

PROJECT REPORT OF AI

**TOPIC:** AI in Smart Cities

**TEACHER NAME: -** ANKITA WADHAWAN

**SUBJECT: -** ARTIFICIAL INTELLIGENCE (INT404) **SECTION: -** K21GP

**PROJECT DONE BY:**

|  |  |  |
| --- | --- | --- |
| Name | Reg. No. | Roll No. |
| Aaryan | 12104731 | 48 |
| Karanjot Singh | 11900827 | 20 |
| Ganji Shreyash Reddy | 12105213 | 44 |
| Aman Kumar | 12017280 | 68 |

**CONTENTS TO BE COVERED.**

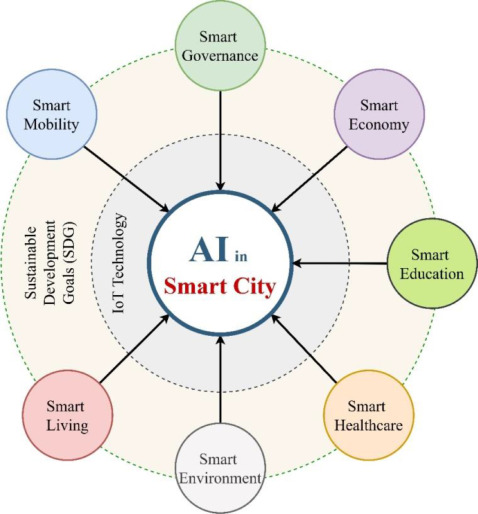
1. Introduction
2. Different applications of AI in smart cities
3. Impact of AI
4. Future Scope
5. Conclusion

# PROJECT DESCRIPTION

## OBJECTIVE

There has never been a greater demand for creative, effective solutions than there is today as cities all over the world get busier and more complicated. A response is coming in the form of "smart cities," which use cutting-edge technology to optimise infrastructure, services, and resources.

The Internet of Things (IoT) and artificial intelligence (AI) are the dynamic combo that define what makes a city really smart. (IoT). The way we live and engage with urban environments is changing as a result of this new technology combo.



**ABSTRACT**

Cities' population densities have been growing faster in recent years. In 2014, 3.3 billion people (54%) of the world's population lived in cities, according to the United Nations Population Fund. Around 5 billion people (68%) will live in cities by 2050. Cities must be clever and intelligent if they are to improve the comfort and cost-effectiveness of urban lifestyles. It is mostly performed through the use of computational intelligence-based technologies for intelligent decision-making. This study examined the usage of artificial intelligence (AI) in the idea of smart cities. We looked at 133 publications from 2014 to 2021 in the areas of healthcare, education, the environment and waste management, agriculture, mobility and smart transportation, risk management, and security, accounting for 97% of Scopus and 73% of WoS.

Furthermore, we found that the adoption of AI in smart cities is more significantly influenced by the energy (10%), mobility (19%), privacy and security (11%), and healthcare (23% impact) sectors. The healthcare sector has increased the use of AI-based innovations by 60% since the epidemic struck cities in 2019. The investigation shows that the influence of AI algorithms like ANN, RNN/LSTM, CNN/R-CNN, DNN, and SVM/LS-SVM on the various smart city domains is larger.

### WORK DISTRIBUTION

#### 1. AARYAN

* About uses of AI in smart cities.
* What are the applications of AI in smart cities.
* Coding of the program.

#### 2. KARANJOT SINGH

* Impact of AI in smart cities.
* What are the applications of AI in smart cities.
* Coding of Program

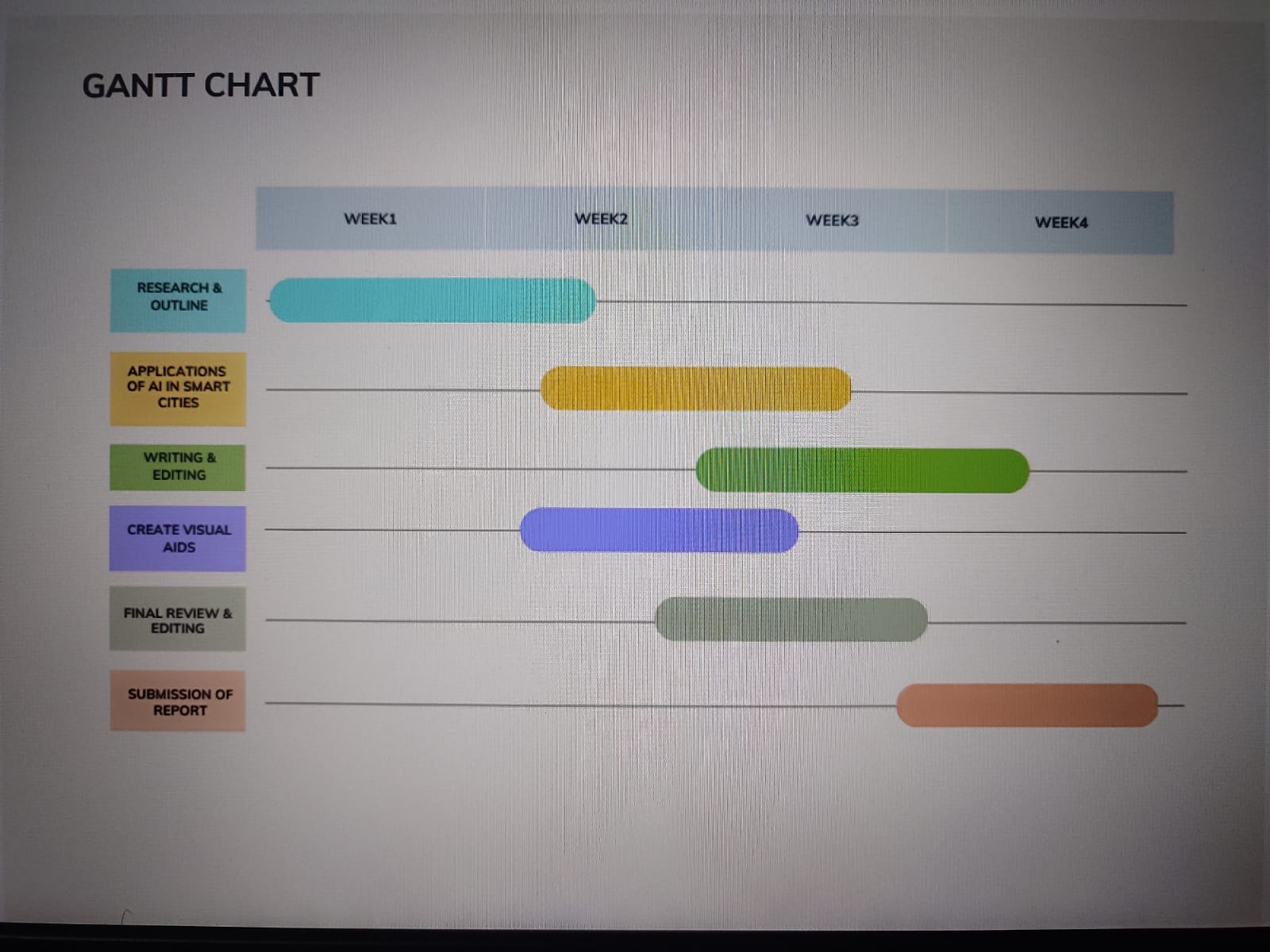
#### 3. GANJI SHREYASH REDDY

* Impact of AI in smart cities.
* Future scope in smart cities
* Coding of the program

**4. AMAN KUMAR**

* Future scope in smart cities
* Conclusion of the project
* Coding of the program.

**GANTT CHART**



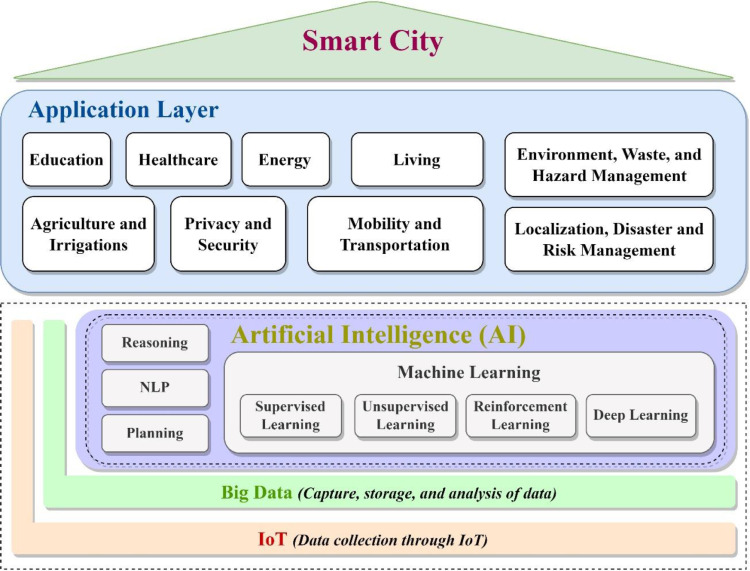
**AI IN SMART CITIES**

Smart City technologies need to process huge amounts of data, or "Big Data," in order to work. In terms of the three "Vs," big data has been defined as "high-volume, high-velocity and/or high-variety information assets", which refers to enormous datasets that are processed quickly (using algorithms) and that utilise various data sources, including mixing diverse datasets.

Artificial intelligence (AI) and big data are related. When a non-human technology is used to "learn from experience and imitate human intelligent behavior," this is referred to as artificial intelligence (AI). Large amounts of Big Data can be efficiently sorted through using AI to produce data forecasts and affordable solutions that power Smart City technologies.

Whether the AI is supervised or unsupervised affects how this functions. In supervised learning, target values and datasets are developed to train artificial intelligence (AI) networks to locate particular answers in the gathered raw data. The AI will then carry out predetermined tasks and actions while investigating fresh angles and prospects that might yield greater results than the status quo. Unsupervised learning involves training AI networks to uncover latent qualities and hidden patterns in the data by posing questions to them using non-labelled and non-classified datasets.

**APPLICATIONS OF AI IN SMART CITIES**



**1)** **SMART ENERGY METERING**

Real-time data on energy use may minimise wastage and loss, increase grid efficiency, optimise storage, and improve preventative infrastructure maintenance for public sector organisations like huge energy, power, and utility firms.

Smart metres are an AI and machine learning application that have great potential in the energy and utilities sector. In order for the government to realise its vision of smart cities and smart industrial zones, AI, ML, and the Internet of Things (IoT) are essential components.

Cities offer a variety of data that may be real-time collected with IoT devices, including energy use. Automatically adjusting power inputs can result in significant cost savings, more reliable supplies, and fewer outages.

Even on a lesser scale, smart metres are helpful. Customers can use them to customise their energy needs and cut costs. The information produced by the procedure could be utilised to create customised pricing and improve supply.

**2) IMPROVED COMMERCIAL CHATBOTS**

Chatbots are widely used. Talking about AI/ML application cases in any industry would be impossible without mentioning at least one "unique" chatbot. However, there are many smart cities out there.

No technician or technology specialist is required to understand how chatbots are already simplifying our lives by revolutionising how customers interact with brands. Customers can already use their favourite messaging service to communicate with brands on their own timetable, make purchases, or obtain the information they require.

Cities are growing "smarter" as practically every retailer or brand adopts a chatbot of some sort, but some are supposedly cleverer than others.

**3) INTELLIGENT SECURITY CAMERAS**

An easy way to use AI is to improve security cameras with computer vision. It's nice to have a video of a crime being committed, but unless someone is aware that something happened, there is simply no incentive to watch the film until it is eventually lost.

An AI-powered surveillance system that searches for illegal behaviour patterns is comparable to a squad of roving detectives that constantly review all available video.

Schools and companies can utilise AI-enhanced security cameras to speed up response times anytime an action is required. The AI can distinguish between people entering an area who match the description and give a warning in real time, for instance, if the person to be discovered is "a white male wearing a blue shirt."

As an alternative to having to sift through hours of film, his images and video can be submitted immediately to the area's first responders, who can use the keywords (white, male, and blue shirt in the example above) to discover clips of videos that might feature him.

**4) COMPUTER VISION AND PARKING SYSTEMS**

By combining many advanced functions such as license plate recognition and pixel detection, off-the-shelf cameras can provide real-time information on space availability to customers and parking operators, and automatically enforce parking payment and duration

Advanced algorithms can provide precise car position estimation and predict parking usage during certain times of the day or night

Eventually, this technology will be a perfect match for the upcoming [autonomous vehicles](https://www.techopedia.com/definition/30056/autonomous-car) which will be able to “speak” directly with parking lots and garages.

**6) REDUCING AIR POLLUTION**

AI has the potential to significantly reduce energy waste as well as air, water, and land pollution. According to estimates, the use of AI technologies might lower world GHG emissions by at least 4% by 2030.

IoT sensors and AI are being utilised in Singapore to collect and analyse data on the city's temperature, pollution levels, and air quality. By combining this data with AI, it may be possible to anticipate where air quality problems will arise in the future, allowing for the eventual mitigation of their impacts through efficient preventative actions.

The system can offer practical advice on how to lessen the consequences of the Chinese city's suffocating air pollution, such as temporarily shutting down some factories.

By examining data from coal-fuelled factories, industrial complexes, weather patterns, and traffic congestion, IBM researchers are putting

a novel AI technique to the test in an effort to lessen the severity of air pollution in Beijing.

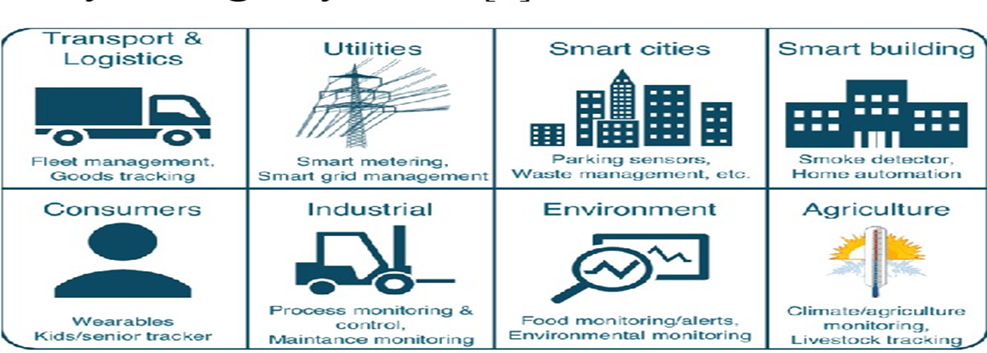
The system can offer practical advice on how to lessen the consequences of the Chinese city's suffocating air pollution, such as temporarily shutting down some factories.

**6) IMPROVING TRAFFIC CONTROL**

Another place where AI can help with the reduction of GHG emissions is transportation systems

Other than making autonomous cars possible soon, transportation can be made more sustainable even today.

Advanced traffic control is used to make cities considerably more liveable and can also aid in lowering air pollution. To improve air quality predictions, deep learning algorithms blend satellite photos with traffic data from cell phones and environmental IoT devices.

AI-based automatic licence plate reader (ALPR) software, like the one Rekor has implemented, can also be used to identify vehicles or for the instantaneous identification of infractions and crimes

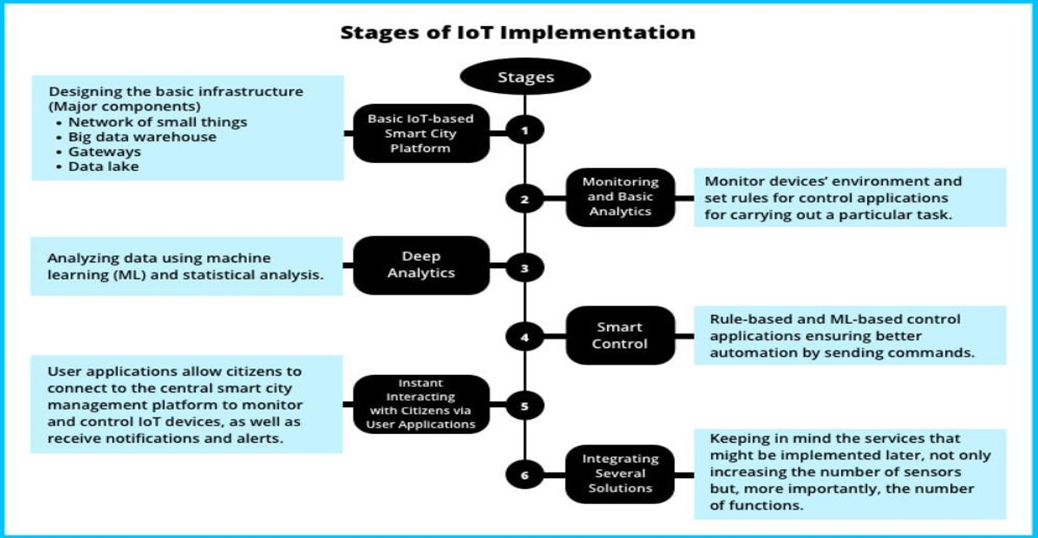
**7) COGNITIVE POWER PLANTS**

Even developing nations understand the importance of employing cutting-edge AI for a leg up on energy development. Beyond Limits is using a technology originated for NASA space missions to build the first cognitive plant in West Africa on behalf of Xcell, a Swiss-based global financial and minerals development company. The efficiency of power generation from natural gas will be amplified many times-fold. The cognitive AI will be embedded in every part of the new power plant from the very beginning of its construction. The efficiency and productivity of natural gas power plants have always been plagued by adverse environmental conditions, such as temperature and humidity. Beyond Limits’ AI is intelligent and aware enough to assist operators with its encoded expert-level human knowledge to make real-time adjustments. Instead of producing unneeded power in excess that cannot be stored, the software can constantly monitor the gas turbines and the entire system to prevent wasted production or lose profit because of the underproduction. Coupled with its ability to coordinate several interconnected production units at the same time, this futuristic power plant will be more efficient, productive, safe, and environmentally friendly.

**IMPACT OF AI IN SMART CITIES**

* Artificial intelligence (AI) has the potential to greatly impact smart cities by improving efficiency, reducing costs, and enhancing the quality of life for residents. Here are some of the ways in which AI is already making an impact in smart cities.
* Smart cities can use AI to see their effect on the local environment, global warming, as well as pollution levels. Using AI and machine learning for pollution control and energy consumption, allows authorities and cities to make well-informed decisions that are best for the environment.
* **3.1 Improved Traffic Management:** AI-powered traffic management systems can help reduce congestion, optimize traffic flow, and enhance road safety. For example, real-time traffic data can be collected from cameras and sensors, and analyzed to optimize traffic signal timings, reroute vehicles, and prevent accidents.
* AI-based traffic management systems can also be used to optimize the use of public transport systems, such as buses and trains. By analyzing data on passenger flows and traffic patterns, AI algorithms can be used to adjust the frequency and routing of public transport services to better meet the needs of commuters.
* Another approach to AI-based traffic management is to use predictive algorithms to anticipate traffic patterns and adjust traffic flow accordingly. For example, if an event is expected to draw a large number of people to a particular area, traffic management systems can be adjusted to accommodate this influx of traffic and prevent congestion.
* Overall, the use of AI in traffic management systems in smart cities has the potential to greatly improve the efficiency of urban transport systems, reduce traffic congestion and air pollution, and improve the overall quality of life for city residents.
* **3.2 Enhanced Energy Management:** AI can be used to optimize energy consumption in smart buildings and homes. For instance, smart thermostats can learn occupants' behavior and adjust heating and cooling settings to conserve energy. AI can also be used to manage renewable energy sources like solar and wind power.
* Through the use of real-time data, Enhanced Energy Management systems can predict energy demand and supply, identify energy inefficiencies, and optimize energy usage to reduce waste and lower carbon emissions. This can be done by implementing various strategies such as load shifting, demand response, and energy storage management.
* **3.3 Improved Public Safety:** AI can be used to enhance public safety by analyzing real- time data from surveillance cameras and sensors to detect potential threats, such as accidents, crimes, and fires. AI can also be used to alert emergency services and dispatch the appropriate response teams.
* AI can be used to analyze data from security cameras, sensors, and other sources to detect and
* prevent crime, and to respond quickly to emergencies.
* **Predictive Policing:** AI-powered predictive policing systems can help law enforcement agencies predict where crimes are likely to occur based on historical crime data and other factors such as weather, time of day, and local events. This can help prevent crime and enable police to respond more quickly and efficiently.
* **Emergency Response:** AI-powered emergency response systems can help emergency services respond more quickly and efficiently to emergencies such as natural disasters, accidents, and terrorist attacks. For example, AI can be used to analyze real-time data from social media, traffic cameras, and other sources to identify the location and severity of an emergency and dispatch the appropriate response team.
* **3.4 Enhanced Healthcare:** AI-powered healthcare systems can help improve patient outcomes and reduce healthcare costs. For example, AI algorithms can be used to analyze medical data and predict patient outcomes, enabling doctors to provide personalized treatment plans. AI can also be used to manage healthcare resources and optimize hospital operations.
* There are some more important points on how AI can enhance healthcare in smart cities:
* **Medical Diagnosis:** AI-powered medical diagnosis systems can help doctors diagnose diseases and provide personalized treatment plans. For example, AI algorithms can analyze medical data such as images and lab results to identify patterns and anomalies that may be missed by human doctors.
* **Remote Patient Monitoring:** AI-powered remote patient monitoring systems can help doctors monitor patients outside of the hospital, reducing healthcare costs and improving patient outcomes. These systems use sensors and wearables to monitor vital signs and health data, and AI algorithms can analyze the data to detect changes and alert doctors to potential health issues.
* **Drug Discovery:** AI can be used to accelerate drug discovery and development, reducing the time and cost of bringing new drugs to market. AI algorithms can analyze large datasets to identify potential drug candidates, predict their efficacy and toxicity, and recommend the best drug development strategies.
* **Healthcare Resource Management:** AI can be used to optimize healthcare resource management in smart cities, improving efficiency and reducing costs. For example, AI algorithms can analyze hospital data to predict patient flow, optimize staffing, and reduce wait times.
* **Personalized Medicine**: AI-powered personalized medicine systems can help doctors tailor treatments to individual patients based on their unique genetics, lifestyle, and medical history. For example, AI algorithms can analyze patient data to predict their response to different treatments and recommend the most effective course of action.
* **3.5 Improved Air Quality:** AI-powered air quality monitoring systems can help detect pollution levels and identify sources of pollution. For example, sensors can be installed throughout the city to monitor air quality, and AI algorithms can be used to analyze the data and provide real-time alerts and recommendations.

**FUTURE SCOPE**

* The integration of AI technology in smart cities is expected to revolutionize urban life by enhancing efficiency, sustainability, and safety. AI can help to optimize the use of resources, enable better decision-making, and improve citizen services. As cities continue to grow and become more complex ,
* 

AI technology will play an increasingly important role in managing urban environments.

* **4.1 Autonomous vehicles:** AI-powered Self-driving cars and other autonomous vehicles will play a key role in future smart cities, reducing traffic congestion and increasing safety. AI algorithms can help these vehicles to navigate complex urban environments, detect obstacles and avoid collisions.
* **4.2 Intelligent traffic management:** AI algorithms can be used to optimize traffic flow, reduce congestion, and improve safety on city streets. By analyzing data from traffic sensors and cameras, AI systems can adjust traffic lights and signage to ensure efficient and safe movement of vehicles.
* Intelligent traffic management involves using AI and other advanced technologies to optimize traffic flow and reduce congestion on city streets. This can help improve transportation efficiency, reduce emissions, and improve safety for drivers, pedestrians, and cyclists.There are some ways that AI can be used for intelligent traffic management
* **Real-time traffic monitoring:** AI algorithms can analyze data from traffic sensors, cameras, and other sources in real-time to monitor traffic flow and identify areas of congestion. This information can be used to optimize traffic signals and adjust traffic flow to reduce congestion.
* **Predictive traffic analysis:** AI can be used to predict traffic patterns and anticipate areas of congestion before they occur. This can help traffic management systems adjust traffic flow and minimize the impact of congestion on drivers and pedestrians.
* **Incident detection and response:** AI algorithms can monitor traffic data for incidents such as accidents or road closures, and provide real-time alerts to traffic management systems. This can help reduce response times and minimize the impact of incidents on traffic flow.
* **Dynamic speed limits:** AI can be used to adjust speed limits based on real-time traffic data, helping to reduce congestion and improve safety on city streets.

**CODE OF APPLICATION**

AI based Traffic Flow Management

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

// A struct to represent a vehicle

struct Vehicle {

    int id;

    int speed;

    int direction;

};

// A function to process real-time data and identify traffic patterns

vector<Vehicle> analyzeData(vector<Vehicle> data) {

    // Here, we can use machine learning algorithms or simple statistical methods to identify traffic patterns

    // For simplicity, we'll just sort the data based on speed and return the top 10 vehicles with the highest speed

    sort(data.begin(), data.end(), [](Vehicle a, Vehicle b) {

        return a.speed > b.speed;

    });

    return vector<Vehicle>(data.begin(), data.begin() + 10);

}

// A function to make traffic flow management decisions based on the analyzed data

void manageTrafficFlow(vector<Vehicle> analyzedData) {

    // Here, we can use decision-making algorithms to decide how to manage traffic flow

    // For simplicity, we'll just print the IDs of the top 10 fastest vehicles

    cout << "Top 10 fastest vehicles: ";

    for (Vehicle v : analyzedData) {

        cout << v.id << " ";

    }

    cout << endl;

}

int main() {

    // In a real program, we would collect real-time data from sensors or cameras

    // For simplicity, we'll just create some dummy data

    vector<Vehicle> data = {

        {1, 60, 0},

        {2, 50, 1},

        {3, 70, 0},

        {4, 40, 1},

        {5, 80, 0},

        {6, 45, 1},

        {7, 75, 0},

        {8, 55, 1},

        {9, 65, 0},

        {10, 30, 1},

        {11, 90, 0},

        {12, 35, 1},

        {13, 85, 0},

        {14, 25, 1},

        {15, 95, 0}

    };

    vector<Vehicle> analyzedData = analyzeData(data);

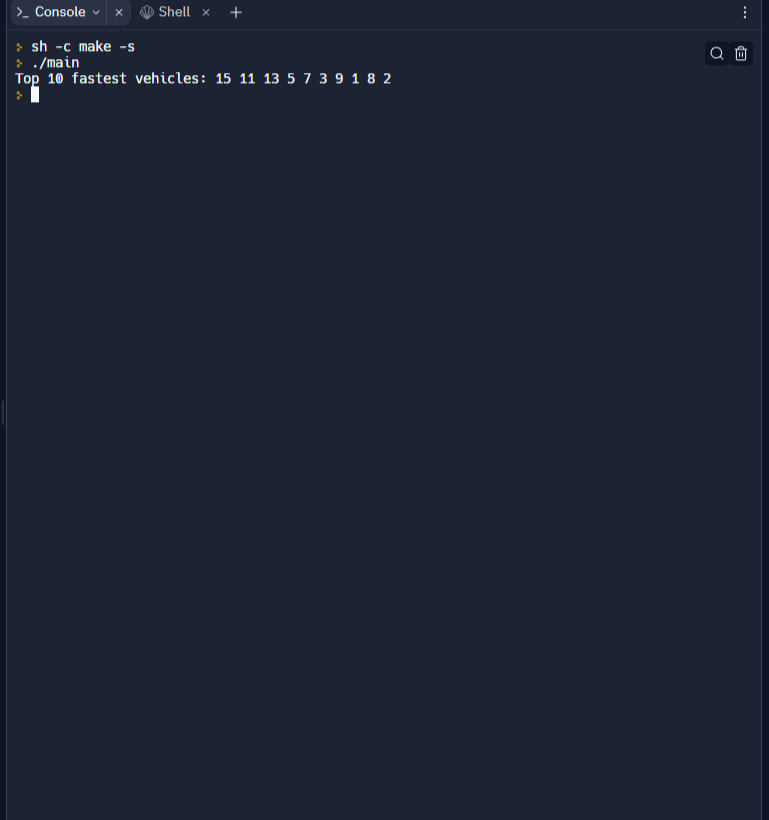
    manageTrafficFlow(analyzedData);

    return 0;

}

In this example, we simulate the collection of real-time data by creating a vector of dummy Vehicle objects. We then pass this data to the analyzeData function, which uses a simple sorting algorithm to identify the top 10 fastest vehicles. Finally, we pass the analyzed data to the manageTrafficFlow function, which simply prints the IDs of the top 10 fastest vehicles. In a real program, we would use more sophisticated algorithms and decision-making strategies.

**OUTPUT**

****

**CODE OF APPLICATION**

AI based Smart Parking Spots

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

// A struct to represent a parking spot

struct ParkingSpot {

    int id;

    bool occupied;

};

// A function to process real-time data and identify available parking spots

vector<ParkingSpot> analyzeData(vector<ParkingSpot> data) {

    // Here, we can use machine learning algorithms or simple statistical methods to identify available parking spots

    // For simplicity, we'll just randomly mark 5 parking spots as available

    random\_shuffle(data.begin(), data.end());

    for (int i = 0; i < 5; i++) {

        data[i].occupied = false;

    }

    return data;

}

// A function to reserve a parking spot

void reserveParkingSpot(vector<ParkingSpot>& data, int id) {

    // Here, we would update the status of the specified parking spot to "occupied"

    // For simplicity, we'll just print a message indicating that the spot has been reserved

    cout << "Parking spot " << id << " reserved." << endl;

}

int main() {

    // In a real program, we would collect real-time data from sensors or cameras

    // For simplicity, we'll just create some dummy data

    vector<ParkingSpot> data = {

        {1, true},

        {2, false},

        {3, true},

        {4, false},

        {5, true},

        {6, false},

        {7, true},

        {8, false},

        {9, true},

        {10, false},

        {11, true},

        {12, false},

        {13, true},

        {14, false},

        {15, true}

    };

    vector<ParkingSpot> analyzedData = analyzeData(data);

    for (ParkingSpot spot : analyzedData) {

        if (!spot.occupied) {

            reserveParkingSpot(analyzedData, spot.id);

            break;

        }

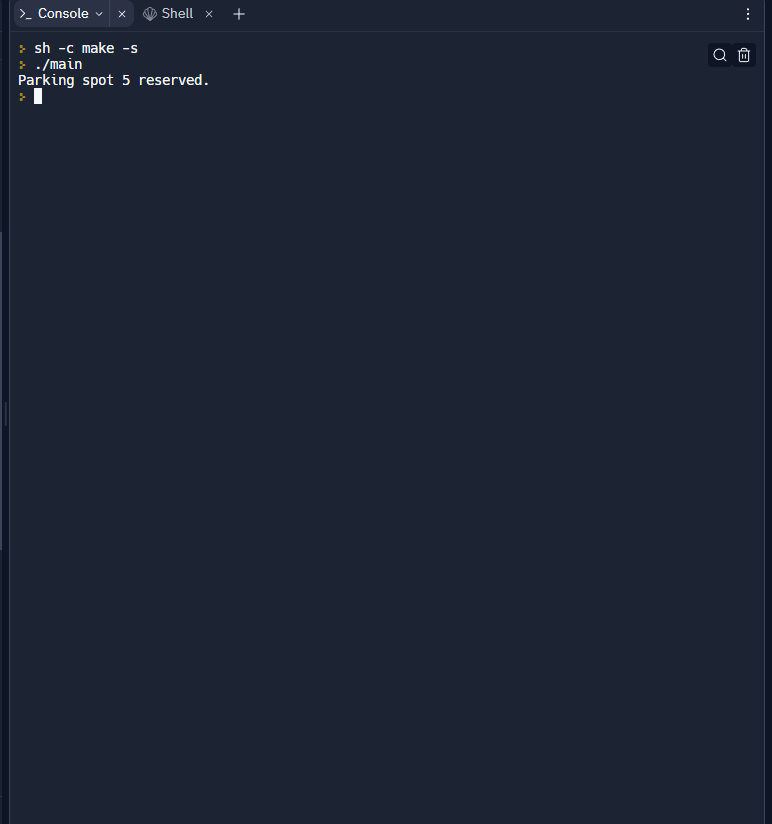
    }

    return 0;

}

In this example, we simulate the collection of real-time data by creating a vector of dummy ParkingSpot objects. We then pass this data to the analyzeData function, which randomly marks 5 parking spots as available. Finally, we iterate over the analyzed data to find the first available parking spot and reserve it by calling the reserveParkingSpot function. In a real program, we would use more sophisticated algorithms to identify available parking spots, and we would also need to update the status of the reserved parking spot to "occupied".

**OUTPUT**

****

**CODE OF APPLICATION**

AI based Cab Booking Service

from tkinter import \*

from tkinter import messagebox

from tkinter.ttk import \*

user\_names = ["Admin", ""]

passwords = {"Admin": "admin"}

options = [

    "None",

    "Phagwara to LPU",

    "Phagwara to Ramamandi",

    "Phagwara to Satnampura",

    "Phagwara to Ludhiana",

    "Phagwara to Phillaur",

    "Phagwara to Chandigarh",

    "Phagwara to Jalandhar",

    "Phagwara to Amritsar",

    "Phagwara to Deepnagar",

    "Phagwara to kirthinagar",

    "Phagwara to Law Gate",

    "Phagwara to Hargobindnagar"

]

rates = {

    "Phagwara to LPU": "100",

    "Phagwara to Ramamandi": "80",

    "Phagwara to Satnampura": "80",

    "Phagwara to Ludhiana": "150",

    "Phagwara to Phillaur": "120",

    "Phagwara to Chandigarh": "300",

    "Phagwara to Jalandhar": "200",

    "Phagwara to Amritsar": "300",

    "Phagwara to Deepnagar": "150",

    "Phagwara to kirthinagar": "80",

    "Phagwara to Law Gate": "120",

    "Phagwara to Hargobindnagar": "80",

}

root = Tk()

root.geometry("700x450+300+100")

root.title("Main Screen")

# for login

username1 = StringVar()

password1 = StringVar()

# for new user

gender\_box = IntVar()

username2 = StringVar()

password2 = StringVar()

name = StringVar()

mobile\_no = StringVar()

Email\_id = StringVar()

# for options

options\_var = StringVar()

mobile\_no\_entry\_text = StringVar()

day\_entry\_text = StringVar()

time\_entry\_text = StringVar()

def show\_rate(child1):

    a = str(options\_var.get())

    rate = rates[a]

    lbl\_fare\_1 = Label(child1, text=rate, font='Times 18 bold')

    lbl\_fare\_1.place(relx=.6, rely=.45, anchor=CENTER)

#------------------------------------for booking-----------------------------------------------

def show\_msg\_booking():

    mob = str(mobile\_no\_entry\_text.get())

    day = str(day\_entry\_text.get())

    time = str(time\_entry\_text.get())

    if mob != "" and day != "" and time != "":

        messagebox.showinfo("Booking Successful", "Booking Successful")

    else:

        messagebox.showerror("ERROR", "Mobile number or day or time is not filled")

#--------------------------------------create account---------------------------------------------

def create\_new\_user():

    user\_new = username2.get()

    pass\_new = password2.get()

    gender = gender\_box.get()

    name\_new = name.get()

    mob = mobile\_no.get()

    email\_new = Email\_id.get()

    if user\_new != "" and pass\_new != "" and (gender != 1 or gender != 2) and name\_new != "" and mob != "" \

            and email\_new != "":

        user\_names[1] = user\_new

        new\_user\_created = {user\_new: pass\_new}

        passwords.update(new\_user\_created)

        messagebox.showinfo("Successful", "New User Created successfully")

    else:

        messagebox.showerror("ERROR", "One or more field is empty")

def new\_user():

    root\_child\_1 = Toplevel(root)

    root\_child\_1.title("New User")

    root\_child\_1.geometry("700x450+300+100")

    lbl2 = Label(root\_child\_1, text="Register for New Account", font='Times 30 bold')

    lbl2.place(relx=.5, rely=.1, anchor=CENTER)

    lbl\_username = Label(root\_child\_1, text='Username :', font='Times 18 bold')

    lbl\_username.place(relx=.3, rely=.25, anchor=CENTER)

    username\_entry = Entry(root\_child\_1, textvariable=username2, width=35)

    username\_entry.place(relx=.6, rely=.25, anchor=CENTER)

    lbl\_password = Label(root\_child\_1, text='Password :', font='Times 18 bold')

    lbl\_password.place(relx=.3, rely=.35, anchor=CENTER)

    password\_entry = Entry(root\_child\_1, show="\*", textvariable=password2, width=35)

    password\_entry.place(relx=.6, rely=.35, anchor=CENTER)

    lbl\_name = Label(root\_child\_1, text='Name :', font='Times 18 bold')

    lbl\_name.place(relx=.3, rely=.45, anchor=CENTER)

    name\_entry = Entry(root\_child\_1, textvariable=name, width=35)

    name\_entry.place(relx=.6, rely=.45, anchor=CENTER)

    lbl\_gender = Label(root\_child\_1, text='Gender :', font='Times 18 bold')

    lbl\_gender.place(relx=.3, rely=.55, anchor=CENTER)

    gender1 = Radiobutton(root\_child\_1, text='Male', variable=gender\_box, value=1)

    gender1.place(relx=.5, rely=.55, anchor=CENTER)

    gender2 = Radiobutton(root\_child\_1, text='Female', variable=gender\_box, value=2)

    gender2.place(relx=.6, rely=.55, anchor=CENTER)

    lbl\_mobile = Label(root\_child\_1, text='Mobile no. :', font='Times 18 bold')

    lbl\_mobile.place(relx=.3, rely=.65, anchor=CENTER)

    mobile\_no\_entry = Entry(root\_child\_1, textvariable=mobile\_no, width=35)

    mobile\_no\_entry.place(relx=.6, rely=.65, anchor=CENTER)

    lbl\_email = Label(root\_child\_1, text='Email ID :', font='Times 18 bold')

    lbl\_email.place(relx=.3, rely=.75, anchor=CENTER)

    email\_entry = Entry(root\_child\_1, textvariable=Email\_id, width=35)

    email\_entry.place(relx=.6, rely=.75, anchor=CENTER)

    submit\_btn1 = Button(root\_child\_1, text='Create', command=lambda: [create\_new\_user()])

    submit\_btn1.place(relx=.6, rely=.9, anchor=CENTER, height=40, width=70)

    exit\_btn2 = Button(root\_child\_1, text='Exit', command=lambda: [root\_child\_1.destroy()])

    exit\_btn2.place(relx=.7, rely=.9, anchor=CENTER, height=40, width=70)

    root\_child\_1.mainloop()

#---------------------------------------------login----------------------------------------

def login\_button():

    user1 = username1.get()

    user1 = str(user1)

    pass1 = password1.get()

    pass1 = str(pass1)

    a = "1"

    for x in user\_names:

        if user1 == x:

            a = x

    if a == user1 and passwords[a] == pass1:

        root\_child\_1 = Toplevel(root)

        root\_child\_1.title("Booking Request")

        root\_child\_1.geometry("700x450+300+100")

        lbl5 = Label(root\_child\_1, text="Book your ride", font='Times 30 bold')

        lbl5.place(relx=.5, rely=.1, anchor=CENTER)

        mobile\_no\_booking = Label(root\_child\_1, text='Mobile no :', font='Times 18 bold')

        mobile\_no\_booking.place(relx=.3, rely=.25, anchor=CENTER)

        mobile\_no\_booking\_entry = Entry(root\_child\_1, textvariable=mobile\_no\_entry\_text, width=35)

        mobile\_no\_booking\_entry.place(relx=.6, rely=.25, anchor=CENTER)

        lbl\_routes = Label(root\_child\_1, text='Select Route :', font='Times 18 bold')

        lbl\_routes.place(relx=.3, rely=.35, anchor=CENTER)

        routes = OptionMenu(root\_child\_1, options\_var, \*options)

        routes.place(relx=.55, rely=.35, anchor=CENTER)

        rate\_btn = Button(root\_child\_1, text="Check fare", command=lambda: [show\_rate(root\_child\_1)])

        rate\_btn.place(relx=.75, rely=.35, anchor=CENTER)

        fare = Label(root\_child\_1, text='Fare in Rs. :', font='Times 18 bold')

        fare.place(relx=.3, rely=.45, anchor=CENTER)

        day = Label(root\_child\_1, text='Day :', font="Times 18 bold")

        day.place(relx=.3, rely=.55, anchor=CENTER)

        day\_entry = Entry(root\_child\_1, textvariable=day\_entry\_text, width=25)

        day\_entry.place(relx=.55, rely=.55, anchor=CENTER)

        day\_format = Label(root\_child\_1, text='Format dd/mm', font="Times 10 italic")

        day\_format.place(relx=.8, rely=.55, anchor=CENTER)

        time = Label(root\_child\_1, text='Time :', font="Times 18 bold")

        time.place(relx=.3, rely=.65, anchor=CENTER)

        time\_entry = Entry(root\_child\_1, textvariable=time\_entry\_text, width=25)

        time\_entry.place(relx=.55, rely=.65, anchor=CENTER)

        time\_format = Label(root\_child\_1, text='Format HH AM/PM', font="Times 10 italic")

        time\_format.place(relx=.8, rely=.65, anchor=CENTER)

        submit\_btn1 = Button(root\_child\_1, text='Book', command=lambda: [show\_msg\_booking()])

        submit\_btn1.place(relx=.6, rely=.75, anchor=CENTER, height=40, width=70)

        exit\_btn3 = Button(root\_child\_1, text='Exit', command=lambda: [root\_child\_1.destroy()])

        exit\_btn3.place(relx=.7, rely=.75, anchor=CENTER, height=40, width=70)

        root\_child\_1.mainloop()

    else:

        messagebox.showerror("ERROR", "Wrong Username or Password")

def login():

    root\_child\_1 = Toplevel(root)

    root\_child\_1.title("Login")

    root\_child\_1.geometry("700x450+300+100")

    lbl3 = Label(root\_child\_1, text="Please login to book a cab", font='Times 30 bold')

    lbl3.place(relx=.5, rely=.25, anchor=CENTER)

    lbl\_username = Label(root\_child\_1, text='Username :', font='Times 20 bold')

    lbl\_username.place(relx=.3, rely=.45, anchor=CENTER)

    username\_entry = Entry(root\_child\_1, textvariable=username1, width=35)

    username\_entry.place(relx=.6, rely=.45, anchor=CENTER)

    lbl\_password = Label(root\_child\_1, text='Password :', font='Times 20 bold')

    lbl\_password.place(relx=.3, rely=.55, anchor=CENTER)

    password\_entry = Entry(root\_child\_1, show="\*", textvariable=password1, width=35)

    password\_entry.place(relx=.6, rely=.55, anchor=CENTER)

    lbl\_new\_user = Label(root\_child\_1, text="Don't have account", font='Times 10 italic')

    lbl\_new\_user.place(relx=.3, rely=.65, anchor=CENTER)

    register\_btn = Button(root\_child\_1, text='Register here', command=lambda: [new\_user()])

    register\_btn.place(relx=.3, rely=.7, anchor=CENTER, height=25, width=100)

    submit\_btn1 = Button(root\_child\_1, text='Login', command=lambda: [login\_button()])

    submit\_btn1.place(relx=.6, rely=.7, anchor=CENTER, height=40, width=70)

    exit\_btn3 = Button(root\_child\_1, text='Exit', command=lambda: [root\_child\_1.destroy()])

    exit\_btn3.place(relx=.7, rely=.7, anchor=CENTER, height=40, width=70)

    root\_child\_1.mainloop()

#-------------------------------------------------available routes---------------------

def available\_route():

    root\_child\_1 = Toplevel(root)

    root\_child\_1.title("Available Routes")

    root\_child\_1.geometry("700x600+300+50")

    lbl4 = Label(root\_child\_1, text='Available Routes', font='Times 30 bold underline')

    lbl4.place(relx=.5, rely=.05, anchor=CENTER)

    lbl\_routes = Label(root\_child\_1, text='Routes', font='Times 25 bold')

    lbl\_routes.place(relx=.25, rely=.13, anchor=CENTER)

    lbl\_fare = Label(root\_child\_1, text='Fare in Rs.', font='Times 25 bold')

    lbl\_fare.place(relx=.75, rely=.13, anchor=CENTER)

    ja\_ph = Label(root\_child\_1, text='Phagwara to LPU', font='Times 19 bold')

    ja\_ph.place(relx=.25, rely=.2, anchor=CENTER)

    ja\_ph\_rate = Label(root\_child\_1, text='100', font='Times 19 bold')

    ja\_ph\_rate.place(relx=.75, rely=.2, anchor=CENTER)

    ja\_am = Label(root\_child\_1, text='Phagwara to Ramamandi', font='Times 19 bold')

    ja\_am.place(relx=.25, rely=.25, anchor=CENTER)

    ja\_am\_rate = Label(root\_child\_1, text='80', font='Times 19 bold')

    ja\_am\_rate.place(relx=.75, rely=.25, anchor=CENTER)

    ja\_lu = Label(root\_child\_1, text='Phagwara to Satnampura', font='Times 19 bold')

    ja\_lu.place(relx=.25, rely=.3, anchor=CENTER)

    ja\_lu\_rate = Label(root\_child\_1, text='80', font='Times 19 bold')

    ja\_lu\_rate.place(relx=.75, rely=.3, anchor=CENTER)

    ph\_ja = Label(root\_child\_1, text='Phagwara to Ludhiana', font='Times 19 bold')

    ph\_ja.place(relx=.25, rely=.4, anchor=CENTER)

    ph\_ja\_rate = Label(root\_child\_1, text='150', font='Times 19 bold')

    ph\_ja\_rate.place(relx=.75, rely=.4, anchor=CENTER)

    ph\_am = Label(root\_child\_1, text='Phagwara to Phillaur', font='Times 19 bold')

    ph\_am.place(relx=.25, rely=.45, anchor=CENTER)

    ph\_am\_rate = Label(root\_child\_1, text='120', font='Times 19 bold')

    ph\_am\_rate.place(relx=.75, rely=.45, anchor=CENTER)

    ph\_lu = Label(root\_child\_1, text='Phagwara to Chandigarh', font='Times 19 bold')

    ph\_lu.place(relx=.25, rely=.5, anchor=CENTER)

    ph\_lu\_rate = Label(root\_child\_1, text='300', font='Times 19 bold')

    ph\_lu\_rate.place(relx=.75, rely=.5, anchor=CENTER)

    am\_ja = Label(root\_child\_1, text='Phagwara to Jalandhar', font='Times 19 bold')

    am\_ja.place(relx=.25, rely=.6, anchor=CENTER)

    am\_ja\_rate = Label(root\_child\_1, text='200', font='Times 19 bold')

    am\_ja\_rate.place(relx=.75, rely=.6, anchor=CENTER)

    am\_ph = Label(root\_child\_1, text='Phagwara to Amritsar', font='Times 19 bold')

    am\_ph.place(relx=.25, rely=.65, anchor=CENTER)

    am\_ph\_rate = Label(root\_child\_1, text='300', font='Times 19 bold')

    am\_ph\_rate.place(relx=.75, rely=.65, anchor=CENTER)

    am\_lu = Label(root\_child\_1, text='Phagwara to Deepnagar', font='Times 19 bold')

    am\_lu.place(relx=.25, rely=.7, anchor=CENTER)

    am\_lu\_rate = Label(root\_child\_1, text='150', font='Times 19 bold')

    am\_lu\_rate.place(relx=.75, rely=.7, anchor=CENTER)

    lu\_ja = Label(root\_child\_1, text='Phagwara to kirthinagar', font='Times 19 bold')

    lu\_ja.place(relx=.25, rely=.8, anchor=CENTER)

    lu\_ja\_rate = Label(root\_child\_1, text='80', font='Times 19 bold')

    lu\_ja\_rate.place(relx=.75, rely=.8, anchor=CENTER)

    lu\_ph = Label(root\_child\_1, text='Phagwara to Law Gate', font='Times 19 bold')

    lu\_ph.place(relx=.25, rely=.85, anchor=CENTER)

    lu\_ph\_rate = Label(root\_child\_1, text='120', font='Times 19 bold')

    lu\_ph\_rate.place(relx=.75, rely=.85, anchor=CENTER)

    lu\_am = Label(root\_child\_1, text='Phagwara to Hargobindnagar', font='Times 19 bold')

    lu\_am.place(relx=.25, rely=.9, anchor=CENTER)

    lu\_am\_rate = Label(root\_child\_1, text='80', font='Times 19 bold')

    lu\_am\_rate.place(relx=.75, rely=.9, anchor=CENTER)

    exit\_btn4 = Button(root\_child\_1, text='Exit', command=lambda: [root\_child\_1.destroy()])

    exit\_btn4.place(relx=.9, rely=.95, anchor=CENTER, height=40, width=70)

    root\_child\_1.mainloop()

lbl1 = Label(root, text="Welcome to Cab Booking System", font='Times 30 bold')

lbl1.place(relx=.5, rely=.25, anchor=CENTER)

login\_btn = Button(root, text="Login", command=lambda: [login()])

login\_btn.place(relx=.3, rely=.5, anchor=CENTER, height=40, width=70)

new\_user\_btn = Button(root, text='New User', command=lambda: [new\_user()])

new\_user\_btn.place(relx=.5, rely=.5, anchor=CENTER, height=40, width=75)

avail\_routes\_btn = Button(root, text='Available Routes', command=lambda: [available\_route()])

avail\_routes\_btn.place(relx=.7, rely=.5, anchor=CENTER, height=40, width=100)

exit\_btn1 = Button(root, text='Exit', command=lambda: [root.destroy()])

exit\_btn1.place(relx=.5, rely=.7, anchor=CENTER, height=40, width=70)

mainloop()

**CONCLUSION**

Even developing nations are aware of how critical it is to use cutting-edge AI to advance the field of energy production.

Every day, AI becomes a more pervasive aspect of the future architecture of our cities. It has already shown to be one of the most beneficial tools that computer technology has provided to humanity and is assisting us in developing a smarter and more effective society.

With it, we are finally able to begin addressing the more significant problems that affect human civilization.