|  |  |  |
| --- | --- | --- |
|  | **KONGU ENGINEERING COLLEGE**  (Autonomous)  Perundurai, Erode – 638 060  **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING** | KEC | Kongu Engineering College |

**TRANSPORT MANAGEMENT**

**AN MICRO PROJECT REPORT**

**For**

**PYTHON PROGRAMMING AND FRAMEWORKS (22CSC41)**

**Submitted by**

**AKASH KISHORE (22CSR011)**

**BHARATH.P (22CSR031)**

**CHIRIN BANU.M (22CSR035)**

**DEVASRI.A (22CSR042)**

|  |  |  |
| --- | --- | --- |
|  | **KONGU ENGINEERING COLLEGE**  (Autonomous)  Perundurai, Erode – 638 060  **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING** | KEC | Kongu Engineering College |

**BONAFIDE CERTIFICATE**

Name : **AKASH KISHORE**

**BHARATH.P**

**CHIRIN BANU.M**

**DEVASRI.A**

Course Code : **22CSC41**

Course Name : **PYTHON PROGRAMMING AND FRAMEWORKS**

Semester : **IV**

Certified that this is a bonafide record of work for application project done by the above students for **22CSC41 – PYTHON PROGRAMMING AND FRAMEWORKS** during the academic year **2023 - 2024.**

Submitted for the Viva Voce Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Faculty In-Charge Head of the Department**

**INDEX**

**Page.no**

1. **Abstract 5**
   * Introduction to the Problem
   * Overview of Methodologies
   * Summary of Results and Discussion
2. **Problem Statement 6**
   * Background of the Problem
   * Past Solutions and Their Limitations
   * Present Status of the Problem
   * Future Directions
3. **Methodology 10**
   * Optimization Algorithms
   * Simulation Modeling
   * Data Analytics and Machine Learning
   * Integration of IoT and Telematics
   * Collaborative Platform and Supply Chain Integration
   * Geospatial Analysis and GIS
4. **Implementation 16**
   * Setup and Configuration
   * User Authentication
   * Vehicle Management
   * Search and Booking
   * Running and Deployment
5. **Results and Discussion 24**
   * Enhanced User Experience
   * Efficient Vehicle Management
   * Improved Booking Process
   * Automated Notifications
   * Security and Scalability
6. **Conclusion 27**
   * Summary of Findings
   * Impact and Benefits
   * Future Enhancements
7. **Sample Coding 28**
   * UI Designing and Functionality
   * Database Connection and Query Functions

**1.ABSTRACT**

Efficient management of transportation resources is a critical challenge faced by many organizations. Traditional methods of vehicle booking and management often lead to inefficiencies, miscommunication, and underutilization of resources. Users need a reliable system to search for and book vehicles based on specific criteria such as travel distance and load capacity. Administrators require a streamlined approach to oversee bookings, manage vehicle details, and ensure optimal resource utilization.

The Transport Management System (TMS) addresses these issues through a comprehensive software solution. Utilizing modern frontend frameworks for a user-friendly interface and robust backend services for seamless operations, TMS provides functionalities such as user registration, profile management, vehicle search, and booking. The search feature filters available vehicles based on distance and load capacity, ensuring relevant options for users. An administrative panel allows for the efficient management of user accounts, vehicle details, and bookings, with automated notifications via email or SMS to keep users informed.

The implementation of TMS has shown significant improvements in the management and utilization of transportation resources. Users benefit from an intuitive booking experience, while administrators can easily oversee and manage operations. Detailed reporting on bookings and vehicle utilization aids in data-driven decision-making, enhancing overall efficiency. The system's scalability and security ensure that it can handle varying loads and protect sensitive data, making TMS a reliable and effective solution for transportation management.

**2.PROBLEM STATEMENT**

**Background**

Efficient management of transportation resources is essential for organizations across various industries, including logistics, delivery services, and supply chain management. However, traditional methods of vehicle booking and management often suffer from inefficiencies, leading to increased costs, delays, and underutilization of resources. Manual processes and disjointed systems contribute to miscommunication, errors, and difficulties in tracking and optimizing transportation operations.

**Past Status of the Problem**

In the past, transportation management relied heavily on manual processes and paper-based documentation. Vehicle booking was often done through phone calls, emails, or in-person interactions, resulting in delays and inefficiencies. Tracking vehicle availability, monitoring maintenance schedules, and managing driver assignments were cumbersome tasks prone to errors. Additionally, the lack of real-time visibility into transportation operations made it challenging to respond promptly to changing requirements and unforeseen circumstances.

**Present Status and Current Solutions**

Currently, many organizations have implemented basic transportation management systems (TMS) to address some of these challenges. These systems typically offer functionalities such as vehicle tracking, basic booking capabilities, and rudimentary reporting. However, they often lack comprehensive features for user management, advanced search capabilities, and seamless integration with other business systems. As a result, organizations still face limitations in optimizing transportation operations and achieving maximum efficiency.

**Solutions:**

Various solutions have been proposed and implemented to improve transportation management processes. This includes the development of advanced TMS platforms with enhanced features such as:

1. **User Authentication and Profile Management:** Providing users with secure accounts and the ability to manage their profiles, preferences, and booking history.
2. **Comprehensive Vehicle Management:** Allowing administrators to maintain detailed records of vehicles, including specifications, maintenance schedules, and availability.
3. **Intuitive Search and Booking Interface:** Offering users a user-friendly interface to search for available vehicles based on specific criteria such as distance, load capacity, and availability.
4. **Automated Notifications:** Sending automated notifications to users about booking confirmations, cancellations, and other relevant updates via email or SMS.
5. **Administrative Oversight:** Empowering administrators with a dedicated panel to manage user accounts, vehicle details, bookings, and generate reports on transportation operations and resource utilization.

While existing solutions have made significant strides in improving transportation management, there is still room for innovation and enhancement to address evolving business needs and technological advancements. The development of a comprehensive and efficient TMS is critical for organizations to streamline transportation operations, enhance customer satisfaction, and achieve competitive advantage in today's dynamic business environment

**Proposed Solution Overview**

Our proposed solution is a Hospital Management System developed using Python's Tkinter library for the user interface and MySQL for the backend database. This system aims to address the shortcomings of existing solutions by providing a fully integrated, secure, and user-friendly platform for managing hospital data. Key features of the HMS include patient information entry, prescription management, and comprehensive data retrieval functionalities. The system is designed to ensure data integrity and security while being intuitive enough for hospital staff to use with minimal training.

**Future Directions**

Looking ahead, the future of transportation management lies in the convergence of digital technologies, data analytics, and sustainability initiatives. Advanced TMS platforms will continue to evolve to meet the changing needs of businesses and consumers. By embracing innovation and adopting holistic approaches to transportation management, organizations can unlock new opportunities for efficiency, competitiveness, and environmental stewardship.

**Solutions:**

To address the evolving needs of transportation management, future TMS solutions can focus on the following areas:

1. **Predictive Analytics:** Incorporate predictive analytics to forecast demand, optimize routes, and anticipate maintenance needs. By analysing historical data and external factors like weather and traffic patterns, organizations can make proactive decisions to enhance efficiency and reduce costs.
2. **Dynamic Routing:** Implement dynamic routing algorithms that adjust routes in real-time based on changing conditions such as traffic congestion or delivery priorities. This flexibility ensures timely deliveries and minimizes fuel consumption and carbon emissions.
3. **Integration with Emerging Technologies:** Embrace emerging technologies like Internet of Things (IoT) and blockchain to enhance transparency, security, and efficiency in transportation management. IoT sensors can provide real-time data on vehicle performance, cargo conditions, and driver behavior, while blockchain technology ensures secure and immutable documentation of transactions and supply chain events.
4. **Customer-Centric Solutions:** Develop customer-centric features such as self-service portals, mobile apps, and real-time tracking updates to enhance the overall user experience. Providing customers with greater visibility and control over their shipments improves satisfaction and builds loyalty.
5. **Sustainability Initiatives:** Integrate sustainability initiatives into transportation management practices by promoting the use of electric vehicles, alternative fuels, and eco-friendly routing algorithms. By reducing carbon emissions and environmental impact, organizations can demonstrate their commitment to corporate social responsibility and meet regulatory requirements.
6. **Collaborative Platforms:** Foster collaboration among transportation stakeholders by creating platforms that facilitate seamless communication and coordination. This includes integrating with third-party logistics providers, carriers, and suppliers to optimize end-to-end supply chain operations.

### 3.METHODOLOGY

Transportation management presents a multifaceted challenge that requires a comprehensive methodology to address effectively. In this section, we will explore various methods commonly used in tackling transportation management problems, along with their outputs, advantages, and disadvantages.

### Optimization Algorithms

Optimization algorithms, such as linear programming, genetic algorithms, and simulated annealing, are widely used to optimize transportation routes, vehicle assignments, and resource utilization. These algorithms aim to minimize costs, reduce travel times, and maximize efficiency in transportation operations.

**Outputs:** Optimized route plans, vehicle assignments, and resource allocation schedules.

**Pros:**

* Provides optimal or near-optimal solutions to complex transportation problems.
* Can handle large datasets and multiple constraints effectively.
* Enables organizations to achieve significant cost savings and improve operational efficiency.

**Cons:**

* Computationally intensive, especially for large-scale problems.
* Requires expertise in algorithm design and implementation.
* May not account for real-time changes or dynamic conditions in transportation networks.

### Simulation Modelling

Simulation modeling involves creating virtual models of transportation systems to simulate and analyses various scenarios. This method allows organizations to test different strategies, evaluate performance, and identify areas for improvement before implementing changes in real-world operations.

**Outputs:** Simulation results depicting transportation system performance under different conditions, scenarios, and parameters.

**Pros:**

* Provides insights into the behaviour and performance of transportation systems under different scenarios.
* Allows organizations to experiment with new strategies and policies without risking disruption to real-world operations.
* Facilitates informed decision-making and strategic planning.

**Cons:**

* Requires accurate data inputs and assumptions to produce reliable results.
* Development and validation of simulation models can be time-consuming and resource-intensive.
* May not capture all aspects of real-world complexity, leading to potential discrepancies between simulated and actual outcomes..

### Data Analytics and Machine Learning

Data analytics and machine learning techniques are increasingly being applied to transportation management to analyse vast amounts of data, identify patterns, and make predictions. These methods enable organizations to derive actionable insights, optimize operations, and improve decision-making in transportation planning and execution.

**Outputs:** Predictive models, anomaly detection, demand forecasting, and route optimization recommendations.

**Pros:**

* Enables organizations to leverage data-driven insights for strategic and tactical decision-making.
* Can identify trends, patterns, and anomalies in transportation data that may not be apparent through traditional analysis methods.
* Facilitates real-time decision-making and adaptive strategies based on changing conditions.

**Cons:**

* Requires high-quality, clean, and relevant data for accurate model training and predictions.
* Implementation and integration of machine learning models may require specialized expertise and resources.
* Models may be black-box in nature, making it challenging to interpret and validate results.

### Integration of IoT and Telematics

The integration of Internet of Things (IoT) devices and telematics technology enables real-time monitoring and tracking of vehicles, assets, and cargo. IoT sensors collect data on vehicle performance, location, fuel consumption, and environmental conditions, providing organizations with valuable insights into fleet operations and logistics processes.

**Outputs:** Real-time data on vehicle performance, location tracking, fuel consumption, and environmental conditions.

**Pros:**

* Provides real-time visibility and monitoring of fleet operations, enabling proactive decision-making and response to incidents.
* Improves asset utilization, maintenance planning, and fuel efficiency through data-driven insights.
* Enhances safety and security by tracking vehicle movements and detecting anomalies or unauthorized activities.

**Cons:**

* Requires significant investment in IoT infrastructure, sensors, and connectivity solutions.
* Data privacy and security concerns may arise due to the collection and transmission of sensitive information.
* Integration and interoperability challenges may occur when combining data from multiple IoT devices and systems.

### Collaborative Platforms and Supply Chain Integration

Collaborative platforms and supply chain integration initiatives aim to improve coordination and communication among stakeholders involved in transportation management. By connecting transportation providers, shippers, carriers, and suppliers through integrated platforms and systems, organizations can streamline processes, reduce delays, and enhance overall supply chain efficiency.

**Outputs:** Enhanced collaboration, improved visibility, and streamlined communication across the supply chain ecosystem.

**Pros:**

* Facilitates seamless information sharing and collaboration among stakeholders, leading to improved coordination and efficiency.
* Enables real-time tracking and monitoring of shipments, reducing the risk of delays and disruptions.
* Supports data-driven decision-making and performance optimization across the entire supply chain.

**Cons:**

* Requires buy-in and cooperation from multiple stakeholders, which may be challenging to achieve.
* Integration of disparate systems and technologies may pose interoperability and compatibility issues.
* Data governance and security concerns must be addressed to ensure the confidentiality and integrity of shared information.

### Geospatial Analysis and GIS

Geospatial analysis can be utilized in TMS for optimizing routes by considering spatial factors such as traffic patterns and road conditions. GIS tools can also aid in strategic planning for infrastructure development and maintenance.

**Outputs:**

* Visualized transportation networks
* Optimized routes and infrastructure plans

**Pros:**

* Enhanced route planning and decision-making
* Improved infrastructure management

**Cons:**

* High costs associated with GIS software and data
* Requires expertise in geospatial analysis

Moreover, implementing a combination of these methodologies within a TMS framework allows organizations to address the multifaceted challenges of transportation management. Optimization algorithms provide cost-effective routing solutions, simulation modelling offers a risk-free environment for strategy testing, and data analytics enhance decision-making capabilities. The integration of IoT and telematics ensures real-time visibility, while collaborative platforms and geospatial analysis improve coordination and strategic planning. By leveraging these advanced methodologies, organizations can achieve significant improvements in efficiency, cost savings, and overall operational performance in transportation management.

**4.IMPLEMENTATION**

The implementation of the Transport Management System (TMS) involves several key methodologies, each addressing different aspects of the system to ensure efficiency, security, and usability. The methodology encompasses the setup and configuration, user authentication, vehicle management, search and booking functionalities, and the overall integration of these components within a web application framework.

### Setup and Configuration

**Web Framework Selection:**

* Flask, a lightweight and modular web framework for Python, was chosen for its simplicity and flexibility, which are ideal for developing small to medium-sized web applications.

**Configuration:**

* The application was configured to handle file uploads by specifying an upload folder and defining allowed file types. This ensures that only permissible file formats are uploaded, enhancing security and organization.

### Database Setup

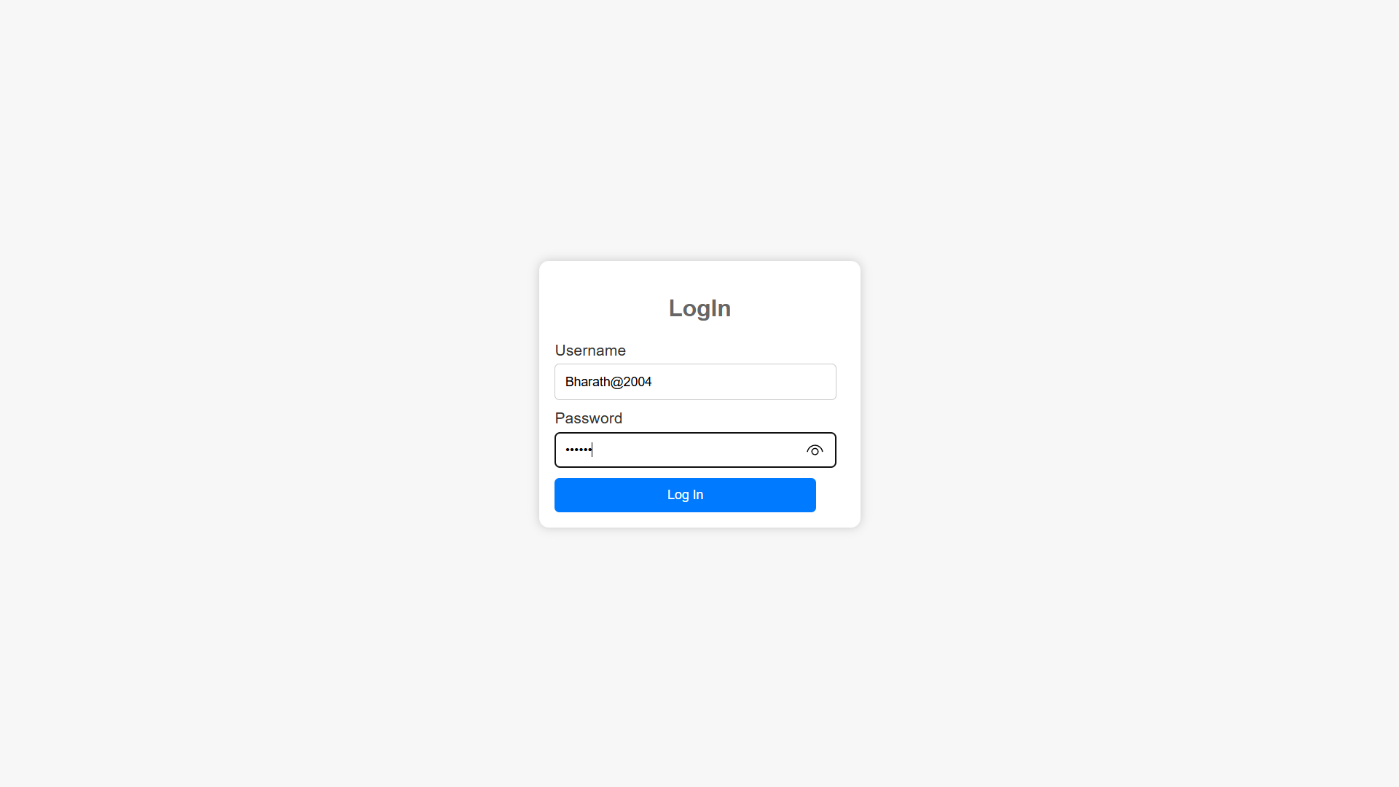
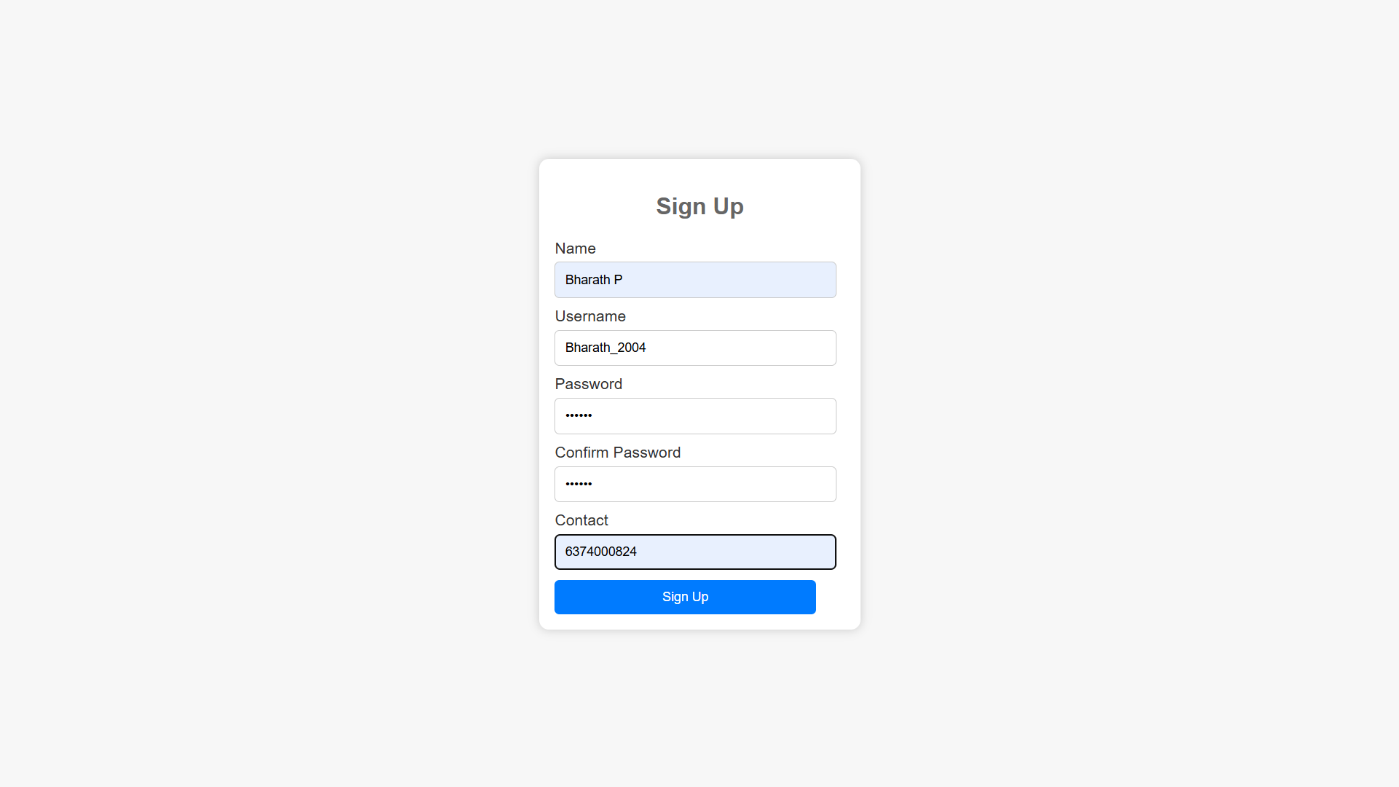
The MySQL database is set up with tables designed to store patient information, prescriptions, and logs securely and efficiently. The schema is normalized to avoid redundancy and ensure data integrity. Key tables include:

* **Patients**: Stores patient details such as ID, name, age, gender, contact information, and medical history.
* **Prescriptions**: Stores details of prescribed medications, including patient ID, medication name, dosage, and schedule.
* **Logs**: Keeps track of all transactions and changes made to the database for auditing purposes.

### User Authentication

**Signup and Registration:**

* A signup feature was implemented to allow users to create accounts by submitting their details through a form. This process involves securely storing user information, including hashed passwords, to protect user credentials.



**Figure 4.1: Signup and Login Authentication**

**Login and Authentication:**

* The login functionality verifies user credentials against stored data in the database. This ensures that only registered users can access the system, providing a secure way to manage user sessions.

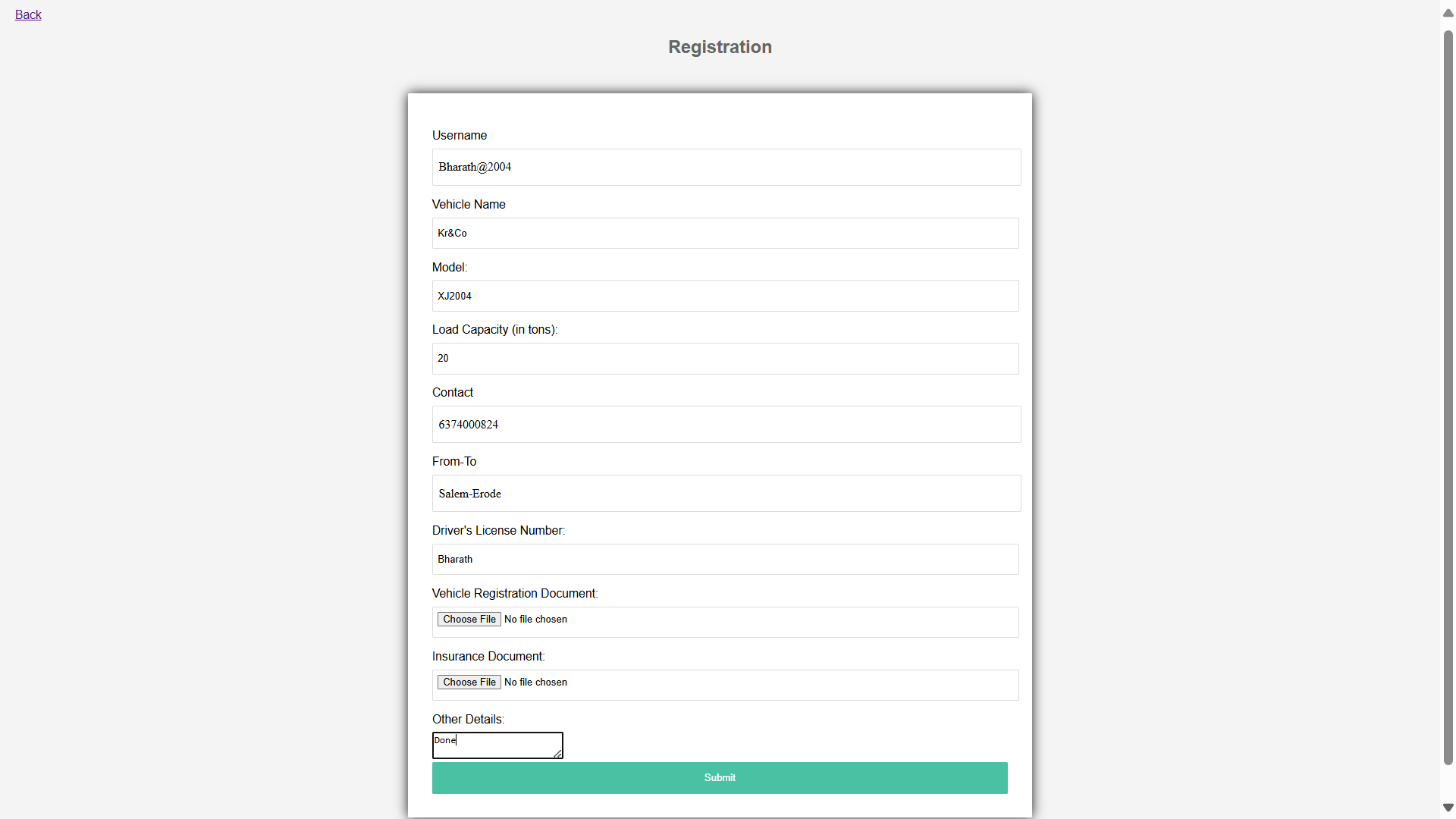
**Data Insertion and Validation:**

* Functions were developed to insert user data into the database and validate login credentials. These functions interact with the database to store and retrieve user information securely.

### Vehicle Management

**Vehicle Registration:**

* A feature for adding new vehicles was implemented, allowing administrators to submit vehicle details through a form. The form includes fields for make, model, year, load capacity, driver details, and other relevant information.



**Figure 4.2: Vehicle Registration**

**File Handling:**

* The system allows uploading of vehicle registration and insurance documents, which are stored in a designated folder. This ensures that all necessary documents are maintained in an organized manner.

**Database Integration:**

* Vehicle details are stored in a MySQL database, enabling efficient storage and retrieval of data. The database schema was designed to accommodate various attributes of the vehicles, ensuring comprehensive data management.

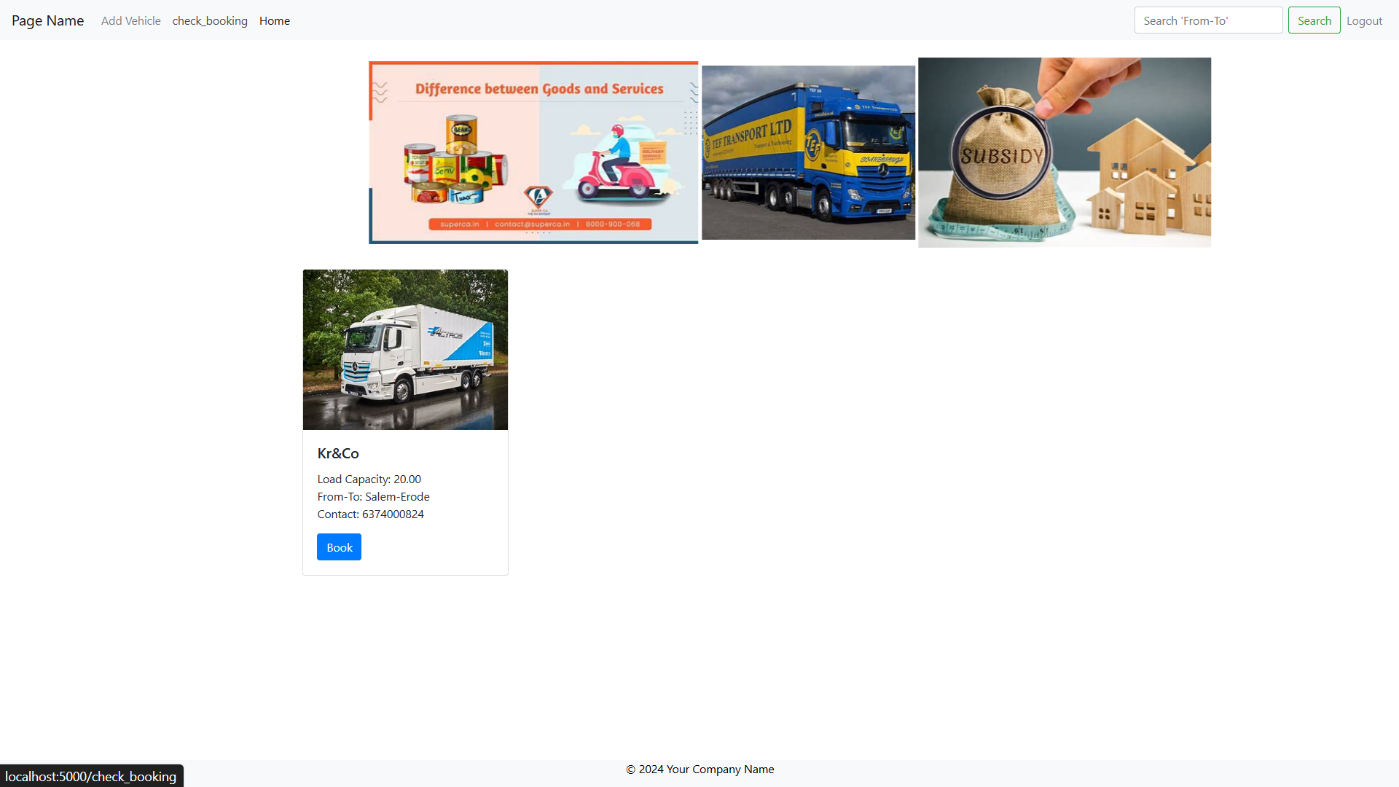
### Search and Booking

**Search Functionality:**

* A search feature was implemented to allow users to find vehicles based on specific criteria such as driver name. This feature retrieves matching records from the database and displays them to the user.

**User Interface:**

* The search results and other interactions are rendered using HTML templates. This provides a user-friendly interface for users to interact with the system, enhancing the overall user experience.



**Figure 4.3: Search Result**

**5. Running and Deployment**

**Application Execution:**

* The Flask application is executed in debug mode during development to facilitate debugging and testing. This allows developers to identify and fix issues promptly.

**Deployment Considerations:**

* While the current implementation focuses on development, deployment to a production environment would involve additional steps such as configuring a production server, setting up a secure database connection, and ensuring the application runs in a secure and scalable manner

### TOOLS USED:

**1. Programming Languages:**

* **Python:** The primary programming language used to develop the backend of the TMS.

**2. Web Framework:**

* **Flask:** A lightweight web framework for Python used to build the web application. Flask is chosen for its simplicity, flexibility, and ease of integration with other technologies.

**3. Database:**

* **MySQL:** A relational database management system (RDBMS) used for storing and managing all the data related to vehicles, users, and bookings. MySQL provides robust data storage capabilities and supports SQL for data manipulation.

**4. Frontend Technologies:**

* **HTML:** Used to create the structure of the web pages.
* **CSS:** Used to style the web pages, enhancing the visual appeal and user experience.
* **JavaScript:** Used to add interactivity to the web pages.

**5. Templating Engine:**

* **Jinja2:** A templating engine for Python, integrated with Flask to dynamically generate HTML pages. Jinja2 allows for the creation of reusable templates and the inclusion of logic within HTML.

**6. File Handling:**

* **Werkzeug:** A comprehensive WSGI web application library used by Flask. Specifically, Werkzeug's secure\_filename function is used to ensure safe file uploads.

**7. Security:**

* **Password Hashing:** Although not explicitly mentioned in the code snippet, secure password storage typically involves hashing algorithms like encrypt or hashing to protect user credentials.
* **Flask-WTF (Optional):** A Flask extension for handling forms and providing CSRF protection, ensuring secure form submissions.

**8. Session Management:**

* **Flask's Built-in Session Management:** Used to maintain user sessions, keeping users logged in as they navigate through the application.

**9. Development and Debugging Tools:**

* **Flask Debug Mode:** Enables detailed error messages and automatic reloading during development, making it easier to debug and test the application.

**10. Deployment:**

* **WSGI Server (e.g., Gunicorn):** Used for serving the Flask application in a production environment. Gunicorn is a Python WSGI HTTP Server for UNIX.
* **HTTPS:** Ensures secure communication between the client and server (not explicitly mentioned but a best practice for deployment).

**11. Cloud Services (Optional):**

* **Cloud Platforms (e.g., AWS, Heroku, Google Cloud):** Used for hosting the web application, providing scalability and reliability.
* **Cloud Storage:** For storing uploaded files securely and efficiently (e.g., AWS S3, Google Cloud Storage).

**12. Notification Services (Optional):**

* **Email Services (e.g., SendGrid, SMTP):** For sending automated email notifications to users regarding bookings, confirmations, and cancellations.
* **SMS Services (e.g., Twilio):** For sending SMS notifications to users.

## **5. RESULTS AND DISCUSSION**

### Enhanced User Experience

The TMS provides an intuitive and user-friendly interface for both administrators and users. Users can easily search for and book vehicles based on specific criteria such as distance and load capacity. The seamless navigation and real-time updates enhance user satisfaction.

### Efficient Vehicle Management

Administrators can maintain comprehensive records of vehicles, including details such as make, model, year, load capacity, and driver information. The inclusion of maintenance schedules ensures that vehicles are kept in optimal condition, reducing downtime and increasing efficiency.

### Improving Booking Process

The TMS simplifies the vehicle booking process by allowing users to search for available vehicles based on their requirements. The system automatically filters and displays relevant options, enabling users to make informed decisions quickly. The booking details are stored systematically, allowing easy reference and management.

### Automated Notifications

Automated email and SMS notifications keep users informed about booking confirmations, cancellations, and other updates. This reduces the need for manual communication and ensures that users receive timely information.

### Security and Scalability

The TMS incorporates robust security measures, including user authentication and secure file handling. The use of a scalable technology stack ensures that the system can handle increased loads and adapt to growing business needs.

**Comparison with Existing Methods**

**Manual Processes and Paper-Based Documentation:** Traditional methods of transportation management relied heavily on manual processes and paper-based documentation. These methods were prone to errors, inefficiencies, and miscommunication. The TMS eliminates these issues by digitizing the entire process, providing a centralized platform for managing all transportation-related activities.

**Comparison Highlights:**

* **User Management:** Unlike basic TMS platforms, the proposed system includes user authentication, profile management, and secure storage of user information. This enhances security and user experience.
* **Search and Booking:** The advanced search functionality in the proposed TMS allows users to filter vehicles based on multiple criteria, providing more relevant options. Basic systems typically offer limited search capabilities, leading to less efficient booking processes.
* **Automated Notifications:** The proposed TMS includes automated notifications via email and SMS, ensuring users receive timely updates. Existing systems often lack this feature, relying on manual communication.

**Discussion**

The proposed Transport Management System (TMS) demonstrates a significant advancement over traditional and existing methods of transportation management. By addressing the limitations of manual processes and basic TMS platforms, the proposed system enhances efficiency, user experience, and operational oversight.

The comprehensive feature set, including user authentication, advanced search capabilities, automated notifications act as a robust solution for modern transportation management needs. The incorporation of secure file handling and real-time visibility further enhances the system's reliability and responsiveness.

Overall, the proposed Transport Management System (TMS) provides a scalable, secure, and user-friendly solution that significantly improves transportation resource management. By comparing the results with existing methods, it is evident that the proposed system offers a comprehensive and efficient approach to addressing the challenges in transportation management.

**6.CONCLUSION**

The Transport Management System (TMS) is a robust solution designed to streamline the management and booking of transportation resources. Addressing the inefficiencies of traditional methods and the limitations of basic TMS platforms, this system leverages modern technologies to enhance user experience, operational efficiency, and resource utilization. The implementation of the TMS has resulted in significant improvements in transportation resource management, enhancing user satisfaction and operational efficiency. Future enhancements could include real-time GPS tracking, predictive maintenance analytics, multi-language support, and mobile application integration. Through the integration of modern technologies and methodologies, the TMS addresses the shortcomings of traditional methods and basic TMS platforms. With features such as user authentication, comprehensive vehicle management, intuitive booking interface, automated notifications, and administrative oversight, the TMS provides a centralized and efficient platform for managing transportation activities. Its successful implementation signifies its potential to streamline operations, increase customer satisfaction, and provide a competitive advantage in the business landscape. In conclusion, the TMS is a significant advancement in transportation management, providing a comprehensive, secure, and efficient solution that meets the evolving needs of modern businesses. It streamlines operations, enhances customer satisfaction, and offers a competitive edge in today's dynamic business environment.

**7.SAMPLE CODING**

from flask import Flask, request, render\_template, redirect

from werkzeug.utils import secure\_filename

import os

import mysql.connector

app = Flask(\_name\_)

app.config['UPLOAD\_FOLDER'] = 'D:\Documents\Python\Py-project-IV'

app.config['ALLOWED\_EXTENSIONS'] = {'pdf', 'jpg', 'jpeg', 'png', 'gif'}

def allowed\_file(filename):

return '.' in filename and \

filename.rsplit('.', 1)[1].lower() in app.config['ALLOWED\_EXTENSIONS']

@app.route('/vehicle', methods=['POST'])

def vehicle():

username = request.form['username']

v\_name = request.form['v\_name']

model = request.form['model']

load\_capacity = request.form['load-capacity']

contact = request.form['contact']

f\_t = request.form['f\_t']

driver\_license = request.form['driver-license']

other\_details = request.form['other-details']

vehicle\_registration = request.files['vehicle-registration']

insurance\_document = request.files['insurance-document']

if vehicle\_registration and allowed\_file(vehicle\_registration.filename):

filename = secure\_filename(vehicle\_registration.filename)

vehicle\_registration.save(os.path.join(app.config['UPLOAD\_FOLDER'], filename))

if insurance\_document and allowed\_file(insurance\_document.filename):

filename = secure\_filename(insurance\_document.filename)

insurance\_document.save(os.path.join(app.config['UPLOAD\_FOLDER'], filename))

cnx = mysql.connector.connect(user='root', password='Bharath@2004',

host='localhost',

database='tms')

cursor = cnx.cursor()

query = "INSERT INTO vehicles(username,v\_name, model, load\_capacity, contact, f\_t, driver\_license, other\_details) VALUES(%s, %s, %s, %s, %s, %s, %s, %s)"

data = (username,v\_name, model, load\_capacity, contact, f\_t, driver\_license, other\_details)

cursor.execute(query, data)

cnx.commit()

cursor.close()

cnx.close()

return redirect('/landing')

@app.route('/landing', methods=['GET'])

def landing():

search\_query = request.args.get('search')

cnx = mysql.connector.connect(user='root', password='Bharath@2004',

host='localhost',

database='tms')

cursor = cnx.cursor()

if search\_query:

query = "SELECT \* FROM vehicles WHERE f\_t = %s"

cursor.execute(query, (search\_query,))

else:

query = "SELECT \* FROM vehicles"

cursor.execute(query)

rows = cursor.fetchall()

column\_names = [i[0] for i in cursor.description]

data = [dict(zip(column\_names, row)) for row in rows]

cursor.close()

cnx.close()

return render\_template('landing.html', data=data)

@app.route('/signup', methods=['POST'])

def signup():

name = request.form.get('name')

username = request.form.get('username')

password = request.form.get('password')

confirm\_password = request.form.get('confirm-password')

contact = request.form.get('contact')

insert\_data(name, username, password, confirm\_password, contact)

return render\_template('login.html')

@app.route('/login', methods=['POST'])

def login():

username = request.form.get('username')

password = request.form.get('password')

if validate\_login(username, password):

return redirect('/landing')

else:

return 'Invalid username or password.'

def insert\_data(name, username, password, confirm\_password, contact):

try:

cnx = mysql.connector.connect(user='root', password='Bharath@2004',

host='localhost',

database='tms')

cursor = cnx.cursor()

query = ("INSERT INTO your\_table "

"(name, username, password, confirm\_password, contact) "

"VALUES (%s, %s, %s, %s, %s)")

cursor.execute(query, (name, username, password, confirm\_password, contact)

cnx.commit()

cursor.close()

cnx.close()

except mysql.connector.Error as err:

print(f"Something went wrong: {err}")

def validate\_login(username, password):

try:

cnx = mysql.connector.connect(user='root', password='Bharath@2004',

host='localhost',

database='tms')

cursor = cnx.cursor()

query = ("SELECT \* FROM your\_table WHERE username = %s AND password = %s")

cursor.execute(query, (username, password))

result = cursor.fetchone()

cursor.close()

cnx.close()

if result:

return True

else:

return False

except mysql.connector.Error as err:

print(f"Something went wrong: {err}")

return False

@app.route('/book', methods=['POST'])

def book():

username = request.form['username']

v\_name = request.form['v\_name']

contact = request.form['contact']

tons = request.form['tons']

date = request.form['date']

cnx = mysql.connector.connect(user='root', password='Bharath@2004',

host='localhost',

database='tms')

cursor = cnx.cursor()

query = "INSERT INTO orders(username, v\_name, contact, tons, date) VALUES(%s, %s, %s, %s, %s)"

data = (username, v\_name, contact, tons, date)

cursor.execute(query, data)

cnx.commit()

cursor.close()

cnx.close()

return render\_template('landing.html', username=username, v\_name=v\_name)

@app.route('/check\_booking', methods=['GET', 'POST'])

def check\_booking():

if request.method == 'POST':

username = request.form['username']

v\_name = request.form['v\_name']

cnx = mysql.connector.connect(user='root', password='Bharath@2004',

host='localhost',

database='tms')

cursor = cnx.cursor()

query = "SELECT \* FROM orders WHERE username = %s AND v\_name = %s"

cursor.execute(query, (username, v\_name))

rows = cursor.fetchall()

column\_names = [i[0] for i in cursor.description]

data = [dict(zip(column\_names, row)) for row in rows]

cursor.close()

cnx.close()

return render\_template('booking\_details.html', data=data)

return render\_template('check\_booking.html')

if \_name\_ == '\_main\_':

app.run(debug=True)