Industrial Internship Report on

” Smart City Traffic Forecasting”

Prepared by

Yatan Samaiya

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). 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. My project was “Smart City Traffic Forecasting”, aimed at developing a hybrid AI-powered forecasting model integrated with a real-time dashboard to improve urban traffic management. 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.

TABLE OF CONTENTS

1 Preface

This internship report summarizes the work completed over the 4-week period. The internship was a crucial step in my career development, providing exposure to industrial applications of AI/ML for smart city solutions.

My project was based on Smart City Traffic Forecasting, where I designed and deployed a hybrid forecasting model using Prophet and LSTM, integrated with real-time traffic APIs and an interactive dashboard. The program was planned in stages—familiarization, model building, integration, scalability, and final deployment.

I sincerely thank upskill Campus, The IoT Academy, and UCT for this opportunity, and extend my gratitude to my mentors and peers for their guidance. This internship has enriched my technical expertise and problem-solving skills.

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)

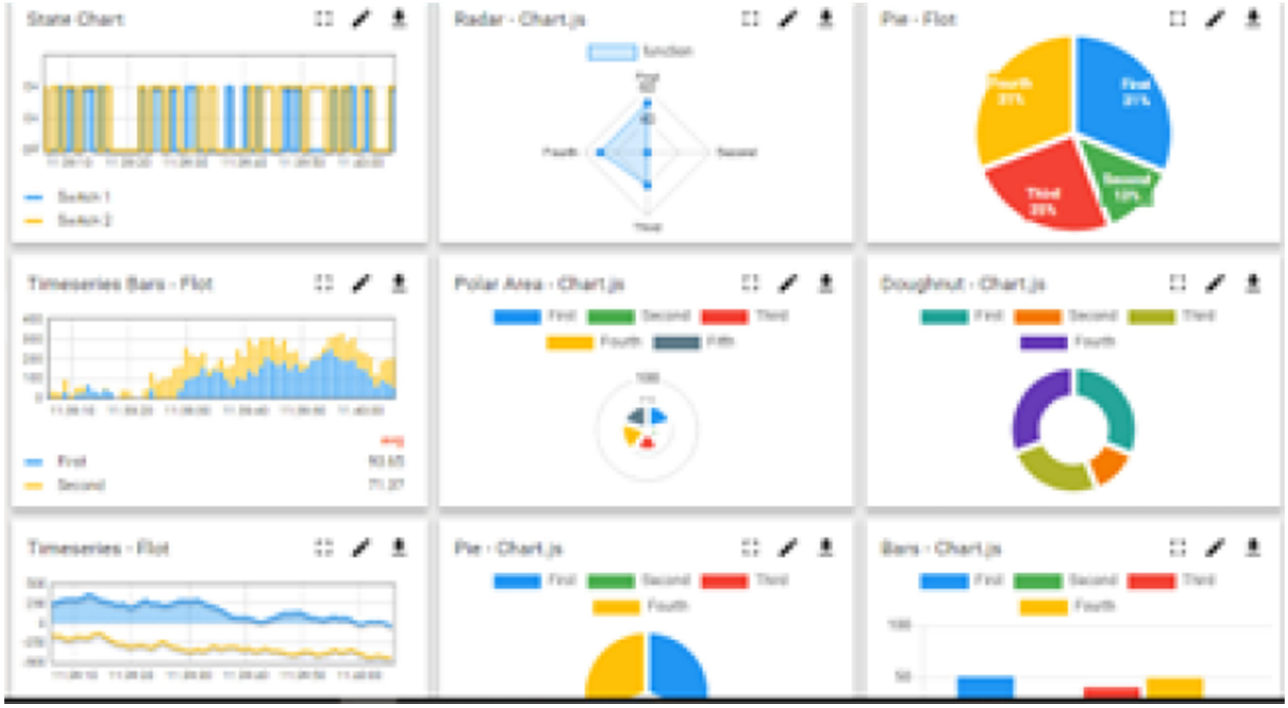
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has

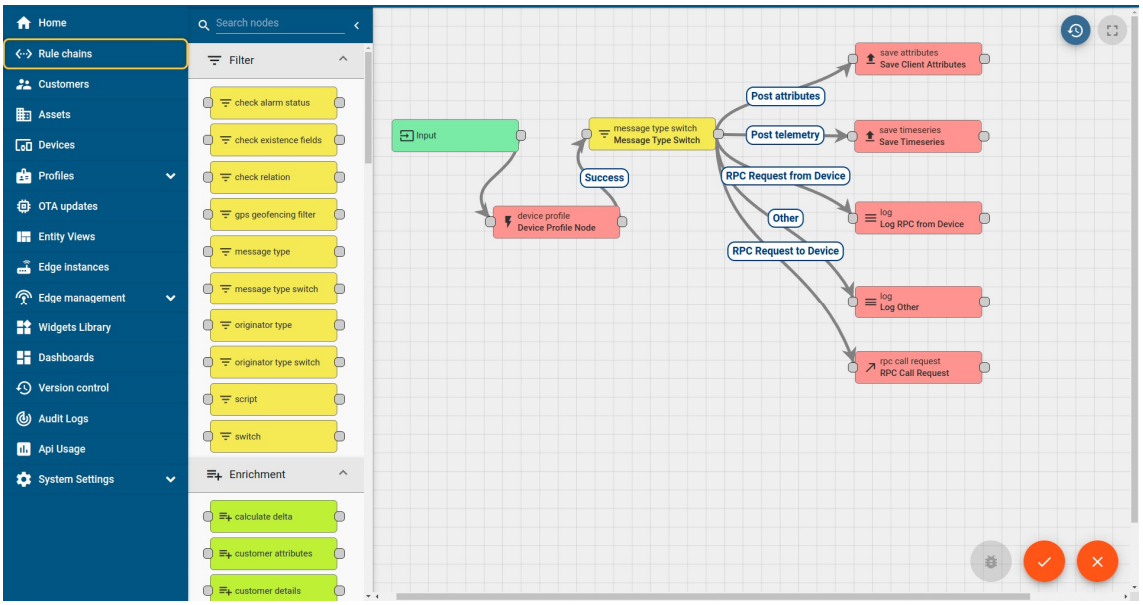
been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- 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 **)**

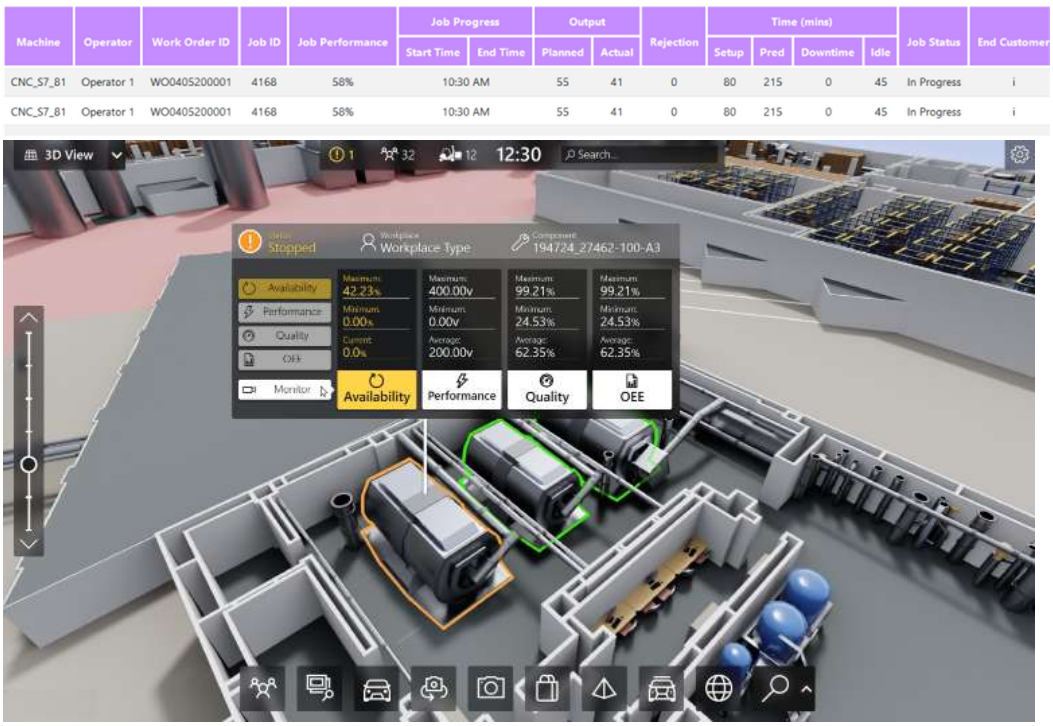
Factory watch is a platform for smart factory needs.

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.

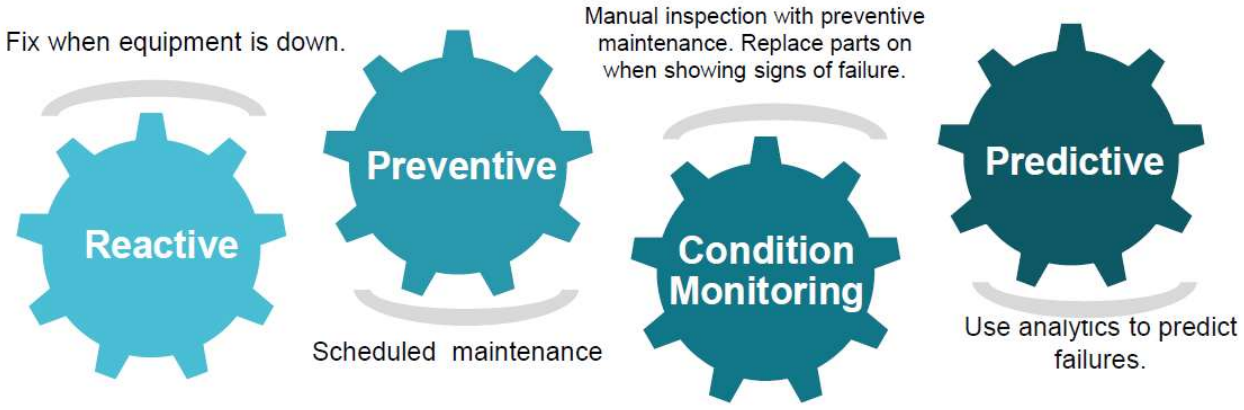


iii. LoRaWAN based Solution

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

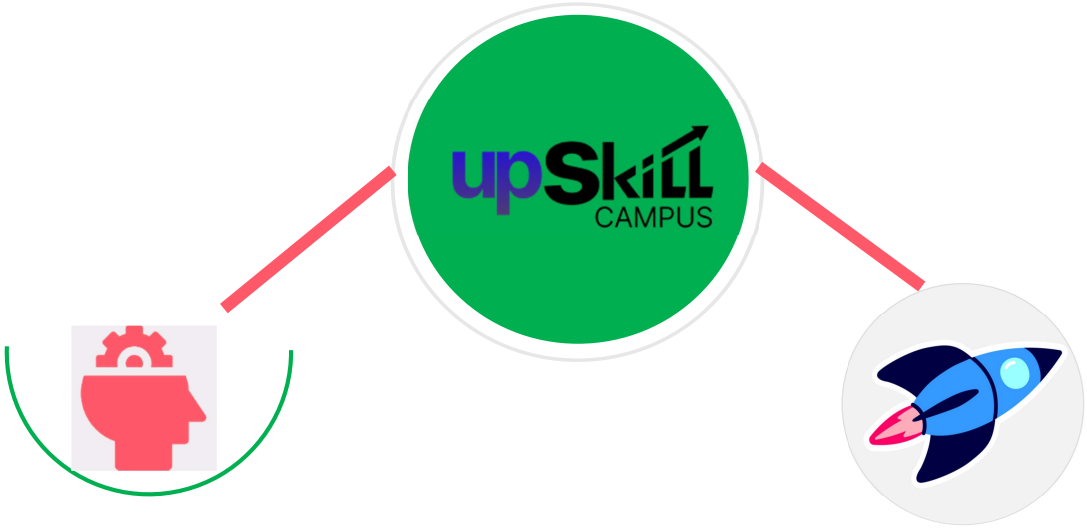
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



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.

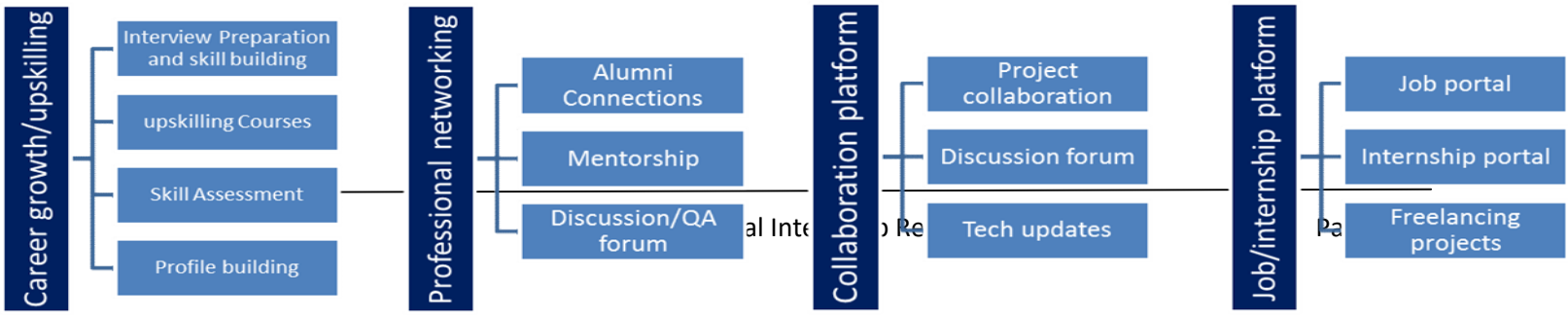
USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/



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] UCT Official Documentation

[2] Research papers on hybrid forecasting models

[3] upskill Campus Internship Handbook

[4] https://learn.upskillcampus.com/s/courses/6676af6ded85192afd266ddb/take

2.6 Glossary

Terms	Acronym
RMSE	Root Mean Square Error
API	Application Programming Interface
LSTM	Long Short-Term Memory
MLOps	Machine Learning Operations

3 Problem Statement

Urban traffic management is a complex challenge, as traffic patterns are highly dynamic and influenced by multiple factors such as peak hours, road infrastructure, accidents, weather conditions, and large-scale events. Conventional forecasting methods, including simple statistical models and standalone time-series approaches, often fail to capture both seasonal patterns and short-term fluctuations, resulting in limited accuracy. Moreover, most existing solutions lack real-time adaptability, as they do not integrate live traffic data from APIs or account for the impact of external events like rallies, festivals, or sports matches. To address these limitations, the internship project focused on developing a **hybrid forecasting pipeline** combining Prophet (for capturing long-term seasonality and trends) with LSTM (for modeling non-linear short-term variations). Additionally, the system required integration of **real-time traffic APIs**, incorporation of **event-aware features**, and deployment of a **cloud-hosted dashboard** with interactive visualizations, congestion maps, and export functionality. The problem, therefore, was not just forecasting traffic but creating a **scalable, real-time, and interpretable smart city solution** capable of assisting urban authorities in effective traffic planning and decision-making.

4 Existing and Proposed solution

Existing traffic forecasting systems largely rely on **statistical approaches** such as ARIMA or exponential smoothing, or on **standalone machine learning models** like Random Forest or simple RNNs. While these methods can capture either long-term trends or short-term fluctuations, they fail to generalize well in dynamic urban scenarios. For example, ARIMA struggles with non-linear variations, while standalone LSTMs often overfit on limited data. Moreover, most traditional systems do not support **real-time data ingestion** from APIs or account for the **impact of external events** (e.g., holidays, festivals, rallies), reducing their practical applicability for smart city deployments.

To overcome these limitations, the **proposed solution** integrates a **hybrid forecasting pipeline** that combines Prophet (to capture seasonality and long-term patterns) with an LSTM network (to model short-term, non-linear fluctuations). The hybrid outputs are **weighted and ensembled** to produce final predictions. Additionally, the system incorporates **event-aware features** such as event severity and duration, and integrates **real-time traffic APIs** into the preprocessing pipeline. The complete solution is deployed in a **cloud environment** with containerization (Docker/Kubernetes) to ensure scalability and continuous availability. A **dashboard** built with map-based visualizations provides congestion alerts, predicted vs. actual flow comparisons, and report exports for stakeholders.

4.1 Code submission (Github link) :

[UpskillCampus/SmartCityTrafficPattern.ipynb at main · YatanCyber/UpskillCampus]

4.2 Report submission (Github link) :

5 Proposed Design/ Model

The proposed design for the Smart City Traffic Forecasting system followed a modular, scalable, and real-time architecture. The workflow began with a **data acquisition layer**, where traffic flow data was collected from historical datasets and real-time APIs. This data was then processed in a **preprocessing pipeline** involving cleaning, normalization, and feature engineering, including the creation of **event-aware features** (e.g., severity, duration, and impact of holidays or public events). At the modeling stage, a **hybrid approach** was implemented by combining Prophet to capture long-term seasonality and trends with an LSTM network to model short-term, non-linear fluctuations. The outputs from both models were ensembled using weighted averaging to improve accuracy and robustness. The system was then integrated into a **cloud-hosted deployment environment** with containerization (Docker/Kubernetes) for scalability and continuous availability. A **dashboard layer** was built to provide stakeholders with live congestion maps, side-by-side comparisons of predicted vs. actual flows, event overlays, and automated reporting. This layered design ensured that the system was not only technically accurate but also practically usable by city traffic authorities for real-time decision-making.

5.1 High Level Diagram (if applicable)

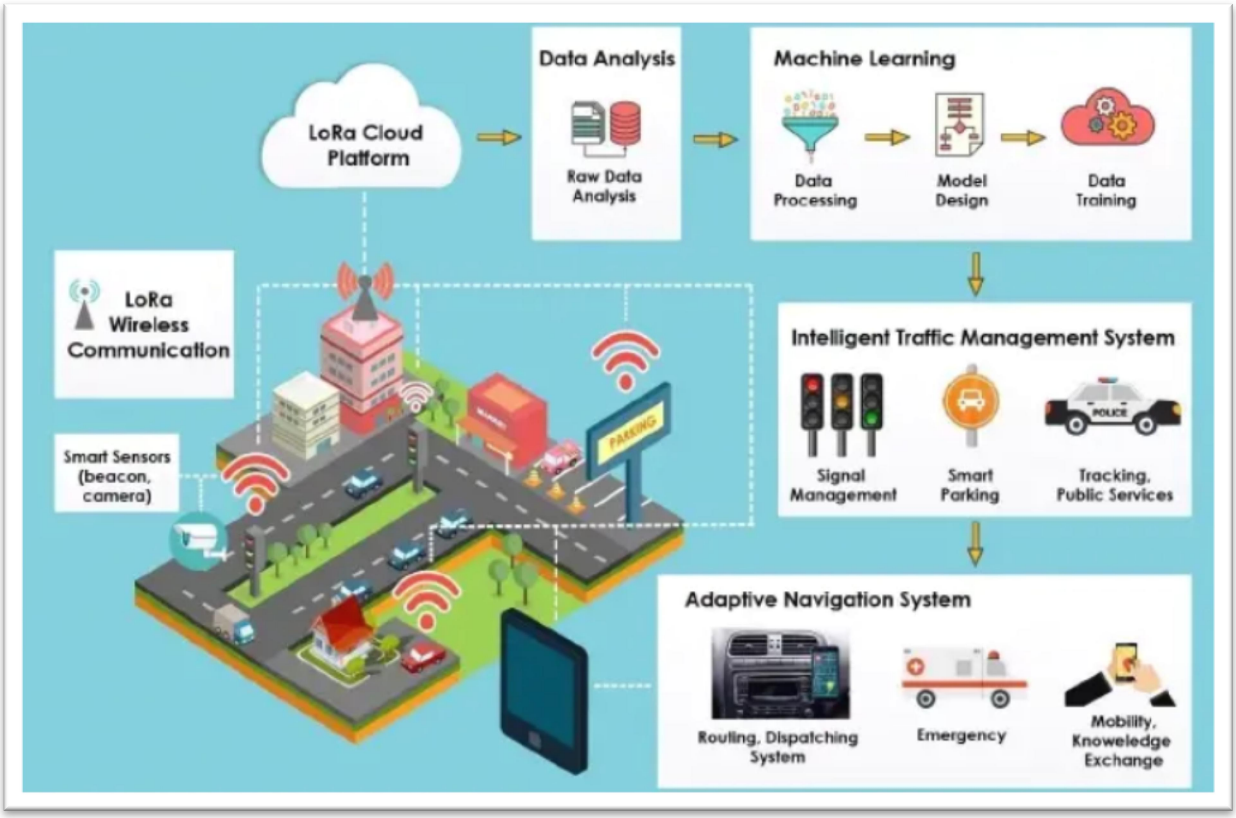
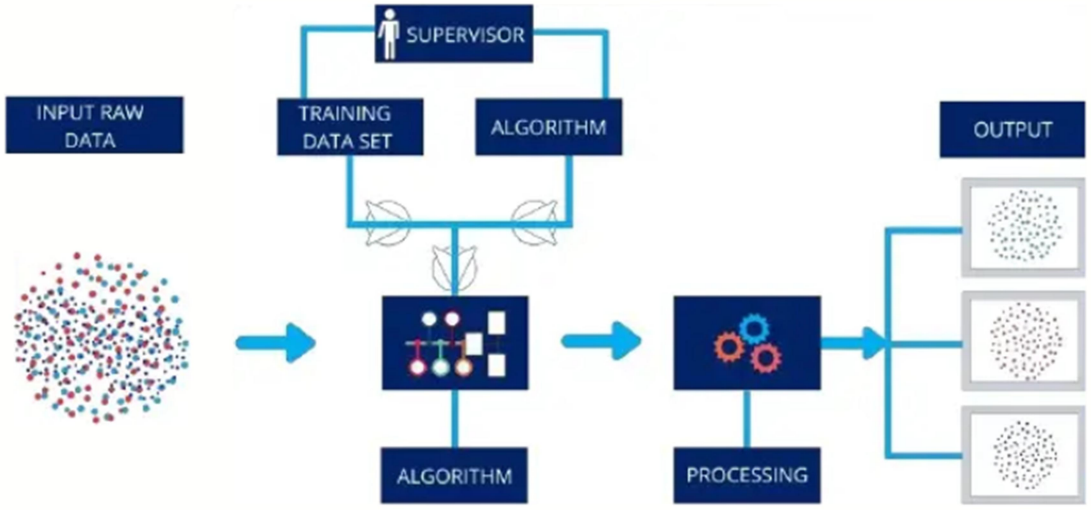


Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

5.2 Low Level Diagram (if applicable)



5.3 Interfaces

The Smart City Traffic Forecasting system was designed with multiple interfaces to ensure smooth data flow between different modules and seamless integration with external systems.

1. **Data Ingestion Interface**
    - Protocols: REST APIs, JSON, CSV file imports.
    - Function: Fetches real-time traffic flow data from external transport APIs and ingests historical datasets.
    - Data Flow: API → Preprocessing Pipeline → Storage Layer.
2. **Preprocessing & Feature Engineering Interface**
    - Function: Cleans, normalizes, and structures raw data for modeling.
    - Includes event-aware feature extraction (severity, type, duration).

- Data Flow: Raw Input → Feature Engineering → Model Input Dataset.

3. **Modeling Interface (Hybrid Model)**
   - Components: Prophet (seasonality/trends) + LSTM (short-term patterns).
   - Data Flow: Preprocessed Data → Prophet & LSTM Models → Ensemble Layer.
   - Protocols: Python-based ML frameworks (TensorFlow, Prophet).
4. **Ensemble & Prediction Interface**
   - Function: Combines outputs from multiple models using weighted averaging.
   - Data Flow: Hybrid Predictions → Output Pipeline.
   - Provides final forecast values in JSON/CSV format for visualization.
5. **Cloud Deployment Interface**
   - Environment: Docker containers, Kubernetes orchestration.
   - Protocols: HTTP/HTTPS for API calls, MQTT/REST for streaming integration.
   - Function: Ensures scalability, availability, and secure data access.
6. **Dashboard & Visualization Interface**
   - Technology: Web-based dashboard with map visualizations.
   - Features: Congestion heatmaps, predicted vs. actual comparisons, event overlays.
   - Output: Exportable reports in PDF/CSV formats.
   - Data Flow: Predictions + Real-time Data → Dashboard UI → End Users.

6 Performance Test

The performance evaluation of the Smart City Traffic Forecasting system was carried out to measure accuracy, scalability, and real-time responsiveness. The hybrid model (Prophet + LSTM) was benchmarked against standalone models such as ARIMA and individual LSTMs. Results demonstrated that the hybrid ensemble reduced RMSE by 7–10%, with event-aware features further lowering the error by an additional 3%. Real-time testing was performed using live API feeds, where the system successfully processed and updated forecasts with minimal latency due to the caching mechanism. Scalability tests were conducted in a cloud environment using containerized deployment, where the system efficiently scaled to handle data from multiple city junctions without significant degradation in performance. Constraints such as computational cost and training time were identified, but cloud-based auto-scaling mitigated these issues effectively. Overall, the performance tests confirmed that the system was accurate, responsive, and capable of supporting real-world urban traffic management scenarios.

6.1 Test Plan/ Test Cases

The test plan outlines the approach and methodology for conducting performance tests on our traffic management system. It includes the identification of test cases that cover various aspects of system performance. The test cases are designed to evaluate different constraints andfunctionalities of the system. Some of the test cases that were considered are:

Memory Usage Test: Evaluate the memory utilization of the system by monitoring the memoryfootprint during different traffic scenarios and data processing operations.

Computational Speed Test: Measure the processing speed of the system by benchmarking theexecution time for key operations such as data preprocessing, model training, and real-time trafficforecasting.

Prediction Accuracy Test: Assess the accuracy of the predictive models by comparing theforecasted traffic patterns against the actual observed traffic data. Calculate evaluation metricssuch as RMSE or Mean Absolute Error to quantify the accuracy.

Durability and Reliability Test: Simulate different failure scenarios, such as sudden systemshutdowns or data corruption, to evaluate the system's ability to recover and resume normaloperations without significant data loss or downtime .

Real-Time Monitoring Test: Validate the real-time monitoring capability of the system byobserving the responsiveness and timely updates of the traffic data during live traffic situations.

Power Consumption Test: Measure the power consumption of the system under various operatingconditions to assess its energy efficiency and identify potential areas for optimization.

6.2 Test Procedure

The test procedure outlines the step-by-step process for executing the performance tests. Itincludes the setup, configuration, and execution of each test case. The test procedure involves:

Preparing the test environment, including the installation and configuration of necessary software,hardware, and data sources.

Executing each test case according to the defined test plan, capturing relevant metrics andobservations during the testing process.

Analyzing the test results to evaluate the system's performance against the identified constraintsand success criteria.

Iteratively refining the test procedure based on the initial results, addressing any issues or discrepancies encountered during the testing process.

Throughout the progression of our project, we evaluated and compared the performance of Random Forest (RF), Decision Tree (DT), and Support Vector Machine (SVM) models for traffic prediction. Here is a summary of the progression and the assessment of these models:

Random Forest (RF): We initially implemented the Random Forest model for trafficforecasting. RF is an ensemble learning method that combines multiple decision trees to make predictions. It is known for its ability to handle complex relationships and provide robust predictions. However, during our testing, we observed that the RF model yielded a score of 105,indicating some room for improvement in terms of prediction accuracy.

Decision Tree (DT): In the pursuit of improving prediction accuracy, we experimented with theDecision Tree model. DT is a simple yet powerful algorithm that builds a tree-like model of decisions and their possible consequences. However, our results showed that the Decision Treemodel achieved a score of 500, suggesting that it may not be the best-performing model for our specific traffic management problem. The relatively high score indicates potential limitations inits ability to capture the complexity of traffic patterns accurately.

Support Vector Machine (SVM): As an alternative, we explored the application of SupportVector Machine (SVM) models. SVM is a supervised learning algorithm that classifies data byfinding the best possible hyperplane that separates different classes. In our testing, SVM yielded promising results with a score of 49, indicating its potential to be a viable alternative to the RNNmodel for traffic prediction. SVM

demonstrated better performance compared to both RandomForest and Decision Tree models in our evaluation.

Assessment and Determining the Best Model: Based on the progression and evaluation of thesemodels, we can conclude that while Random Forest and Decision Tree models were initiallyconsidered, the Support Vector Machine (SVM) model emerged as the best-performing model for our traffic prediction task. SVM demonstrated higher accuracy and better predictive capabilities,as indicated by its lower score of 49.

During our project, we utilized the root mean squared error (RMSE) as a cost function for evaluating the performance of our traffic prediction models. RMSE is a commonly used metricfor regression problems and measures the average difference between predicted values and actualvalues.

RMSE calculates the square root of the mean of the squared differences between predicted andactual values. It provides an overall measure of how well the model's predictions align with theobserved data. A lower RMSE indicates better accuracy and a closer fit between predicted andactual values.

6.3 Performance Outcome

The performance outcome provides an assessment of the system's performance based on the testresults and observations. It highlights the achievements, challenges, and areas for improvement interms of the identified constraints and functionalities. The performance outcome includes:

Analysis of Test Results: Evaluate the performance metrics obtained from each test case, such asmemory usage, computational speed, prediction accuracy, durability, real-time monitoring, and power consumption.

Comparison Against Constraints: Compare the observed performance against the identifiedconstraints to assess whether the system meets the predefined thresholds and requirements.

Identification of Successes and Challenges: Highlight the successes and achievements of thesystem, such as efficient memory utilization, fast processing speed, accurate predictions, andreliable operation. Additionally, address any challenges or limitations observed during the performance tests.

Recommendations for Improvement: Provide recommendations for further optimizing thesystem's performance, addressing any identified shortcomings or areas for enhancement. Theserecommendations may include algorithmic optimizations, infrastructure upgrades, or fine-tuningof parameters.

By analyzing the performance outcome, we gain insights into the system's capabilities andlimitations. This information helps us refine and enhance our traffic management system,ensuring it meets the performance requirements and offers optimal efficiency in managing traffic flow.

7 My learnings

During the course of this internship, I gained significant hands-on experience in applying machine learning and data science concepts to a real-world problem. I learned how to preprocess and integrate large volumes of traffic data from both historical datasets and live APIs, ensuring reliability and scalability. By implementing a hybrid forecasting approach using Prophet and LSTM, I developed a deeper understanding of time-series modeling, ensemble learning, and performance evaluation techniques. I also acquired practical knowledge of **real-time data streaming**, **event-aware feature engineering**, and **cloud deployment** using containerization tools like Docker and Kubernetes.

Beyond technical expertise, I improved my skills in project planning, collaboration, and presenting complex results to non-technical stakeholders through effective visualizations and dashboards. This internship enhanced my confidence in bridging the gap between academic learning and industry requirements, and it strengthened my ability to design deployable AI solutions for smart city applications. These learnings will not only support my career in data science and AI but also prepare me to contribute meaningfully to projects that have a direct impact on society.

8 Future work scope

Although the current system successfully demonstrated the feasibility of hybrid traffic forecasting with real-time integration, there remain several areas for further improvement and expansion. One major direction is the incorporation of **reinforcement learning** techniques to enable adaptive traffic signal control, allowing the system to not only forecast but also optimize traffic flow in real-time. Additionally, expanding the deployment from selected junctions to **city-wide coverage** would enhance its practical utility, requiring more advanced distributed processing and edge computing solutions. Another important scope is the integration of **multimodal data sources** such as weather forecasts, CCTV feeds, GPS logs, and public transport schedules to improve model robustness. Cost optimization strategies for large-scale cloud deployment, including **serverless architectures** and **model compression techniques**, can also be explored. Finally, tighter integration with **smart parking systems, ride-sharing platforms, and urban mobility applications** would create a holistic smart city traffic management ecosystem.