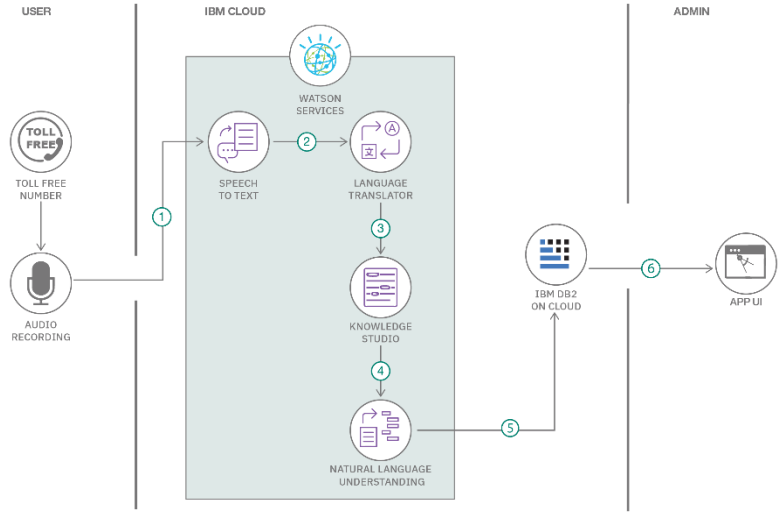
Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	27 June 2025
Team ID	LTVIP2025TMID35510
Project Name	Traffic Telligence Advanced Traffic Volume Estimation With Machine Learning
Maximum Marks	4 Marks

Technical Architecture:



Traffic Telligence Advanced Traffic Volume Estimation With Machine Learning

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web interface to input features and view predictions	HTML, CSS, JavaScript, Bootstrap
2.	Application Logic-1	Handles frontend-backend integration logic	Python (Flask Framework)
3.	Application Logic-2	Weather-based input handling and data augmentation	OpenWeatherMap API Integration (via Python)

4.	Application Logic-3	Model inference and prediction engine	Scikit-learn, XGBoost
5.	Database	Store historical traffic & weather data	SQLite / MySQL (for local or cloud storage)
6.	Cloud Database	Scalable cloud-hosted traffic data storage	AWS RDS, IBM Cloudant
7.	File Storage	Model storage, logs, CSV uploads	Local filesystem / AWS S3 / IBM Block Storage
8.	External API-1	Real-time weather feature integration	OpenWeatherMap API
9.	External API-2	Optional external identifiers for smart city infra	Government APIs (optional, e.g. Aadhar if user-auth required)
10.	Machine Learning Model	Predicts traffic volume based on temporal/weather input	XGBoost, Random Forest, SVR (Pickled Models)
11.	Infrastructure (Server / Cloud)	Application deployment infrastructure	Local (Flask), Cloud (Heroku, AWS EC2, IBM Cloud Foundry)

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Local (Flask), Cloud (Heroku, AWS EC2, IBM Cloud Foundry)	Flask, Pandas, Scikit-learn, XGBoost
2.	Security Implementations	Secure APIs, model files and form inputs	HTTPS, Input Validation, JWT Authentication
3.	Scalable Architecture	Easily scalable across services (UI, API, ML models)	Microservices design (Flask Blueprints + WSGI)
4.	Availability	Cloud-hosted model with HA capability	AWS EC2 + Load Balancer, IBM Cloud Foundry
5.	Performance	Real-time inference < 500ms, pre-loaded models, optional Redis caching	Pickle + Flask, Caching Layer (e.g., Redis)

References:

Architectures | IBM

Reference Architecture Examples and Best Practices

Home | C4 model

How to Draw Useful Technical Architecture Diagrams | by Jimmy Soh | The Internal Startup | Medium