payment simulation.

4.3 Special Features

- **Speed :**

The real time speed of a vehicle is not constant, thus in order to calculate our vehicle's average speed rather than considering it as constant we have defined a method. By selecting a random average speed from a list of speed limitations, the algorithm determines the average speed of a car. While the list comprehension creates random integers between 0 and 10, the `random.randint` method produces an integer between 6 and the list length. The average is computed and rounded, and each limit is randomly adjusted by the `modified_limits` loop.

```
speed_limits = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120] # List of speed limits
num_steps = random.randint(6, len(speed_limits))
steps = [random.randrange(10) for _ in range(num_steps)] # Generate random steps

# Add random steps to each speed limit
modified_limits = [limit + step for limit, step in zip(speed_limits, steps)]
avg_speed = round((sum(modified_limits)/12))
```

- **Fine for exceeding the speed**

In order to maintain the congestion free and safe transportation another method is integrated which would charge an extra rate for the speed limit to exceed its particular value. The system checks the autogenerated average speed value and applies charges depending whether it exceeds the limit or not.

```
#method to calculate the fine for the vehicle (if Avg.speed > 50)
def calculate_fine(speed, speed_limit):
    # Checking if driver is over speeding
    if speed >= speed_limit:
        base_fine = 100 # No fine if under speed limit
        excess_speed = speed - speed_limit
        surcharge_per_mph = 5 # Hypothetical surcharge
        speeding_surcharge = excess_speed * surcharge_per_mph
        total_fine = base_fine + speeding_surcharge

    else:
        total_fine=0

    return total_fine
```

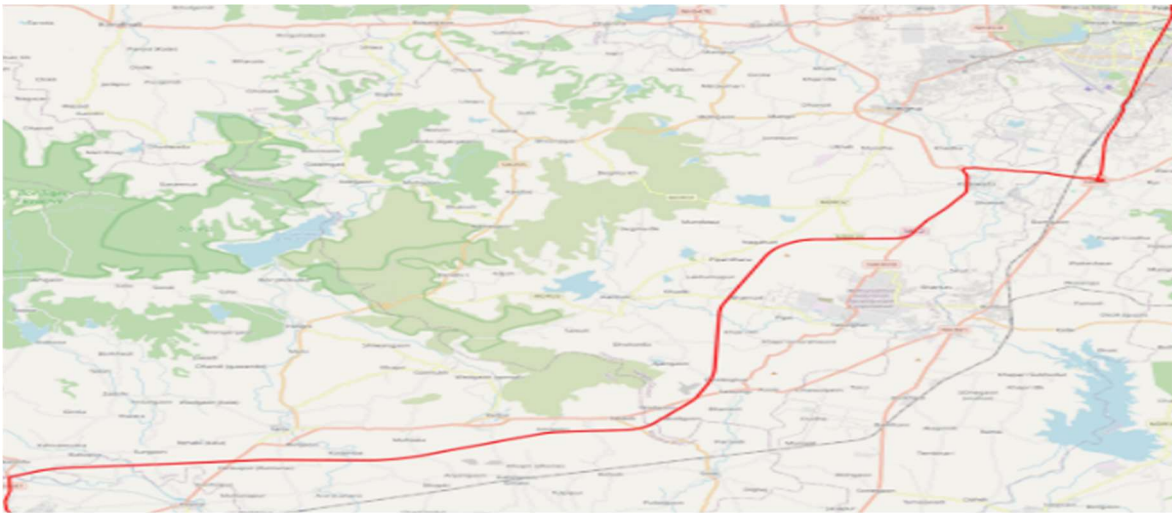
If the vehicle exceed the speed limit a flash message is popup in user



Screenshot 4.8: fine message

- **Visualization of the route traveled by the vehicle**

To plot the route traveled by the vehicle and display it as an image highlighting the traveled route Matplotlib library is used with its “**tilemapbase**” module.



Screenshot 4.9: visualization of path traveled

- **Admin portal**


To monitor the system from anywhere an admin portal is defined which provides the necessary rights to the admin of that particular zone to control the rates over the routes leading to a controlled and efficient transportation of vehicles. There will be a defined password to access the admin portal which would be only known to the admin which defines the security of our app. The admin could control the rates of a particular vehicle for a particular road and will be able to see the whole history of vehicles which enter the zone and their details in the simulation report.

A screenshot of an admin login page. The page has a dark background. It contains three input fields: 'Email:' with a placeholder 'Enter email', 'Password:' with a placeholder 'Enter password', and 'Admin Password:' with a placeholder 'Enter password'. Below these fields is a blue 'Submit' button.

Screenshot 4.10: Admin login page

5. Result and Discussion

The Python simulation effectively illustrated the feasibility of a toll system based on the extracted GPS coordinates. The system eliminates physical toll booths and approaches to improve traffic flow, streamlines congestion, and reduces travel times. Fair and accurate toll charges are guaranteed for users through dynamic toll calculation based upon distance traveled and type of vehicle inside authorized zones. The simulation also established the foundation for a scalable system that can be modified to various road networks, toll structures, and vehicle types. The resulting data offers useful **insights** on distance traveled by the vehicle, payment deduction according to distance traveled and vehicle type for toll optimization.



jhon wick
Mobile : +91 9897100007
Email : jhonwick@gmail.com
country : INDIA

USER ID # 3

INVOICE

Description	Amount
NAGPUR to WARDHA	55.207km.
Avg. Speed	59/km
Time Taken	56min
Vehicle type	Truck
Road Tax:	276.04/-
Late Fees:	0/-
fine:	145/-
Balance:	0/-
TOTAL TAX:	421.04/-

Date :2024-07-04

Thanks From Toll Plaza !!

print

Rates:

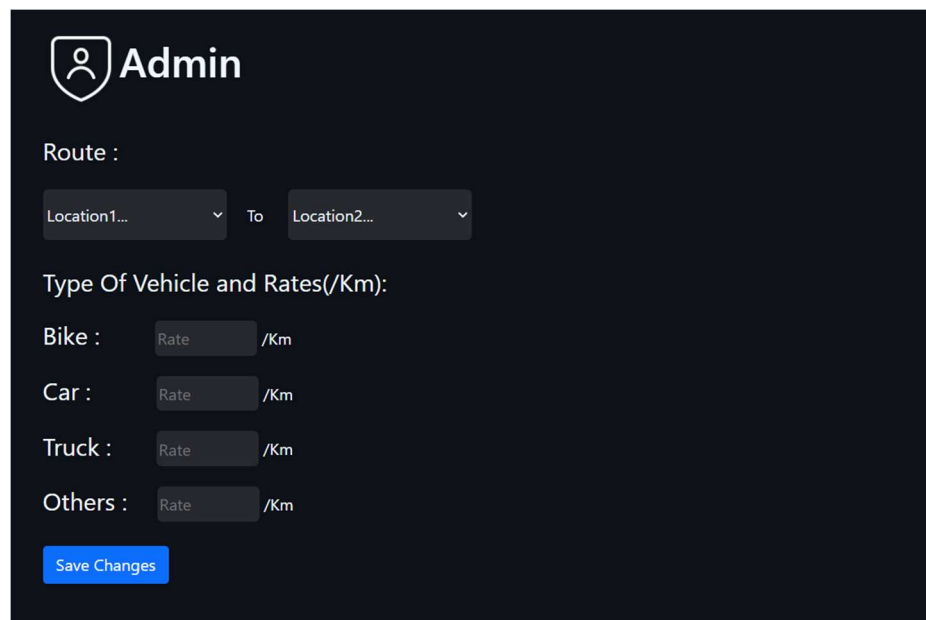
| Bike : 3/Km | Car : 4/Km | Truck : 5/Km | others : 6/Km |

Screenshot 5.1: invoice of traveled path

The existing system's reliance upon a .CSV file for location data is one of its limitations. Although this strategy works, it has limitations with regard to scalability and efficiency.

Real-time GPS tracking would provide several benefits, including streamlined data collecting, increased accuracy, and simpler system expansion for future use.

The project offers a unique combination of user simulation, secure data storage, administrative control, and **dynamic toll rate adjustment**. This makes the system efficient and adaptable to changing traffic conditions, potentially leading to a smoother and more cost-effective transportation experience.



Admin

Route :

Location1... To Location2...

Type Of Vehicle and Rates(/Km):

Bike : /Km

Car : /Km

Truck : /Km

Others : /Km

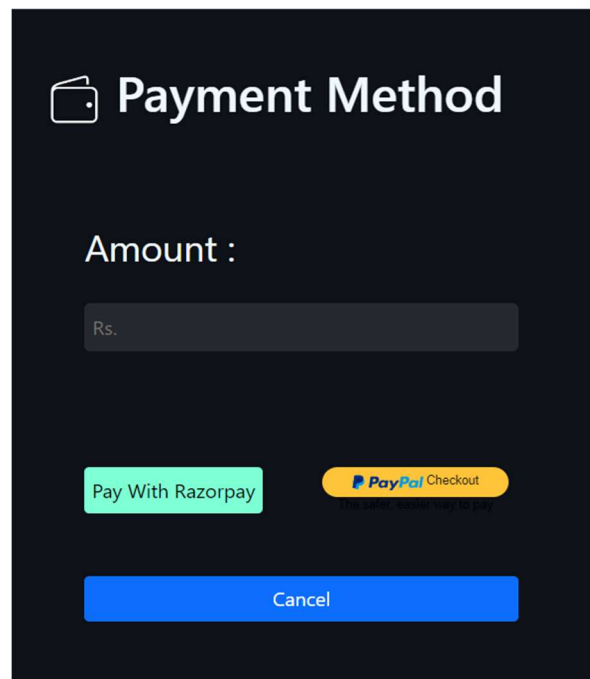
[Save Changes](#)

Vehicle history:

Sr.No	User-id	From		To	Vehicle	Distance	fine	Toll-Tax	Total
1	1	nagpur	↔	wardha	Truck	55.207	0.0	276.04	276.04
2	1	nagpur	↔	karanja	Truck	180.123	0.0	1080.74	1080.74
3	2	nagpur	↔	wardha	Bike	55.207	0.0	165.62	165.62
4	3	nagpur	↔	wardha	Truck	55.207	0.0	276.04	276.04
5	3	nagpur	↔	karanja	Truck	180.123	0.0	1080.74	1080.74

Screenshot 5.2: admin page and user history

In order to select a payment method for payment integration. The project has a recharge option for the user to recharge their wallet for the automated payment of toll-tax. There are two of the various payment vendors for comparison. These include **RazorPay API**, **PayPal API**. There is no API linked to both of payment methods as for the testing purpose we provide a certain amount to the user but whenever there will be real time payment automation the user needs to recharge the wallet to lead a smooth transaction. However, specific requirements like transaction costs, security features, simplicity of integration, and user experience will determine which solution fits best for our project.



Screenshot 3.13: payment method

The following table describes the features and security levels of the payment vendor's used:

Components	RazorPay API	PayPal API
Features	Easy integration with any platform is a well-known feature of the Razorpay API. To ensure developers feasible integration, it provides detailed documentation, client libraries, and SDKs for various kinds of programming languages.	With advanced benefits like recurring payments, fraud prevention, and thorough transaction disclosure, PayPal's API offers secure credit and debit card payment processing. Its extensive customer base is served by its specific alternatives and global reach.
Security	To assure the security of online transactions, Razorpay provides robust safety features like SSL encryption, 3D Secure authentication, and PCI-DSS compliance.	PayPal implements Secure Sockets Layer (SSL) protocol technology with 128-bit data encryption to secure all transactions and is less secure than Razorpay.

6. Conclusion and Future scope

A more effective and efficient toll collection technique is provided by the GPS-toll based system simulation that is integrated into Python. The simulation demonstrates the advantages of utilizing GPS coordinates for dynamic toll calculations. By eliminating physical toll booths, the system has the potential to reduce traffic congestion.

The system is equipped with additional features of Admin portal, Speed limit and the graph displaying the entire route traced by the vehicle. The admin portal makes the system portable providing easy monitoring from any place. The fine application after exceeding the speed limit serves with the management of traffic congestion and the route graph traveled by the vehicle maintains the transparency of the system indicating the exact route, its distance traveled and tax deducted accordingly. Use of the predefined GPS coordinates saved in a .CSV file eliminates the need of real time GPS detection making the simulation unaffected from the external factors like weather condition which could affect the signal strength.

The project has discovered potential areas for growth in the future since including real-time GPS tracking could make the system more effective. The initiative serves as an approach towards efficient and adaptable toll collection strategies. All things considered, this project helps strengthen transportation infrastructure by showing how Python may be used to replicate GPS-based toll systems.

Future Development

- Using a GPS device in order to fetch real time coordinates of the simulating vehicle which would result in fast and efficient simulation within precise time.
- To more effectively regulate traffic congestion, implement time-based tolls, with prices that change dependent on the time of day (peak vs. off-peak hours).
- Investigate linking with navigation apps to give users accessibility to real-time toll information and route optimization that takes into account the flow of traffic.

- The system may establish a direct connection with user bank accounts in order to go beyond a virtual wallet to offer automated toll deductions for a more authentic experience.
- Eliminating the need for a microprocessor, the system can receive location data straight up from the car's integrated GPS, creating setup easier and possibly enhancing accuracy.
- Utilizing cloud services for database storage and real-time monitoring of huge data sets, the scalability of the system can be boosted.
- Using a GPS device to analyze motion of a vehicle and if it is stationary for a longer time than the defined time limit resulting in the toll deduction up to the distance respectively.

7. Individual Contribution

Shravani S. Shegokar

Developed UI for GPS toll-based system simulation with a focus on optimal user experience. Gave precedence to interactivity and ease of use so that customers could easily envision the simulation process without any complications. Moreover, it introduced data visualization capabilities in the feature, depicting displayed costs, locations of tolls and paths simulated.

Shriyansh R. Fasate

The core logic was developed using the Flask framework. Establish the communication between the frontend and backend for a seamless user experience by designing communication protocols. Extract GPS coordinates accurately from CSV data retrieved: develop the toll system simulation program that determines distance between selected toll locations and vehicle-specific tax through the GPS information received.

Reference

1. Senapati, Biswa Ranjan, Pabitra Mohan Khilar, and Naba Krushna Sabat. "An automated toll gate system using vanet." 2019 IEEE 1st international conference on energy, systems and information processing (ICESIP). IEEE, 2019.
2. Sanghvi, Kruti, and Amol Joglekar. "Automating the payment of toll tax at toll plazas." International Journal of Computer Science and Information Technologies 6.3 (2015): 2884-2887.
3. Soomro, Shoaib Rehman, Mohammad Arslan Javed, and Fahad Ahmed Memon. "Vehicle number recognition system for automatic toll tax collection." 2012 international conference of robotics and artificial intelligence. IEEE, 2012.
4. Hushangabade, H., and S. V. Dhopte. "Dynamic Approach towards Toll Tax Collection and Vehicle Tracking With the Help of RFID." International Journal of Engineering and Innovative Technology 3.1 (2013): 368-371.
5. S. K. Nagothu, "Automated toll collection system using GPS and GPRS," 2016 International Conference on Communication and Signal Processing (ICCSP), Melmaruvathur, India, 2016, pp. 0651-0653, doi: 10.1109/ICCSP.2016.7754222.
6. J. Y. Tan, P. J. Ker, D. Mani and P. Arumugam, "Development of a GPS-based highway toll collection system," 2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, 2016, pp. 125-128, doi: 10.1109/ICCSCE.2016.7893557.

7. M. Randriamasy, A. Cabani, H. Chafouk and G. Fremont, "Geolocation Process to Perform the Electronic Toll Collection Using the ITS-G5 Technology," in *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8570-8582, Sept. 2019, doi: 10.1109/TVT.2019.2931883.
8. Wycoff, Eliot, Gao, Grace Xingxin, "A Python Software Platform for Cooperatively Tracking Multiple GPS Receivers," *Proceedings of the 27th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2014)*, Tampa, Florida, September 2014, pp. 1417-1425.
9. M. Tortora, E. Cordelli and P. Soda, "PyTrack: A Map-Matching-Based Python Toolbox for Vehicle Trajectory Reconstruction," in *IEEE Access*, vol. 10, pp. 112713-112720, 2022, doi: 10.1109/ACCESS.2022.3216565.
10. Talele, K.H., Bharambe, D., Pawar, K., & Mhaisgawali, P.A. (2020). "AUTOMATIC TOLL MANAGEMENT SYSTEM USING NFC AND MOBILE COMPUTING."
11. Rafiya Hossain, Moonmoon Ahmed, Md. Mozadded Al-fasani, Hasan U. Zaman,(2008), "Advanced Security System Integrated With RFID Based Automated Toll Collection System", Third Asian Conference on Defence Technology.