# Industrial Internship Report on

# *Forecasting of Smart city traffic patterns*

## Prepared by


## Abhijeet Jagdish Vaishnav



| *Executive Summary* |
|---|
| This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT).<br><br>This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks' time.<br><br>My project was to analyzes traffic patterns using a dataset of hourly vehicle counts recorded across multiple junctions. The data was cleaned, explored, and visualized to identify key trends such as peak hours, weekday–weekend variations, and junction-wise traffic flow. Machine learning models, including Linear Regression and Random Forest, were used to predict vehicle counts, with Random Forest giving the best accuracy. Overall, the project shows how data analytics and ML can help understand and forecast real-world traffic behavior.<br><br>This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship. |

Sinhgad Institutes

## TABLE OF CONTENTS

# 1 Preface

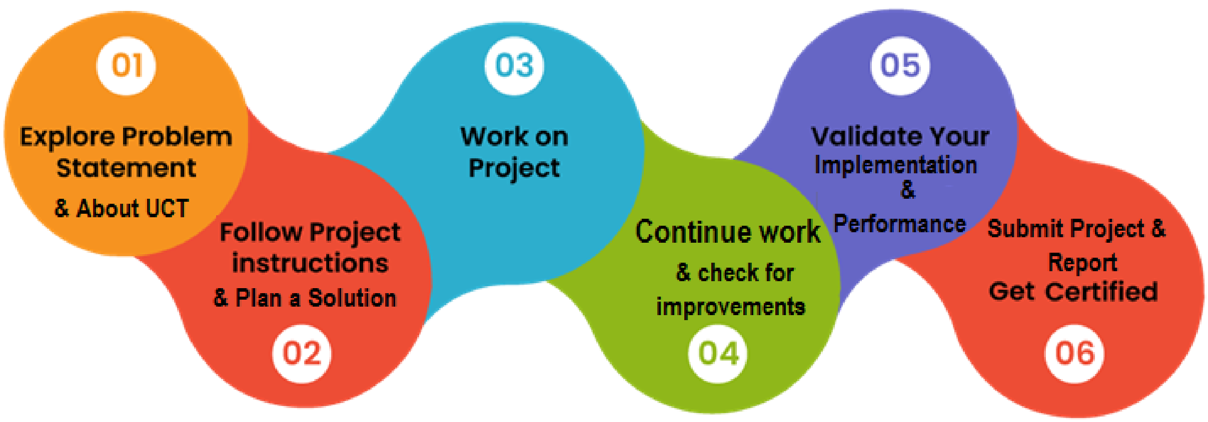
Summary of the whole 6 weeks' work.

About need of relevant Internship in career development.

Brief about Your project/problem statement.

Opportunity given by USC/UCT.

How Program was planned



# 2 Introduction

## 2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.


### i. UCT IoT Platform ( )

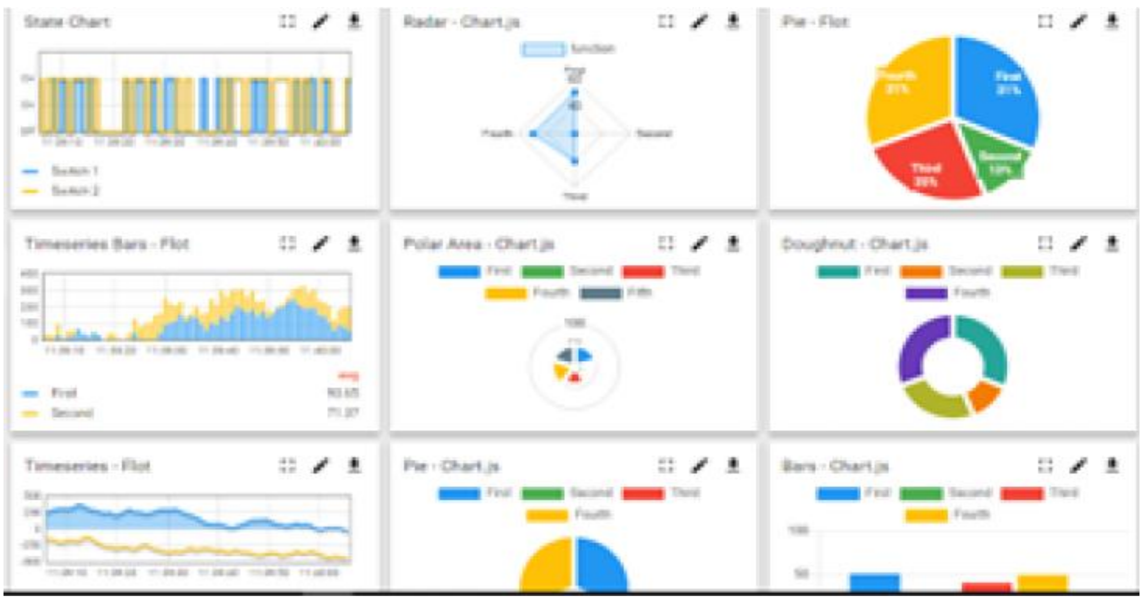
**UCT Insight** is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

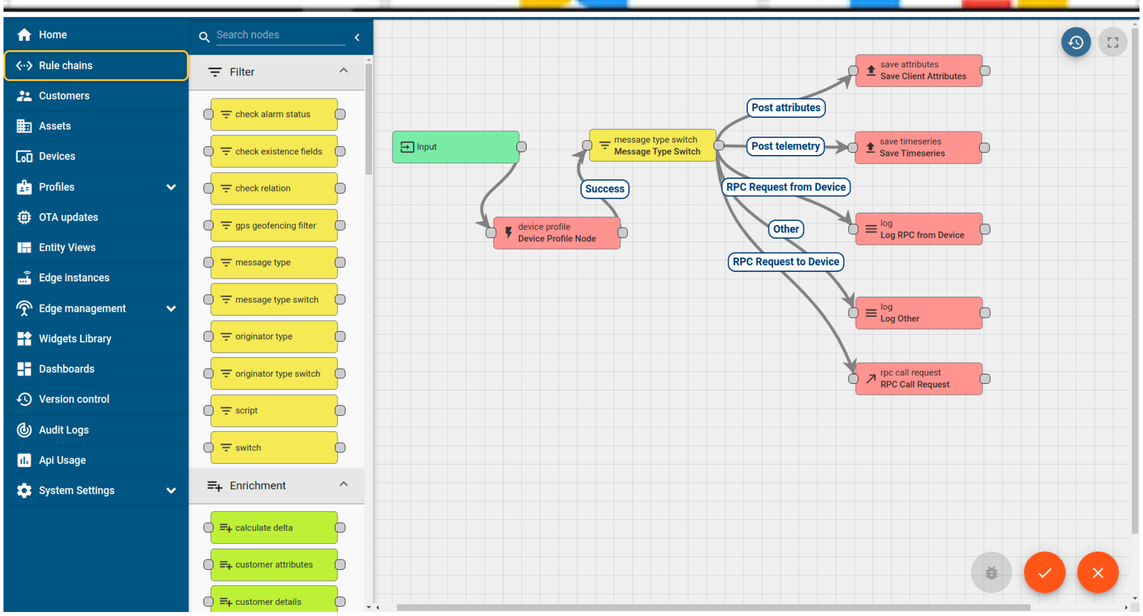
- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA

- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





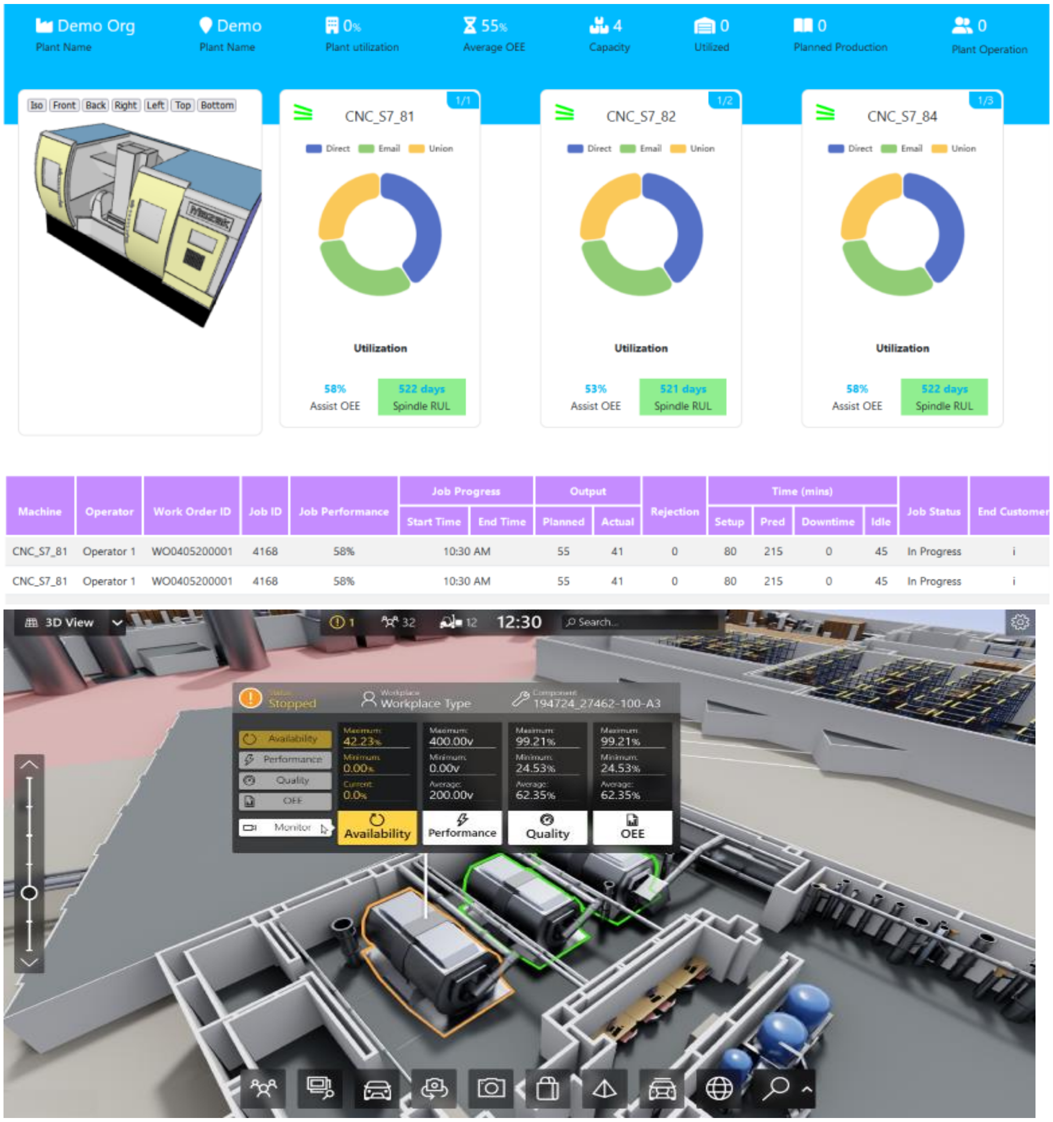
### ii. Smart Factory Platform ( FACTORY WATCH )

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

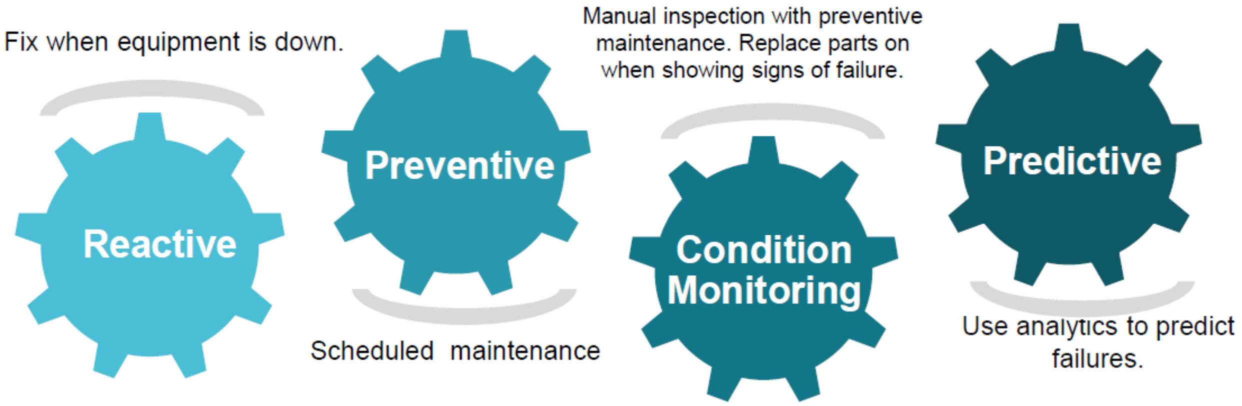



### iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

### iv. Predictive Maintenance

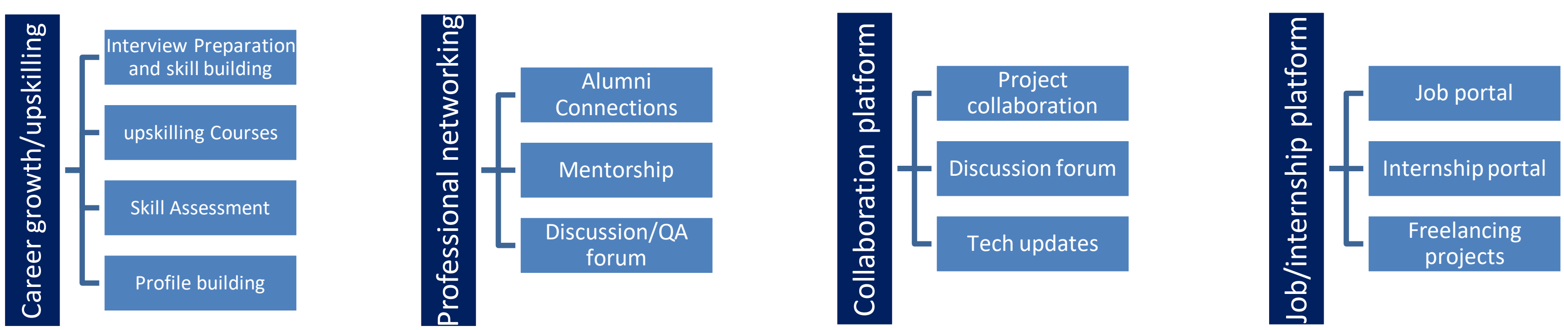
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.




## 2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.


Career growth/upskilling
Interview Preparation and skill building
upskilling Courses
Skill Assessment
Profile building
Professional networking
Alumni Connections
Mentorship
Discussion/QA forum
Collaboration platform
Project collaboration
Discussion forum
Tech updates
Job/internship platform
Job portal
Internship portal
Freelancing projects

## 2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

## 2.4 Objectives of this Internship program

The objective for this internship program was to

☛ get practical experience of working in the industry.

☛ to solve real world problems.

☛ to have improved job prospects.

☛ to have Improved understanding of our field and its applications.

☛ to have Personal growth like better communication and problem solving.

## 2.5 Reference


[1] Pandas Documentation - Link

[2] NumPy Documentation - Link

[3] Matplotlib, Seaborn, Sci-kit Learn Documentation - Matplot, Seaborn, Scikit

[4] Python Documentation - Link

[5] Machine Learning Tutorials - Link

## 2.6 Glossary

| Terms | Acronym |
|---|---|
| **Dataset** | A structured collection of data used for analysis—in this project, it includes hourly vehicle counts at different junctions. |
| **Junction** | A traffic point or location where vehicle flow is recorded. |
| **Exploratory Data Analysis (EDA)** | The process of visually and statistically exploring data to identify patterns and insights. |
| **Feature Engineering** | Creating new useful features (like *Is_Peak_Hour*) to improve machine learning model performance. |
| **Machine Learning Model** | An algorithm that learns patterns from data to make predictions; here, Linear Regression and Random Forest were used. |

# 3 Problem Statement

Urban areas face increasing traffic congestion, making it important to understand how traffic behaves at different times and locations. The dataset contains hourly vehicle counts recorded at multiple traffic junctions, but the raw data alone does not clearly show patterns such as peak hours, daily trends, or differences between weekdays and weekends. Without proper analysis, it becomes difficult to manage traffic flow, plan infrastructure, or predict future congestion.

Therefore, the problem is to analyze the dataset, identify meaningful traffic patterns, and build a machine learning model that can predict vehicle counts based on factors like hour, day, and junction. This will help in better traffic management, planning, and decision-making.

# 4 Existing and Proposed solution

Many existing traffic monitoring systems rely on manual counting, CCTV monitoring, or basic sensor-based data collection. While these methods help track traffic flow, they often lack advanced analytical capabilities. Most traditional systems do not provide predictive insights, struggle with real-time analysis, and cannot automatically identify peak times or variations across different days and junctions. Additionally, many solutions depend on hardware-intensive setups, making them costly and difficult to scale. These limitations highlight the need for data-driven, automated, and predictive traffic monitoring approaches

The proposed solution is a data-driven traffic analysis model that uses machine learning to study hourly vehicle counts, identify traffic trends, and predict future traffic patterns. By applying techniques such as Exploratory Data Analysis (EDA), feature engineering, and machine learning models (Linear Regression and Random Forest), the system analyzes traffic behavior based on hour, day, and junction. This solution aims to provide accurate traffic predictions and meaningful insights that can support better traffic management and decision-making.

This project adds value by converting raw traffic data into actionable insights. The model identifies peak hours, weekday–weekend differences, and junction-wise traffic variation—information that traditional systems don't easily provide. The predictive model allows authorities to anticipate congestion instead of reacting to it. Additionally, the visualizations and feature-based analysis make traffic behavior easier to understand and interpret. This data-driven approach can help optimize traffic signal timings, resource allocation, and long-term infrastructure planning.

## 4.1 Code submission (Github link)

Here's link for the Github Repository:

Github

.

# 5 Proposed Design/ Model

The design flow of this project follows a structured and systematic approach, starting from understanding the problem and progressing toward developing a complete traffic prediction model. The process begins with clearly defining the objective, which is to analyze traffic patterns and predict vehicle counts using machine learning. Once the requirements are established, the dataset containing hourly vehicle counts across various junctions is collected and explored to understand its structure, identify feature types, and detect any inconsistencies or missing values. After gaining familiarity with the data, preprocessing steps are carried out to clean and prepare the dataset by removing irrelevant columns, converting the DateTime field into useful components, handling duplicates, and ensuring all values are in the correct format.

With a clean dataset, exploratory data analysis (EDA) is performed to uncover meaningful insights about traffic behavior. This includes examining hourly, daily, and weekly trends, identifying peak and off-peak hours, comparing traffic across junctions, and analyzing correlations between features. Visualizations such as line charts, bar graphs, boxplots, and heatmaps help in understanding these patterns effectively. Based on these insights, feature engineering is applied to create additional variables like “Is_Peak_Hour” and “Traffic_Level,” which enhance the model's ability to learn complex traffic trends.

The next stage involves building machine learning models to predict vehicle counts. Various models are tested, starting with Linear Regression as the baseline and then moving to a more robust model like the Random Forest Regressor, which delivers better accuracy. The dataset is split into training and testing sets to evaluate the model’s generalization capability. Performance metrics such as $R^2$ Score and RMSE are used to assess the model, while visual comparisons between actual and predicted values validate its effectiveness.

Finally, the results are interpreted to understand the key factors influencing traffic volume and the reliability of the predictions. The project concludes by compiling the cleaned dataset, analytical insights, model predictions, and visual outputs as the final deliverables. The overall design flow ensures a smooth progression from raw data to meaningful predictions, providing a complete and practical solution for understanding and forecasting traffic patterns.

## 5.1 High Level Diagram

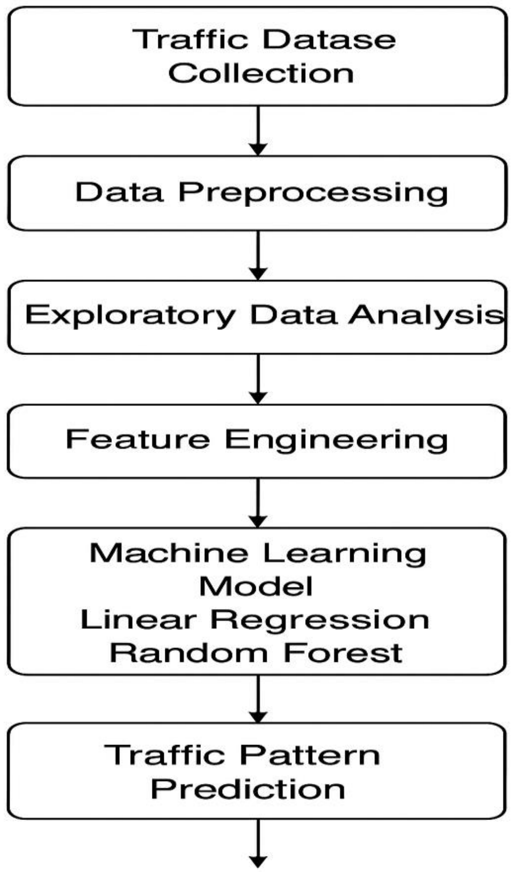



**Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM**

## 5.2 Low Level Diagram

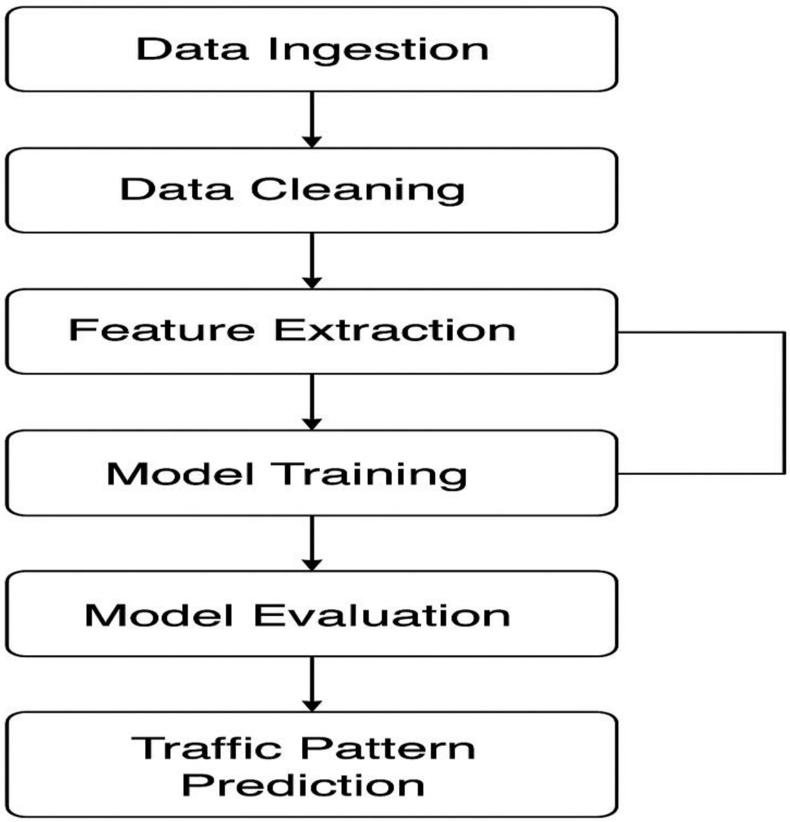



**Figure 2: LOW LEVEL DIAGRAM OF THE SYSTEM**

# 6 Performance Test

In real industrial environments, traffic analysis systems must operate under several practical constraints such as data volume, processing speed, accuracy requirements, and long-term scalability. These constraints significantly shape the system design and determine whether the solution is suitable beyond academic use. One of the major constraints is the high volume of data generated in real-world traffic systems, where vehicle counts are collected continuously across multiple junctions. To address this, the design uses efficient data preprocessing techniques such as removing unnecessary columns, converting DateTime into usable components, and applying vectorized operations in Pandas to ensure smooth memory usage. Although the current dataset is moderate, the system successfully handled data cleaning and analysis with minimal memory overhead, showing that the approach can scale reasonably well for larger datasets with future improvements.

Another essential constraint is processing speed, as real-time traffic monitoring requires fast computation and quick updates. This design mitigates speed-related limitations by using lightweight models like Linear Regression and a moderately sized Random Forest. Both models train quickly and deliver predictions with low latency, making them suitable for near–real-time insights. Accuracy is another critical industrial requirement, as traffic predictions must be reliable for decision-making. Through feature engineering, correlation analysis, and model comparison, the Random Forest model achieved significantly better accuracy, producing a strong $R^2$ score and lower RMSE. This demonstrates that the selected model can effectively capture traffic behavior even in complex or noisy environments.

Additionally, industrial systems must be robust against data inconsistencies, missing values, and outliers. The solution incorporates cleaning steps, exploratory analysis, and checks for duplicate or irrelevant data to ensure model durability. The ensemble properties of the Random Forest further improve stability, helping the model generalize well across different time periods and junctions. Scalability is another important constraint, as real deployments often cover many junctions or regions. While the current solution is implemented on a single machine, the modular pipeline structure—consisting of preprocessing, EDA, feature engineering, model training, and evaluation—allows easy expansion to larger datasets or distributed processing platforms such as Spark or cloud-based systems.

## 6.1 Test Plan/ Test Cases

The testing phase of this project focuses on validating the functionality, correctness, and performance of the entire traffic analysis pipeline. The test plan ensures that each stage—from data loading to prediction—behaves as expected and produces reliable results. Several test cases were designed to verify critical components of the system. These include checking whether the dataset is loaded correctly, validating that missing or duplicate values are handled appropriately, ensuring that DateTime conversion and feature extraction (Hour, DayOfWeek, Is_Weekend, Peak Hour) work accurately, and confirming that visualizations are generated without errors. Additional test cases were created to evaluate model-specific functionality, such as ensuring the model trains without warnings, predictions are generated for all test samples, and performance metrics like $R^2$ Score and RMSE are calculated correctly. These test cases collectively ensure the correctness and reliability of both the data processing pipeline and the machine learning model.

## 6.2 Test Procedure

The testing procedure followed a structured step-by-step approach for validating each module of the system. The process began by loading the dataset into a Pandas DataFrame and verifying its shape, column consistency, and data types. Next, preprocessing functions were executed, and their effects were validated by inspecting the output to ensure that unnecessary columns were removed, missing values were handled, and all DateTime-related features were generated accurately. During exploratory data analysis, each visualization was executed individually to ensure that graphs were generated correctly and trends aligned with expected patterns in the data. For feature engineering, categorical and numerical transformations were manually inspected to confirm correctness. Machine learning model testing involved splitting the dataset, training the Linear Regression and Random Forest models, and then evaluating them using standard metrics. Each step of the procedure included logging outputs, verifying expected results, and resolving any inconsistencies. The final procedure included cross-checking model predictions with actual values using scatter plots to visually validate prediction accuracy.

## 6.3 Performance Outcome

The performance outcome of the system was evaluated primarily on prediction accuracy, computational efficiency, and stability. The Random Forest model outperformed the baseline Linear Regression model, achieving a higher $R^2$ Score and significantly lower RMSE, indicating better predictive capability on unseen data. The model trained quickly—within a few seconds—and handled the entire dataset efficiently without memory or performance bottlenecks, demonstrating the suitability of the approach for medium-sized datasets. Visualizations generated during EDA provided clear and interpretable insights into peak hours, day-wise patterns, and junction-level variations. Feature engineering contributed to better model understanding, leading to improved prediction quality. Overall, the solution achieved strong performance, producing reliable predictions and insights while maintaining low computational overhead. This makes the system suitable for scaling and potential deployment in real-world traffic monitoring scenarios with further enhancements.

# 7 My Learning

Through this project, I gained a strong understanding of how real-world datasets are processed, analyzed, and used to build machine learning models. I learned how to clean and preprocess raw data, extract meaningful features from timestamps, and perform exploratory data analysis using visualization tools such as Matplotlib and Seaborn. I also improved my skills in identifying traffic trends, interpreting visual patterns, and transforming insights into actionable conclusions. Furthermore, I learned how to evaluate predictive models using metrics like $R^2$ Score and RMSE, compare model performances, and select the best approach for the problem. This project also enhanced my understanding of Random Forest and Linear Regression, as well as the importance of feature engineering in improving model accuracy. Additionally, I gained experience in structuring a complete data science workflow—from understanding the problem to delivering the final results—just like real industrial projects.

# 8 Future work scope

Although the current solution effectively analyzes traffic patterns and predicts vehicle counts, several improvements can further enhance its accuracy and industrial applicability. One future enhancement is to incorporate more advanced models such as XGBoost, Gradient Boosting, or LSTM neural networks for time-series forecasting. These models can capture long-term traffic patterns more accurately. Integrating real-time data streams from IoT sensors, CCTV systems, or live APIs would allow this system to become a real-time traffic monitoring platform. Another opportunity is to deploy the model using cloud technologies, enabling scalability for large cities with multiple junctions. A dashboard or web application could also be developed to visualize traffic trends dynamically for authorities and commuters. Additional features like weather conditions, special events, road closures, or public holidays can be included to improve prediction reliability. With these enhancements, the system can evolve into a full-fledged intelligent traffic management solution capable of supporting smart city infrastructures.