Intel® Unnati Industrial Training Program 2024

“GPS based Toll System using Python”

Submitted by:

AKKSHITA PRABHU L

1EP22EC004

Department of Electronics and Communication Engineering

East Point College of Engineering and Technology

Under the guidance of

Parvati Patil

Department of Electronics and Communication

East Point College of Engineering and Technology

Jnana Prabha, East Point Campus, Virgonagar Post, Avalahalli, Bangalore-560049
Department of Electronics & Communication Engineering 2023-2024

ACKNOWLEDGEMENT

Behind every achievement lies an unfathomable sea of gratitude to those who actuated it, without whom it would never have come to existence. To them our praise the word of gratitude imprinted not just on this paper but deep in our heart.

First and Foremost, I would like to convey my heartfelt thanks to the **Intel® Unnati Industrial Training Internship Team** for giving us this wonderful opportunity and for the exposure we gain through this Program.

I express my profound gratitude towards **Dr. S M Venkatpathi**, **Chairman**, **East Point College of Engineering and Technology**, for providing necessary infrastructure.

Also honour to **Dr. Mrityunjaya V Latte**, **Principal**, **EPCET**, for creating good environment for carrying out our project.

I express our sincere regards and thanks **to Dr. Yogesh G S**, **Vice Principal** and Head of the department, ECE, **EPCET**, for continuously keeping us

Our sincere thanks to our guide **Mrs. Parvati Patil** and **Mrs. Malini V L**, Assistant Professors, Department of ECE, for their guidance and encouragement throughout the course of project preparation and completion. Their timely assistance and valuable suggestions were the motivation factor for the completion of the project successfully.

We also thank all the teaching and non-teaching staff of EPCET, for their Valuable guidance. Last but not least, we would like to utilize this opportunity to express a sense of gratitude and love to beloved family and to our dearest friends for their support and strength.

INTRODUCTION

In today's fast-paced world, efficient transportation systems are crucial for economic development and societal progress. Among the various components of these systems, toll roads play a vital role in managing traffic flow, reducing congestion, and funding infrastructure maintenance and expansion. However, for many drivers, navigating the complexities of toll calculations and route planning can be a daunting task. This project, the GPS based Toll Road Simulator, aims to address these challenges by providing an innovative, user-friendly solution that simplifies the process of calculating toll costs based on selected routes and vehicle types.

The GPS based Toll Road Simulator is designed with the primary goal of enhancing the user experience by integrating interactive mapping technologies with real-time toll calculation. This application allows users to visualize toll points on a map, select their desired routes by clicking on these points, and calculate the total toll costs based on the distance travelled and the type of vehicle used. By offering an intuitive interface and dynamic toll calculation, the Toll Road Simulator aims to make route planning more accessible and efficient for drivers.

One of the unique aspects of this project is its focus on user interactivity and ease of use. Traditional methods of toll calculation often involve manual look-ups or complex navigation through multiple websites and documents. In contrast, the GPS based Toll Road Simulator leverages graphical user interface (GUI) technologies to provide a seamless and engaging user experience. With features such as toll point selection through map interaction, vehicle type customization, and hover information displaying toll point names, the application ensures that users can plan their journeys with minimal effort and maximum accuracy.

The technological foundation of the GPS based Toll Road Simulator is built on Python, a versatile and powerful programming language known for its ease of use and extensive library support. Key libraries used in this project include Tkinter for the GUI, Pillow (PIL) for image processing and map display, and Geopy for precise distance calculations between GPS coordinates. These technologies work in concert to deliver a robust and responsive application capable of handling the demands of real-time toll calculations and interactive map navigation.

At the heart of the GPS based Toll Road Simulator is the interactive map, which provides a visual representation of toll points across India. Users can click on these points to add them to their route, with the application dynamically calculating the total distance and associated toll costs. The inclusion of vehicle type selection further customizes the toll calculation, taking into account the varying rates for different types of vehicles such as cars, trucks, and commercial vehicles. This feature ensures that the toll cost estimates are accurate and tailored to the specific needs of each user.

The GPS based Toll Road Simulator also includes a simulated payment gateway, allowing users to experience a complete end-to-end journey planning and toll payment process. By integrating this feature, the application not only calculates toll costs but also simulates the subsequent steps a driver would take to settle their toll charges, thereby providing a holistic solution to the problem at hand.

To summarize, the GPS based Toll Road Simulator is a comprehensive and innovative tool designed to simplify the complexities of toll road navigation and cost calculation. By combining interactive mapping, real-time toll calculations, and user-friendly interfaces, this project aims to make route planning more accessible and efficient for drivers. The use of Python and its supporting libraries ensures a robust and scalable application that can be expanded to include additional features and regions in the future. Through the Toll Road Simulator, we envision a future where drivers can plan their journeys with ease, ensuring smoother and more efficient travel experiences on toll roads.

PROBLEM STATEMENT

Transportation infrastructure, particularly toll roads, is essential for the economic and social development of a country. Toll roads help manage traffic flow, reduce congestion, and generate revenue for the maintenance and expansion of road networks. However, the process of calculating toll costs and planning routes is often cumbersome for drivers.

The problem statement and the objective is to create a GPS based Toll System to eliminate the traffic and congestion at the Toll Plazas on Indian Highways.

To understand the drawbacks of manual toll collection, such as traffic congestion, long queues, and environmental impact due to vehicle emissions.

To Understand the frustration of waiting in long queues on busy days and to ease the process the toll fee payment and collection.

To increase the accuracy of the system by using GPS to track the movement of vehicles.

To ensure a seamless travelling experience for the user as well as a sure and secure mode of collecting toll revenue for the authorities.

Drivers face several challenges:

1. **Manual Toll Calculation**: Toll costs vary based on the type of vehicle and the distance travelled. Manually calculating these costs involves looking up toll rates and distances between multiple toll points, which is time-consuming and error-prone.
2. **Route Planning**: Determining the most cost-effective and efficient route requires drivers to consider multiple toll points, each with different rates, which complicates route planning.
3. **Lack of Interactive Tools**: Existing tools for toll calculation are often static and do not provide interactive features that allow users to visually select their routes and see toll costs dynamically.

These challenges highlight the need for an intuitive and interactive tool that simplifies toll calculation and route planning for drivers. Such a tool should provide real-time calculations, allow for easy selection of toll points, and accommodate different vehicle types to offer accurate toll costs.

UNIQUE IDEA AND SOLUTIONS

The Toll Road Simulator addresses these challenges by providing an interactive, map-based application that simplifies toll calculation and route planning. This project integrates several innovative solutions to enhance the user experience:

1. **Interactive Map Interface**: The core of the Toll Road Simulator is its interactive map, which visually represents toll points across India. Users can click on these toll points to select their route. This visual approach makes it easy to plan routes and understand the layout of toll points.
2. **Dynamic Toll Calculation**: The application calculates the total toll cost in real-time based on the selected route and vehicle type. As users select or deselect toll points, the application updates the total distance and toll cost dynamically, providing immediate feedback.
3. **Vehicle Type Customization**: Different vehicle types have varying toll rates. The application includes a dropdown menu where users can select their vehicle type, ensuring that the toll calculations are accurate and tailored to their specific needs.
4. **Hover Information**: To further enhance the user experience, the application displays the names of toll points when users hover their mouse over them. This feature helps users easily identify and select the desired toll points.
5. **Simulated Payment Gateway**: After calculating the toll costs, users can proceed to a simulated payment gateway, which provides a complete end-to-end experience. This feature allows users to understand the full process of route planning, toll calculation, and payment.
6. **Ease of Use**: By integrating these features into a single application, the Toll Road Simulator ensures that users can plan their journeys with minimal effort. The intuitive interface, combined with real-time calculations and interactive elements, simplifies the entire process.
7. **Scalability**: The application's design allows for easy expansion. Additional toll points can be added, and the map can be updated to include new regions. This scalability ensures that the application remains relevant and useful as road networks evolve.

FEATURES OFFERED BY THE SYSTEM

The Toll Road Simulator offers a range of features designed to simplify the process of route planning and toll calculation for drivers. These features include:

1. **Interactive Map Interface**: The application features a visually interactive map of India, displaying toll points as clickable markers. Users can select their desired route by simply clicking on these markers, which makes the route planning process intuitive and straightforward.
2. **Real-time Toll Calculation**: As users select or deselect toll points on the map, the application dynamically calculates the total distance of the route and the associated toll costs based on the selected vehicle type. This real-time feedback ensures that users have accurate and up-to-date information to make informed decisions.
3. **Vehicle Type Selection**: The toll rates vary depending on the type of vehicle. The application includes a dropdown menu that allows users to select their vehicle type (e.g., Car/Jeep/Van/LMV, Light Commercial Vehicle/LGV/Mini-Bus, Bus/Truck, etc.). This feature ensures that the toll cost calculations are specific to the type of vehicle being used.
4. **Hover Information**: To aid in route planning, the application displays the names of toll points when users hover their mouse over them. This feature provides additional context and helps users easily identify and select the correct toll points.
5. **Route Confirmation**: After selecting the desired toll points, users can confirm their route with a single key press (Enter). The application then provides a detailed summary of the total distance and the toll cost for the selected route.
6. **Simulated Payment Gateway**: To provide a complete user experience, the application includes a simulated payment gateway. After calculating the toll costs, users can proceed to this gateway to simulate the payment process, enhancing the realism and functionality of the application.

PROCESS FLOW

The process flow of the Toll Road Simulator can be broken down into the following steps:

1. **Vehicle Type Selection**:
    - Users start by selecting their vehicle type from a dropdown menu on the initial screen.
    - This selection is crucial as it determines the toll rate applied during the calculation process.
2. **Interactive Route Planning**:
    - Once the vehicle type is selected, users are presented with an interactive map displaying various toll points across India.
    - Users can click on toll points to add or remove them from their route. Selected toll points change colour to indicate inclusion in the route.
3. **Dynamic Toll Calculation**:
    - As users select toll points, the application calculates the total distance between the selected points and computes the toll cost based on the vehicle type.
    - The total distance and toll cost are displayed in real-time, providing immediate feedback.
4. **Route Confirmation**:
    - After planning the route, users press Enter to confirm their selections.
    - The application provides a detailed summary of the total distance and toll cost for the selected route.
5. **Simulated Payment Process**:
    - Users can choose to proceed to a simulated payment gateway to complete the toll payment.
    - The application simulates the payment process and provides feedback on the success or cancellation of the payment.

ARCHITECTURE DIAGRAM

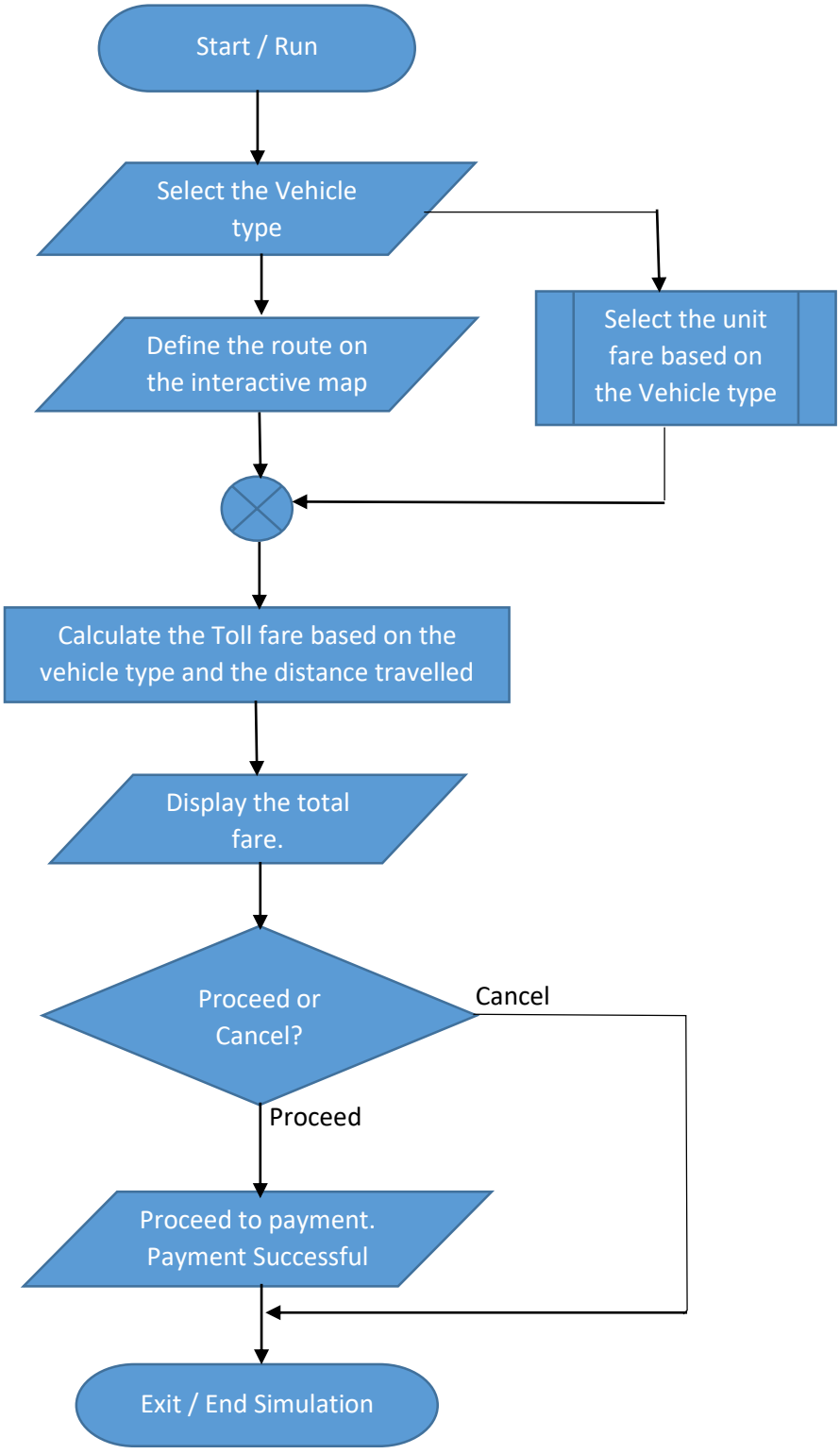


Figure 1: Flowchart depicting the architecture of the GPS based Toll System

The explanation for the flow chart is as follows

1. **Start**: The entry point of the application.
2. **Vehicle Type Selection**: Users begin by selecting their vehicle type from a dropdown menu.
3. **Interactive Map with Toll Points**: Users are presented with an interactive map displaying toll points across India.
4. **Dynamic Toll Calculation**: As users select toll points, the application calculates the total distance and toll cost in real-time based on the vehicle type.
5. **Route Confirmation**: Users confirm their selected route by pressing Enter.
6. **Payment Simulation**: Users proceed to a simulated payment gateway to complete the toll payment.
7. **End**: The process concludes

TECHNOLOGIES USED

The GPS based Toll Road Simulator leverages several technologies to provide its features and functionality in the form of Python Libraries. They are:

1. **Python**: The primary programming language used for developing the application. Python's simplicity and extensive library support make it an ideal choice for this project.

2. **Tkinter**: A standard GUI (Graphical User Interface) library in Python, Tkinter is used to create the application's user interface. It handles the creation of windows, buttons, labels, and other UI elements.

3. **Pillow (PIL)**: A powerful image processing library in Python. Pillow is used to load and display the map image, as well as to handle image resizing and manipulation.

4. **Geopy**: A library for geocoding and distance calculations. Geopy is used to calculate the distance between GPS coordinates, which is essential for computing the total distance of the selected route.

TEAM MEMBERS AND CONTRIBUTIONS

This project is built and submitted by me, Akkshita Prabhu L from East Point College of Engineering and Technology. I am a student pursuing Bachelor of Engineering in Electronics and Communication. I am currently in my second year and fourth semester of my course. I have taken up this internship as an individual applicant. I have thereby made all the contributions myself.

PROJECT CODING AND PYTHON PROGRAM

```
import tkinter as tk
from tkinter import messagebox, ttk
from PIL import Image, ImageTk
from geopy.distance import geodesic
# Define toll zones with GPS coordinates and names (example toll points in
India)
TOLL_ZONES = [
 ((28.613939, 77.209021), 5, "Delhi"),
 ((19.076090, 72.877426), 7, "Mumbai"),
 ((12.971599, 77.594566), 6, "Bangalore"),
 ((13.082680, 80.270718), 5, "Chennai"),
 ((22.572645, 88.363892), 7, "Kolkata"),
 ((26.912434, 75.787270), 4, "Jaipur"),
 ((23.259933, 77.412613), 4, "Bhopal"),
 ((18.520430, 73.856743), 4, "Pune"),
 ((17.385044, 78.486671), 5, "Hyderabad"),
 ((21.170240, 72.831062), 4, "Surat"),
 ((15.317277, 75.713888), 4, "Hubli"),
 ((11.016844, 76.955832), 4, "Coimbatore"),
 ((9.925201, 78.119774), 4, "Madurai"),
 ((15.849695, 74.497674), 4, "Belgaum"),
 ((19.751480, 75.713888), 4, "Aurangabad"),
 ((22.719568, 75.857727), 4, "Indore"),
 ((23.022505, 72.571362), 4, "Ahmedabad"),
 ((24.585445, 73.712479), 4, "Udaipur"),
 ((26.846708, 80.946159), 4, "Lucknow"),
 ((27.023804, 74.217933), 4, "Ajmer"),
 ((29.945690, 78.164247), 4, "Haridwar"),
 ((30.316496, 78.032188), 4, "Dehradun"),
 ((31.104814, 77.173403), 4, "Shimla"),
 ((32.726602, 74.857026), 4, "Jammu"),
 ((34.083656, 74.797371), 4, "Srinagar"),
 ((25.317645, 82.973914), 4, "Varanasi"),
 ((25.594095, 85.137566), 4, "Patna"),
 ((26.449923, 74.639916), 4, "Ajmer"),
 ((28.408912, 77.317789), 4, "Faridabad"),
 ((28.459497, 77.026638), 4, "Gurgaon"),
 ((30.901011, 75.857276), 4, "Ludhiana"),
 ((25.448425, 78.568459), 4, "Jhansi"),
 ((23.610180, 85.279935), 4, "Ranchi"),
 ((23.831457, 91.286778), 4, "Agartala"),
 ((24.800615, 93.950017), 4, "Imphal"),
 ((27.471012, 94.911964), 4, "Dimapur"),
 ((27.176670, 78.008075), 4, "Agra"),
 ((22.719568, 75.857727), 4, "Indore"),
 ((22.307159, 73.181219), 4, "Vadodara"),
 ((21.145800, 79.088155), 4, "Nagpur"),
 ((20.937424, 77.779551), 4, "Amravati"),
 ((19.876165, 75.343314), 4, "Jalna"),
 ((17.686816, 83.218482), 4, "Visakhapatnam"),
 ((16.506174, 80.648015), 4, "Vijayawada"),
 ((15.139393, 76.921443), 4, "Bellary"),
```

```
 ((13.628756, 79.419181), 4, "Tirupati"),
 ((12.295810, 76.639381), 4, "Mysore"),
 ((11.941591, 79.808313), 4, "Puducherry"),
 ((10.786999, 78.705566), 4, "Tiruchirappalli"),
 ((9.287625, 76.654793), 4, "Kottayam"),
 ((8.524139, 76.936638), 4, "Thiruvananthapuram"),
 ((26.144518, 91.736237), 4, "Guwahati"),
 ((25.360809, 85.011017), 4, "Bihar Sharif"),
 ((24.412966, 85.323961), 4, "Hazaribagh"),
 ((22.998763, 87.854975), 4, "Durgapur"),
 ((22.572646, 88.363895), 4, "Kolkata"),
 ((20.296059, 85.824540), 4, "Bhubaneswar"),
 ((21.145800, 79.088155), 4, "Nagpur"),
 ((16.994444, 73.300000), 4, "Kolhapur"),
 ((12.971598, 77.594566), 4, "Bangalore"),
 ((10.850516, 76.271083), 4, "Palakkad"),
 ((11.740086, 92.658640), 4, "Port Blair"),
 ((28.669156, 77.453758), 4, "Ghaziabad"),
 ((27.204612, 77.497684), 4, "Mathura"),
 ((25.448425, 78.568459), 4, "Jhansi"),
 ((23.634501, 92.731681), 4, "Aizawl"),
 ((22.806943, 86.202875), 4, "Jamshedpur"),
 ((21.170240, 72.831062), 4, "Surat"),
 ((19.876165, 75.343314), 4, "Jalna"),
 ((18.112437, 79.019301), 4, "Warangal"),
 ((15.828126, 78.037279), 4, "Kadapa"),
 ((13.082680, 80.270718), 4, "Chennai"),
 ((10.991621, 76.961632), 4, "Coimbatore"),
 ((9.939093, 78.121719), 4, "Madurai"),
 ((8.088306, 77.538451), 4, "Nagercoil"),
 ((12.971598, 77.594566), 4, "Bangalore"),
 ((11.127123, 78.656891), 4, "Erode"),
 ((10.167168, 76.641271), 4, "Kottayam"),
 ((8.713912, 77.756652), 4, "Tirunelveli"),
 ((11.108524, 77.341066), 4, "Tiruppur"),
 ((12.971598, 77.594566), 4, "Bangalore"),
 ((12.295810, 76.639381), 4, "Mysore"),
 ((10.850516, 76.271083), 4, "Palakkad"),
]
# Define canvas size
CANVAS_WIDTH = 800
CANVAS_HEIGHT = 800
# Define the mapping from canvas coordinates to GPS coordinates (you'll need
to
fine-tune this)
def canvas_to_gps(x, y):
 # Map image dimensions
 img_width = 800
 img_height = 800
 # GPS bounds (top-left and bottom-right corners of the map image)
 gps_top_left = (35.0, 68.0) # Example coordinates
 gps_bottom_right = (5.0, 97.0) # Example coordinates
 lat = gps_top_left[0] - (y / img_height) * (gps_top_left[0] -
gps_bottom_right[0])
```

```
 lon = gps_top_left[1] + (x / img_width) * (gps_bottom_right[1] -
gps_top_left[1])
 return (lat, lon)
# Convert GPS coordinates to canvas coordinates
def gps_to_canvas(lat, lon):
 # Map image dimensions
 img_width = 800
 img_height = 800
 # GPS bounds (top-left and bottom-right corners of the map image)
 gps_top_left = (35.0, 68.0) # Example coordinates
 gps_bottom_right = (5.0, 97.0) # Example coordinates
 x = (lon - gps_top_left[1]) / (gps_bottom_right[1] - gps_top_left[1]) *
img_width
 y = (gps_top_left[0] - lat) / (gps_top_left[0] - gps_bottom_right[0]) *
img_height
 return (x, y)
class VehicleSelection(tk.Tk):
 def __init__(self):
 super().__init__()
 self.title("Select Vehicle Type")
 self.geometry("400x200")
 self.resizable(False, False)
 self.vehicle_type_var = tk.StringVar()
 self.vehicle_type_var.set('Select Vehicle Type')
 ttk.Label(self, text="Select Vehicle Type").pack(pady=10)
 self.vehicle_type_menu = ttk.Combobox(self,
textvariable=self.vehicle_type_var)
 self.vehicle_type_menu['values'] = [
 'Car/Jeep/Van/LMV',
 'Light Commercial Vehicle/LGV/Mini-Bus',
 'Bus/Truck (2 axles)',
 'Three-axle Commercial Vehicle',
 'Heavy Construction Machinery/Multi-axle (4-6 axles)',
 'Oversized Vehicle (7+ axles)'
 ]
 self.vehicle_type_menu.pack(pady=10)
 self.proceed_button = ttk.Button(self, text="Next",
command=self.start_route_selection)
 self.proceed_button.pack(side=tk.LEFT, padx=20, pady=20)
 self.cancel_button = ttk.Button(self, text="Cancel", command=self.destroy)
 self.cancel_button.pack(side=tk.RIGHT, padx=20, pady=20)
 def start_route_selection(self):
 vehicle_type = self.vehicle_type_var.get()
 if vehicle_type == 'Select Vehicle Type' or vehicle_type == '':
 messagebox.showwarning("Select Vehicle Type", "Please select a vehicle
type.")
 return
 self.destroy()
 app = TollSimulator(vehicle_type)
 app.mainloop()
class TollSimulator(tk.Tk):
 def __init__(self, vehicle_type):
 super().__init__()
 self.vehicle_type = vehicle_type
```

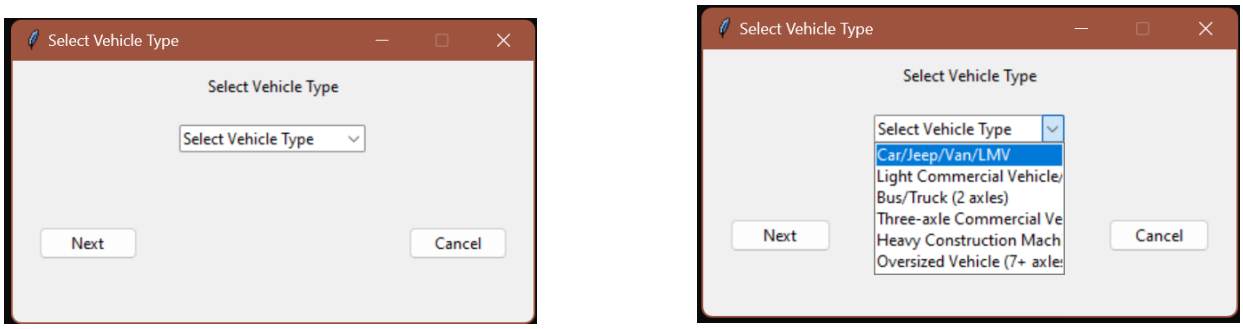
```
 self.title("Toll Road Simulator")
 self.geometry(f"{CANVAS_WIDTH}x{CANVAS_HEIGHT+100}") # Adjusted window
size to fit the image and dropdown
 self.canvas = tk.Canvas(self, width=CANVAS_WIDTH, height=CANVAS_HEIGHT)
 self.canvas.pack()
 # Load and scale down the map image
 self.map_image =
Image.open("C:/Users/akksh/OneDrive/Desktop/india-road-map.gif")
 self.map_image = self.map_image.resize((CANVAS_WIDTH, CANVAS_HEIGHT),
Image.LANCZOS)
 self.map_photo = ImageTk.PhotoImage(self.map_image)
 self.canvas.create_image(0, 0, anchor=tk.NW, image=self.map_photo)
 self.toll_points = []
 self.route_points = []
 self.hover_label = tk.Label(self, text="", bg="yellow", fg="black")
 for zone in TOLL_ZONES:
 x, y = gps_to_canvas(*zone[0])
 point = self.canvas.create_oval(x-5, y-5, x+5, y+5, fill="blue",
outline="black")
 self.toll_points.append((point, zone[0], zone[2]))
 self.canvas.bind("<Button-1>", self.on_click)
 self.canvas.bind("<Motion>", self.on_motion)
 self.bind("<Return>", self.confirm_route)
 self.status_label = tk.Label(self, text="Click on toll points to mark your
route. Press Enter to confirm.")
 self.status_label.pack()
 def on_click(self, event):
 for point, gps, name in self.toll_points:
 x1, y1, x2, y2 = self.canvas.coords(point)
 if x1 <= event.x <= x2 and y1 <= event.y <= y2:
 if point not in self.route_points:
 self.route_points.append(point)
 self.canvas.itemconfig(point, fill="green")
 self.status_label.config(text=f"Toll point {name} added to
route.")
 else:
 self.route_points.remove(point)
 self.canvas.itemconfig(point, fill="blue")
 self.status_label.config(text=f"Toll point {name} removed from
route.")
 break
 def on_motion(self, event):
 for point, gps, name in self.toll_points:
 x1, y1, x2, y2 = self.canvas.coords(point)
 if x1 <= event.x <= x2 and y1 <= event.y <= y2:
 self.hover_label.place(x=event.x + 10, y=event.y)
 self.hover_label.config(text=name)
 return
 self.hover_label.place_forget()
 def confirm_route(self, event):
 if len(self.route_points) < 2:
 messagebox.showwarning("Insufficient Points", "Please select at least
two toll points to calculate the toll.")
 return
```

```
 route_gps = [gps for point, gps, name in self.toll_points if point in
self.route_points]

 total_distance = 0.0
 for i in range(len(route_gps) - 1):
 total_distance += geodesic(route_gps[i], route_gps[i+1]).km

 toll = self.calculate_toll_cost(total_distance, self.vehicle_type)
 self.status_label.config(text=f"Total Distance: {total_distance:.2f} km.
Toll Cost: ₹{toll:.2f}.")
 # Proceed to payment gateway
 self.proceed_to_payment(toll)
 def calculate_toll_cost(self, distance, vehicle_type):
 rates = {
 'Car/Jeep/Van/LMV': 0.65,
 'Light Commercial Vehicle/LGV/Mini-Bus': 1.05,
 'Bus/Truck (2 axles)': 2.20,
 'Three-axle Commercial Vehicle': 2.40,
 'Heavy Construction Machinery/Multi-axle (4-6 axles)': 3.45,
 'Oversized Vehicle (7+ axles)': 4.20
 }
 toll_rate_per_km = rates.get(vehicle_type, 1.0)
 return distance * toll_rate_per_km
 def proceed_to_payment(self, toll):
 response = messagebox.askyesno("Proceed to Payment", f"The toll fare is
₹{toll:.2f}. Do you want to proceed to payment?")
 if response:
 # Simulate payment processing
 messagebox.showinfo("Payment Successful", "Payment has been processed
successfully!")
 else:
 messagebox.showinfo("Payment Cancelled", "Payment has been cancelled.")
if __name__ == "__main__":
 vehicle_selection = VehicleSelection()
 vehicle_selection.mainloop()
```

SNAPSHOTS OF THE OUTPUT SIMULATION UI



(a)before opening the drop down list *(b)after opening the drop down list*

Figure 2: The first window that appears when the simulation is run, a widow with the drop down list of categories/types of vehicles.

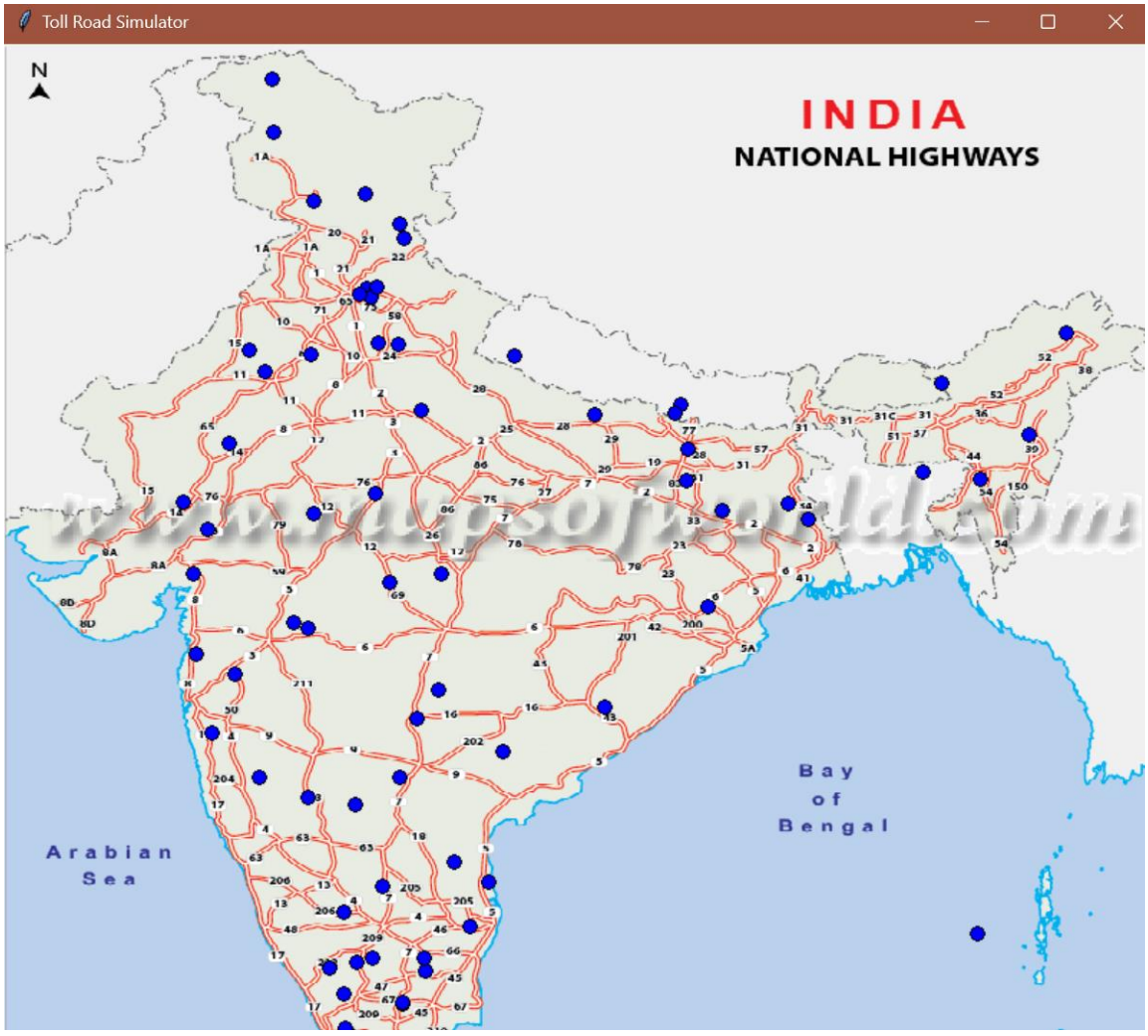


Figure 3: The Interactive Map window that appears after the type of vehicle is selected and confirmed by clicking onto "Next"

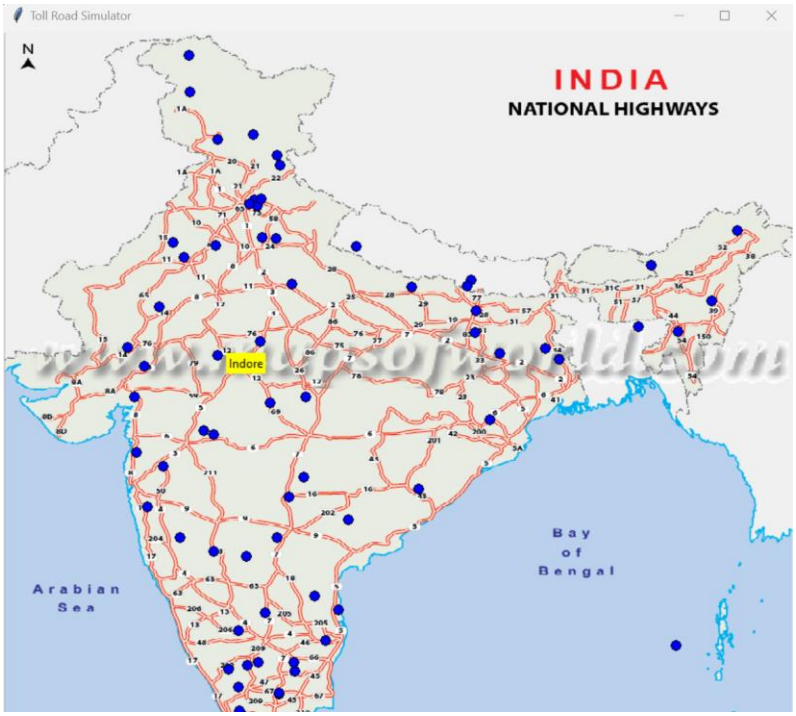


Figure 4: The Toll Gate label appearing over the interactive map when you hover the pointer over the toll points.

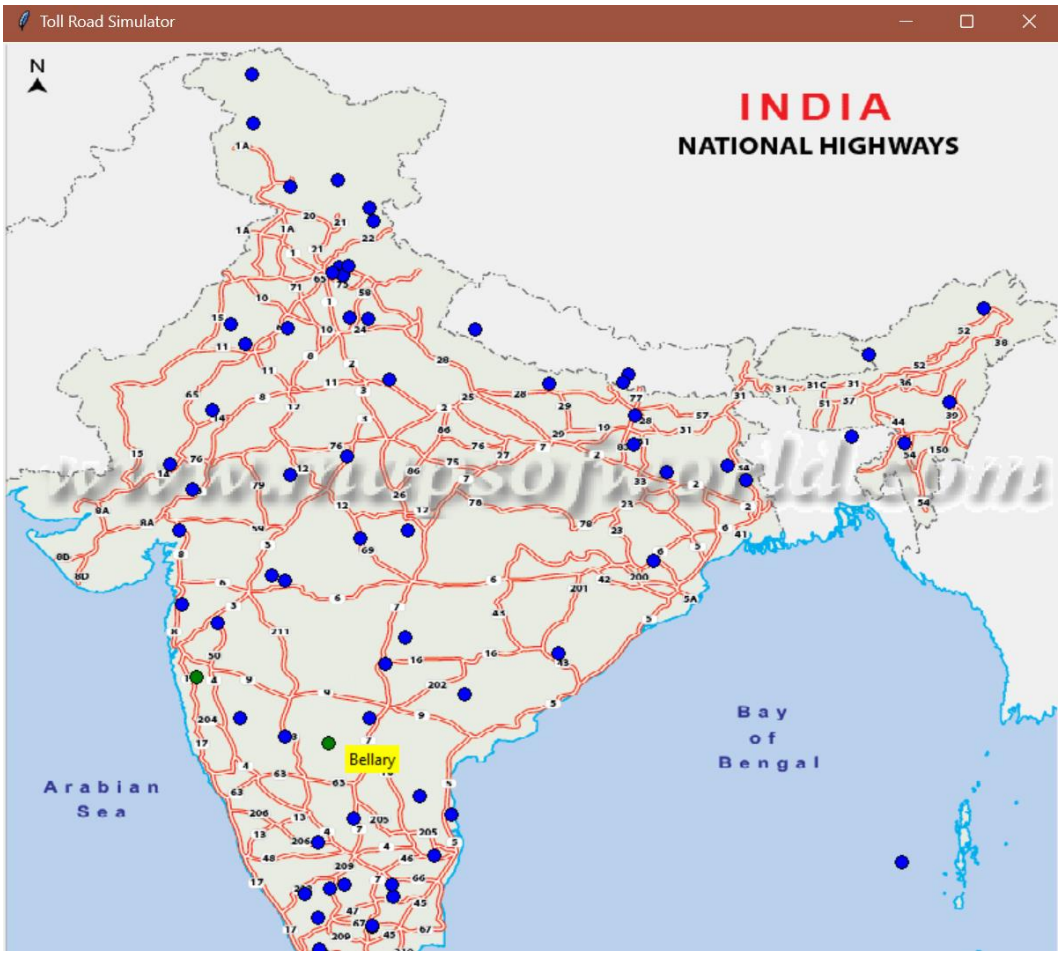


Figure 5: The Toll Points on the map turning green from blue after they are clicked to indicate that they are selected as the tolling points encountered along the travel, or as the starting and the destination locations.

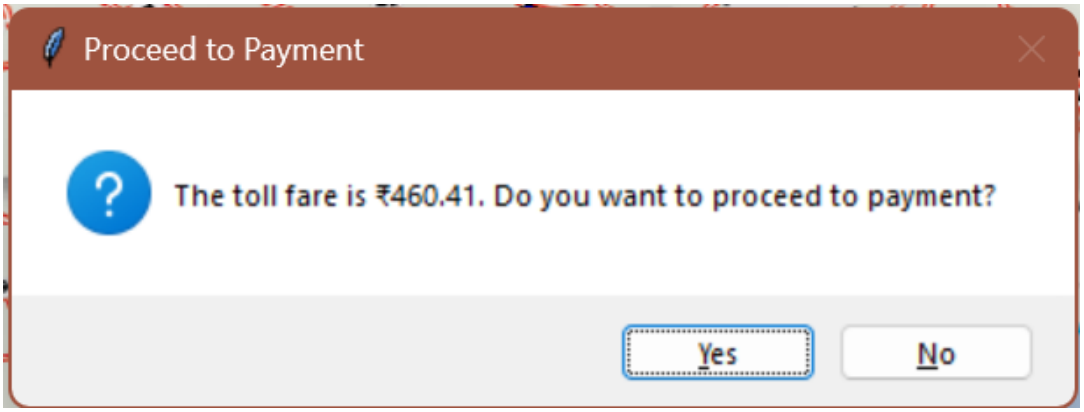


Figure 6: The Simulation UI Window that appears after the route/start-to-end locations are confirmed by pressing enter. It displays the total toll fare calculated and asks the user to proceed to payment.

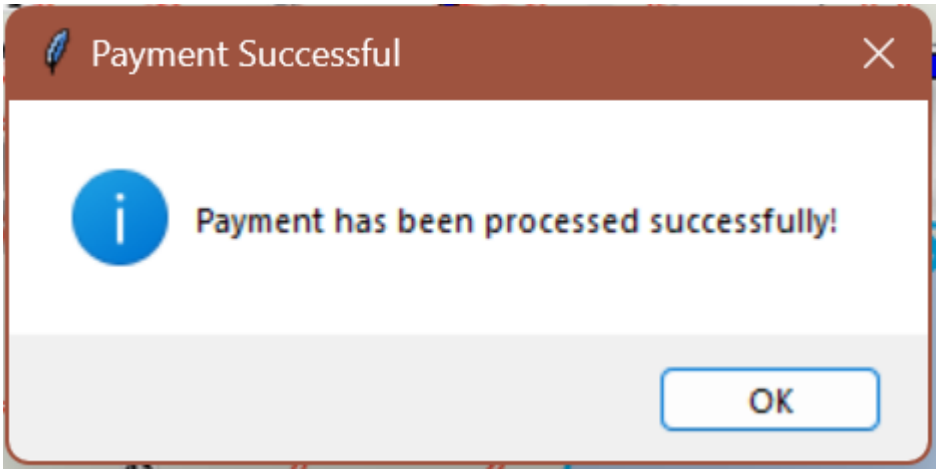


Figure 7: The final simulation UI window that appears once the user confirms their payment. After this, the simulation ends and exits from the system once the user presses "OK".

CONCLUSION

The GPS based Toll Road Simulator is a comprehensive solution designed to address the complexities of toll calculation and route planning for drivers. By integrating an interactive map interface, real-time toll calculation, vehicle type customization, and a simulated payment gateway, the application offers a user-friendly and efficient way to plan journeys on toll roads.

The use of Python and its supporting libraries ensures that the application is robust, scalable, and easy to maintain. The interactive features, such as toll point selection and hover information, provide a seamless user experience that simplifies the planning process.

The GPS based Toll Road Simulator not only enhances the user experience but also showcases the potential of combining geographical data with interactive applications to solve real-world problems. This project serves as a testament to the power of modern programming tools and libraries in creating innovative solutions that make everyday tasks easier and more efficient.

The GPS based Toll Road Simulator is a valuable tool for drivers, providing a practical solution to the often cumbersome task of toll calculation and route planning. Its intuitive design and comprehensive feature set make it an essential application for anyone navigating toll roads. As road networks continue to evolve, the Toll Road Simulator can be easily expanded and adapted, ensuring its relevance and usefulness for years to come.

THANK YOU!