**Traffic Telligence: Advanced Traffic Volume Estimation with ML**

**Team Information**

|  |  |
| --- | --- |
| **Student Name** | **Roll Number** |
| **Nandanavanam Nikhil (L)** | **22AK1A3061** |
| **Shaik Naseer Hussain** | **22AK1A3058** |
| **Praveen Kumar** | **22AK5A30** |
| **Kethineni Nikitha** | **22AK1A3063** |

**Phase 1: Brainstorming and Ideation**

**Objective:**

**Problem Statement:**

Urban traffic congestion is a growing challenge, impacting commute times, fuel consumption, and city planning efficiency. Traditional traffic monitoring methods often rely on expensive sensors or manual observation. Traffic Telligence aims to revolutionize traffic volume estimation by leveraging machine learning to analyze real-time video feeds and aerial imagery. This intelligent system can automatically detect, track, and quantify vehicle flow on highways and city roads, offering scalable, cost-effective, and data-driven insights for smart city infrastructure.

**Purpose:**

Traffic Telligence is designed to predict traffic volume based on various influencing factors such as weather conditions, time of day, and seasonal trends using robust machine learning models. The goal is to provide accurate and timely traffic volume estimations that can support smart city initiatives and intelligent transportation systems. By leveraging historical and real-time data, the system can anticipate traffic patterns and help mitigate congestion.

**Impact:**

This project has the potential to revolutionize urban mobility by enabling data-driven traffic management.

**Benefits:**

* Optimize signal timings.
* Plan road maintenance during low-traffic periods.
* Enhance public safety.
* Provide better route suggestions through navigation apps.
* Reduce travel time and vehicle emissions.
* Improve quality of life in urban environments.

**Key Points**

* Accurately predicting traffic volume under varying conditions to reduce congestion.
* Improve road usage efficiency.
* Support proactive traffic management.

**Problems We Aim to Solve**

|  |  |
| --- | --- |
| **Issue** | **Description** |
| Urban congestion | Increased travel time, fuel consumption, pollution |
| Traditional monitoring | Reactive, not predictive |
| Lack of intelligent forecasting | No integration of weather, time, seasonal trends |

**Proposed Solution**

Develop a machine learning-based prediction system that can estimate traffic volume using historical and real-time data.

**Key Features:**

* Data collection from weather, time/date, holidays, past traffic.
* Feature engineering: day of week, season, temperature, rain, snow.
* ML Models: XGBoost, LightGBM, Random Forest.
* Web interface for predictions.
* Continuous improvement with new data.

**Target Users**

|  |  |
| --- | --- |
| **User Group** | **Benefit** |
| Traffic Management Authorities | Optimize traffic flow |
| Urban Planners | Data-driven planning |
| Commuters and Drivers | Early warnings, alternate routes |
| Navigation/Transport Apps | Better route optimization |
| Logistics/Delivery Companies | Improved scheduling |

**Expected Outcomes**

* Real-time traffic congestion analysis.
* Optimized traffic signal timings.
* Long-term traffic pattern insights.

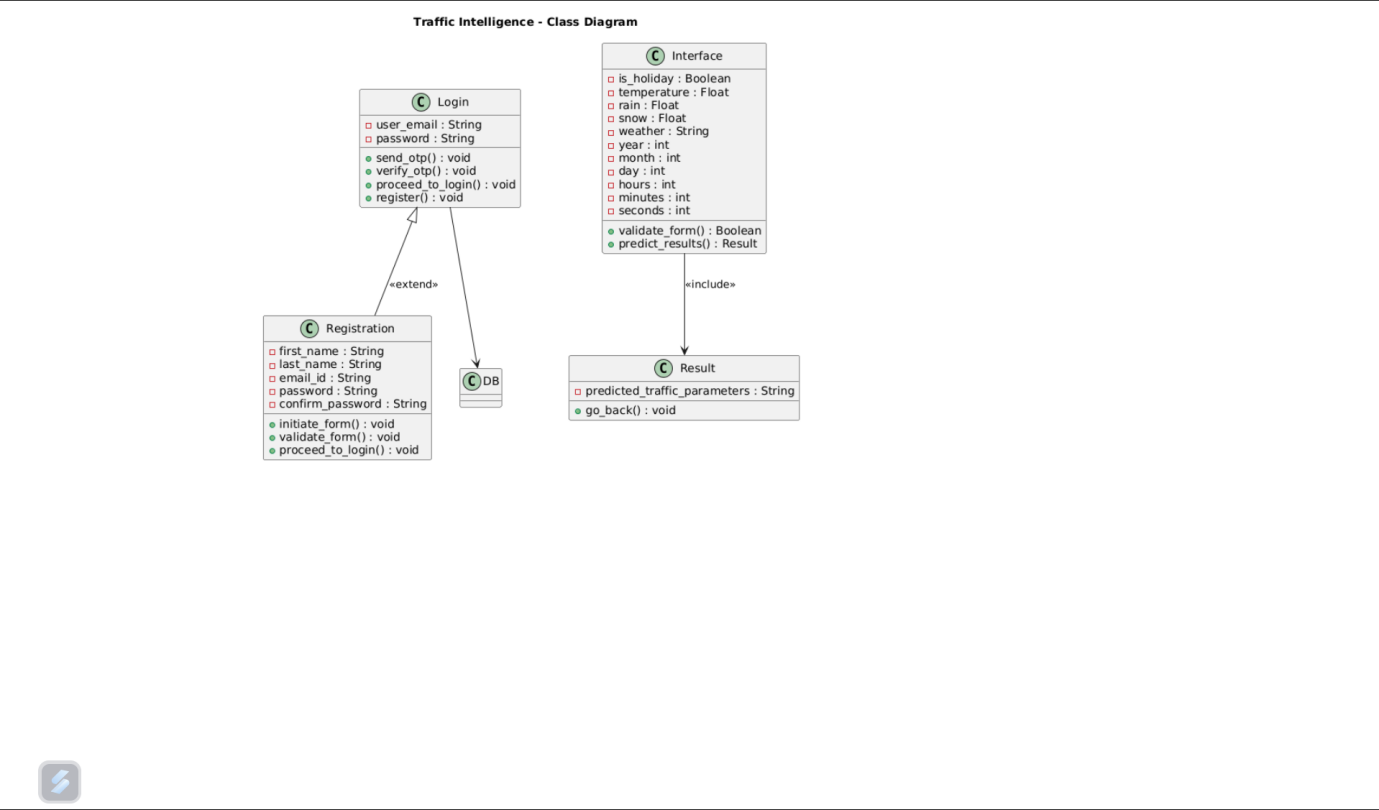
**Phase 2: Requirement Analysis**

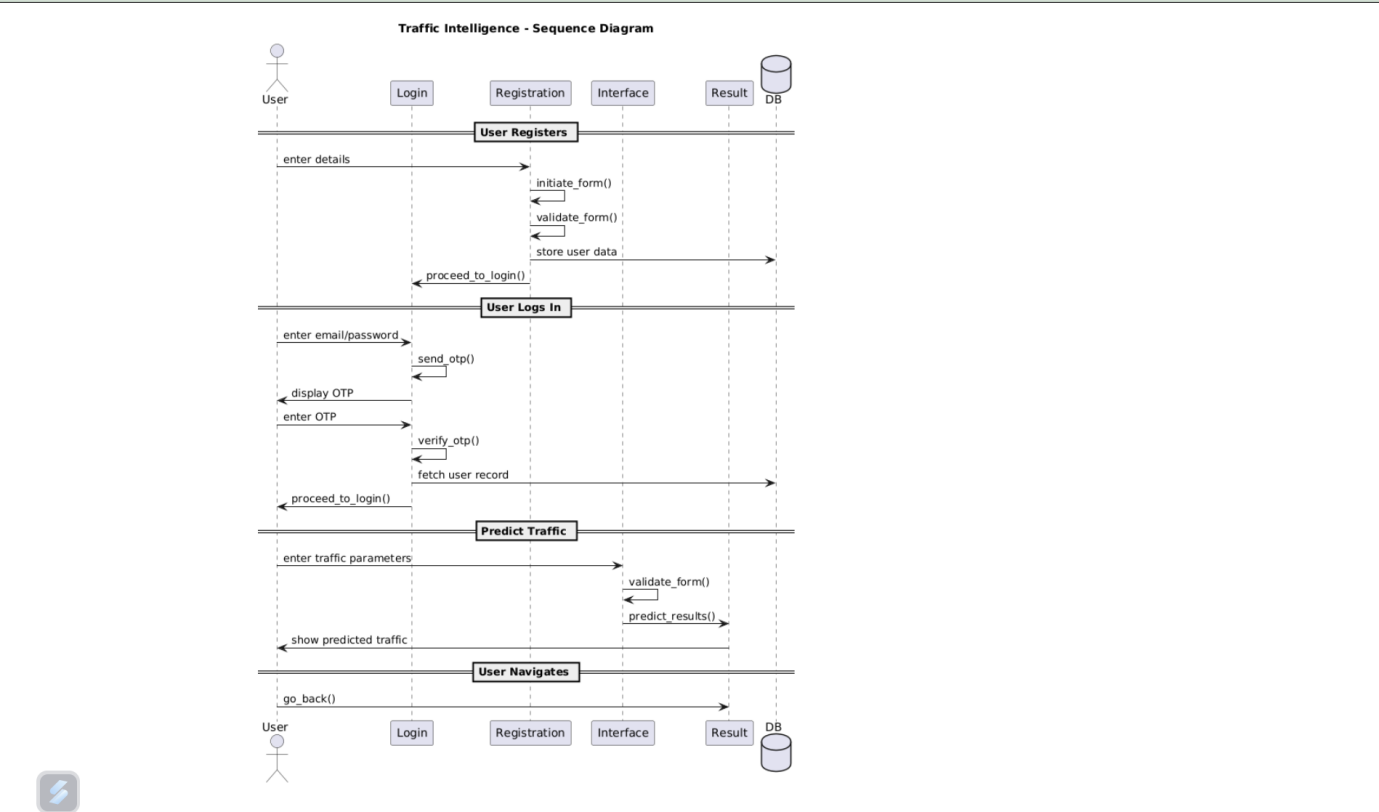
**System Architecture.**

1. **Data Input:**
   * traffic volume.csv (historical data).
2. **Backend:**
   * Flask server (app.py, traffic.py).
   * Pre-trained model (model.pkl, encoder.pkl).
   * API endpoints for predictions.
3. **Frontend:**
   * HTML templates (login.html, register.html, interface.html, dashboard.html, result.html).
   * User inputs weather, date/time, other parameters.
   * Displays predictions.
4. **Model:**
   * Regression model (XGBoost/Random Forest).
   * Encodes categorical variables (encoder.pkl).
5. **User Flow:**
   * User -> Login/Register -> Interface (input form) -> Prediction -> Result Display.

**Phase 3: Project Design**

**UML Diagram**

****

****

**Phase 4: Project Planning (Agile Methodologies).**

| **Sprint** | **Tasks** |
| --- | --- |
| Sprint 1 | Data collection & preprocessing |
| Sprint 2 | Model training & evaluation |
| Sprint 3 | Backend API development |
| Sprint 4 | Frontend design & integration |
| Sprint 5 | Deployment & testing |
| Sprint 6 | Final bug fixing and documentation |

**Phase 5: Project Development.**

**Tech Stack**

|  |  |
| --- | --- |
| **Category** | **Tools/Libraries** |
| Programming Language | Python (backend), HTML/CSS/JavaScript (frontend) |
| ML/Data Science | Pandas, NumPy, Scikit-learn, XGBoost, Matplotlib, Seaborn |
| Web Framework | Flask |
| API Tools | Flask APIs, Pickle for model serialization |
| IDE | Visual Studio Code |
| Version Control | Git |
| Database | MongoDB (planned) |

**Development Process**

|  |  |
| --- | --- |
| **Stage** | **Activities** |
| Data Collection | Import traffic volume dataset (CSV) |
| Data Preprocessing | Handle nulls, encode categories, scale features |
| Model Development | Train regression models (XGBRegressor, Random Forest) |
| Model Evaluation | Evaluate using MAE, RMSE |
| Model Saving | Serialize using Pickle/Joblib |
| Flask App Creation | Build web app to serve ML model |
| Frontend Design | HTML forms (login, register, interface, result) |
| Deployment | Local Flask app or Heroku/Render |
| Testing | Check predictions with real/edge inputs |

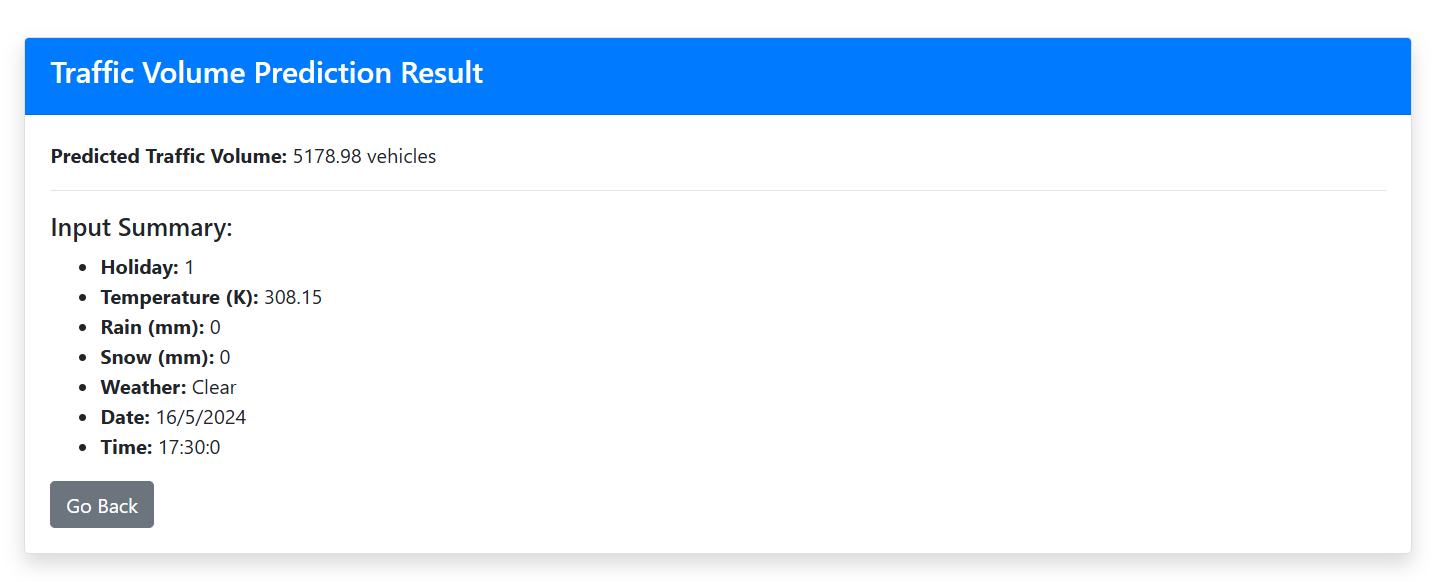
**Challenges and Fixes**

|  |  |
| --- | --- |
| **Challenge** | **Solution** |
| Null values in data | Used Pandas to handle missing values |

**Phase 6: Functional and Performance Testing**

The project which we had done was successfully executed and all the pages aree working properly

* The below page is used to show the results after prediction



* It also gives the input summary in which the user entered for predicting the traffic volume of the new instance.
* We executed this for various test cases and for various scenarios. By changing the specfic fields such as date,time etc. i.e we tried to predict the values of the different scenarios and calculated the results.

**Bugs and improvements**

* Some of the bugs we faced are it showing different output in the application and if we execute the same thing as a single code. To overcome this we changed the logic present in the flask application and rectified that bug i.e showing different values for same input record

**Final Validation:**

* The project which we have developed now satisfies all the initial requirements mentioned.
* In this we also added a feature for user-authentication by using OTP-verification additionally. In this instead of listing the list of holidays we modified that into yes/no i.e if the day which we need to predict the estimated traffic volume is whether holiday or not.

In this page the user will enter the new instance data

