****

**Capstone Project Concept Note and Implementation Plan**

Real-Time Air Quality Prediction and Classification System

**Group – 1 Members**

1. **Mohammad Wasil Jalali**
2. **Bahir Saidi**
3. **Mohammad Aref Rezvan Panah**
4. **Habibullah Farahmand**

Contents

[**Concept Note 3**](#_Toc186983097)

[**1. Project Overview 3**](#_Toc186983098)

[**2. Objectives 3**](#_Toc186983099)

[**3. Background 3**](#_Toc186983100)

[**4. Methodology 3**](#_Toc186983101)

[**5. Architecture Design Diagram 4**](#_Toc186983102)

[**6. Data Sources 5**](#_Toc186983103)

[**7. Literature Review 5**](#_Toc186983104)

[**Implementation Plan 6**](#_Toc186983105)

[**1. Technology Stack 6**](#_Toc186983106)

[**Programming Languages 6**](#_Toc186983107)

[**Libraries and Frameworks 6**](#_Toc186983108)

[**Database and Storage 6**](#_Toc186983109)

[**Deployment and Monitoring 6**](#_Toc186983110)

[**Hardware Components 7**](#_Toc186983111)

[**2. Timeline 7**](#_Toc186983112)

[**Task Distribution 8**](#_Toc186983113)

[**Additional Responsibilities for Collaborative Tasks 9**](#_Toc186983114)

[**Gantt Chart: 11**](#_Toc186983115)

[**3. Milestones 11**](#_Toc186983116)

[**4. Challenges and Mitigations 12**](#_Toc186983117)

[**5. Ethical Considerations 12**](#_Toc186983118)

[**6. References 13**](#_Toc186983119)

# Concept Note

## 1. Project Overview

The **"Real-Time Air Quality Prediction and Classification System"** is a capstone project designed to address the growing challenge of air pollution. This project aligns with **Sustainable Development Goal 3 (Good Health and Well-Being)** and **Goal 13 (Climate Action)** by providing a solution to monitor and predict air quality in real-time. It aims to empower individuals with actionable insights and support policymakers with data-driven recommendations. By using advanced machine learning, this project offers a scalable and user-friendly approach to mitigate health risks and improve environmental awareness, especially in underserved regions.

## 2. Objectives

This project focuses on the following goals:

* **Predict Air Quality Index (AQI):** Build a reliable system to predict AQI based on real-time data.
* **Classify Air Quality Levels:** Translate AQI values into easy-to-understand categories to inform public health decisions.
* **Deliver Personalized Insights:** Provide tailored recommendations, such as when it’s safest to engage in outdoor activities.
* **Support Policy Decisions:** Generate reports and alerts to help authorities take timely action.

## 3. Background

Air pollution is a global issue affecting millions of lives and exacerbating climate change. Current monitoring systems often lack accessibility, real-time updates, or user-centric features. Machine learning, particularly deep learning methods like LSTMs, offers a powerful solution to analyze complex data patterns and make precise predictions. This project aims to bridge the gap between traditional monitoring systems and the need for real-time, actionable, and localized air quality information.

## 4. Methodology

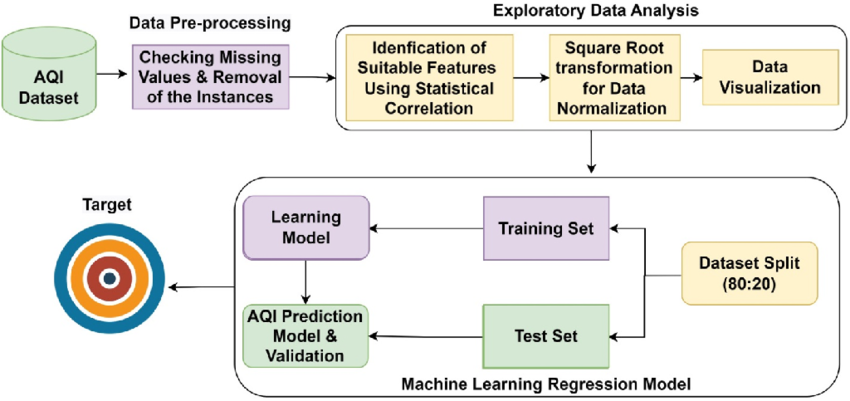
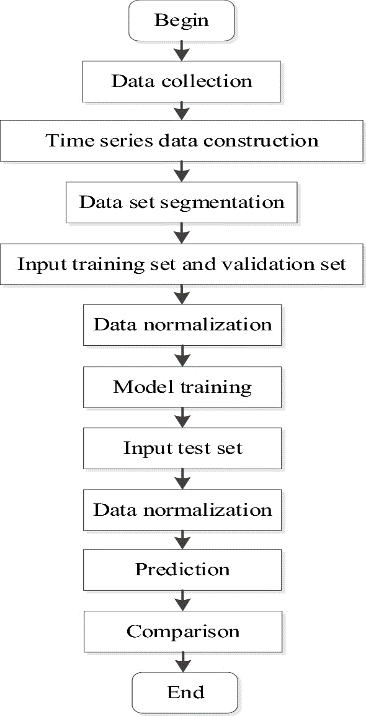
The project combines advanced machine learning techniques to provide accurate and meaningful insights:

* **Data Preprocessing:** Handle missing data using imputation techniques and normalize the dataset for consistent model training.
* **Model Selection:**
  + Use **LSTM models** to account for temporal patterns in air quality data.
  + Integrate **attention mechanisms** to prioritize critical data features for better predictions.
  + Apply **generative AI** to create personalized insights for users.
* **Technologies and Frameworks:** The system will be built using TensorFlow for modeling, Django for backend development, and APIs like OpenWeather for dynamic data updates.

## 5. Architecture Design Diagram

The project architecture consists of the following components:

* **Data Collection:** Gather information from NEPA and OpenWeather API.
* **Preprocessing Unit:** Prepare data by handling gaps, normalizing, and engineering features.
* **Prediction Module:** Utilize LSTM and attention-based models to forecast AQI.
* **Recommendation Engine:** Use generative AI to provide personalized suggestions.
* **User Interface:** Create web and mobile platforms for real-time data visualization.
* **Policy Dashboard:** Design tools for generating detailed reports and alerts for policymakers.



*A visual diagram illustrates how these components interact, emphasizing the flow of data and the integration of predictive models.*

## 6. Data Sources

The project relies on two key data sources:

1. **NEPA Dataset:** Local air quality data from Kabul, which provides historical pollutant levels but requires augmentation due to missing entries.
2. **OpenWeather API:** A global data source offering real-time information about pollutants and weather conditions.

Before modeling, the data will undergo cleaning, normalization, and feature engineering to ensure it meets the requirements for predictive analysis.

## 7. Literature Review

Previous studies have demonstrated the effectiveness of LSTM models for handling time-series data in air quality prediction. Incorporating attention mechanisms has been shown to improve accuracy by focusing on the most critical data points. Generative AI techniques have proven effective in creating user-specific insights. Building on these findings, this project integrates these methodologies into a comprehensive system that prioritizes usability, scalability, and real-world impact. This approach addresses gaps in existing solutions, such as accessibility for underserved regions and support for actionable insights.

# Implementation Plan

## 1. Technology Stack

To build the **"Real-Time Air Quality Prediction and Classification System,"** the following technologies and tools will be utilized:

### Programming Languages

* **Python:** For backend development, machine learning model implementation, and data preprocessing.
* **JavaScript:** For interactive frontend development.

### Libraries and Frameworks

* **Machine Learning and Data Processing:**
  + **TensorFlow/Keras:** To build and train LSTM models and incorporate attention mechanisms.
  + **Pandas & NumPy:** For data manipulation and preprocessing.
  + **Scikit-learn:** For feature selection, evaluation metrics, and baseline modeling.
* **Data Visualization:**
  + **Matplotlib & Seaborn:** For exploratory data analysis (EDA) and visualization of insights.
  + **Plotly/Dash:** For creating interactive visualizations on the user interface.
* **Web Development:**
  + **Django:** Backend framework for building APIs and integrating machine learning models.
  + **JavaScript/HTML/CSS:** For frontend development, ensuring a responsive and user-friendly interface.
* **API Integration:**
  + **Requests:** To fetch real-time data from OpenWeather and other APIs.
  + **Django REST Framework:** For API development and integration with the frontend.

### Database and Storage

* **MySQL:** For storing user data, historical predictions, and application logs.
* **AWS S3:** For secure storage of datasets and trained model files.

### Deployment and Monitoring

* **Docker:** To containerize the application for seamless deployment across environments.
* **AWS EC2/Heroku:** For hosting the application and ensuring scalability.
* **GitHub Actions:** For continuous integration and deployment (CI/CD).
* **Prometheus & Grafana:** For monitoring system performance and uptime.

### Hardware Components

* **Local Machines:** Equipped with GPUs for model training.
* **Cloud Infrastructure:** Cloud-based GPU resources (AWS or Google Cloud) for faster training and scalability during deployment.

# 2. Timeline

The following timeline outlines the stages of the project with detailed tasks, deadlines, and team member responsibilities. The timeline ensures that data collection, preprocessing, model development, training, evaluation, and deployment are carried out efficiently.

|  |  |  |  |
| --- | --- | --- | --- |
| **Stages** | **Month** | **Weeks** | **Task Description** |
| Data Collection & Preprocessing | December | Weeks (2nd - 4th) | - Gather historical air quality data from NEPA and real-time data from OpenWeather API.  - Identify reliable data sources (smart meters, IoT sensors, open datasets).  - Extract data from APIs, CSV files, and databases.  - Verify data quality (check for missing values, errors, and inconsistencies).  - Store data in structured formats (MySQL).  - Handle missing values (imputation techniques).  - Remove duplicates and irrelevant columns.  - Convert data types (timestamps, numerical values).  - Perform feature engineering (time-based and lag features).  - Scale and normalize data (Min-Max Scaling, Standardization).  - Detect and handle outliers (Z-score, IQR method).  - Encode categorical variables (one-hot encoding, label encoding).  - Split data into training (80%), validation (20%), and testing (10%).  - Save the cleaned dataset for model development. |
| Model Development | December- January | Dec(week-4th)  Jan(week-1st) | - Implemented basic models (e.g., ARIMA, XGBoost).  - Developed deep learning models (e.g., LSTM, CNN).  - Perform hyperparameter tuning.  - Implement cross-validation for model evaluation. |
| Model Training & Evaluation | January | Jan weeks  (1st – 2nd ) | - Train initial models on sample data.  - Test models with preliminary evaluation metrics.  **-** Split data into training and test sets  **-** Train models using training data  **-** Evaluate model performance using metrics like accuracy, precision, recall, F1 score  - Fine-tune hyperparameters based on model performance  - Complete training on the full dataset.  - Evaluate models using MAE, RMSE, and MAPE.  - Finalize the best-performing model. |
| Model Deployment | January | Jan weeks  (1st – 4th ) | **-** Develop a web interface (using Django) for deploying the model  **-** Deploy the trained model to the cloud (AWS/Google Cloud)  **-** Test deployment and ensure it works as expected  - Finalize the web interface.  - Containerize the application with Docker.  - Deploy on cloud infrastructure.  - Set up monitoring for model drift. |
| Final Report & Presentation | February | Feb week(1st ) | - Draft initial methodology and architecture sections.  - Finalize the report and include results, visualizations, and analysis.  - Prepare presentation slides for delivery. |

# Task Distribution

|  |  |
| --- | --- |
| **Group Members** | **Tasks** |
| Mohammad Wasil Jalali | - Gather time-series air quality data from APIs.  - Clean and preprocess data (handle missing values, outliers).  - Create temporal features (hour, day, month).  - Perform clustering using K-Means for region identification.  - Develop basic models (ARIMA, XGBoost).  - Evaluate models using metrics (e.g., MAE, RMSE, MAPE).  -Develop deep learning models (LSTM, CNN) for prediction and classification.  - Implement reinforcement learning policies for feature variability handling.  - Test attention mechanisms for feature importance analysis.  - Perform cross-validation and optimize model performance.  - Visualize model results and predictions. |
| Bahir Saidi | - Write formal applications to organization NEPA to request relevant air quality data.  - Assist with data collection and preprocessing.  - Develop the backend to integrate the machine learning models with the Django application.  - Handle database storage for historical data and model outputs.  - Containerize the application using Docker.  - Deploy the application on the cloud (AWS/GCP). |
| Mohammad Aref Rezvan Panah | - Design and implement the frontend using HTML, CSS, and JavaScript.  - Create a user-friendly interface for inputting location and viewing predictions.  - Develop visualization tools (e.g., feature importance graphs, AQI level displays).  - Ensure responsive design for desktop and mobile platforms.  - Set up API integration to fetch real-time air quality data.  - Assist with working on backend of project. |
| Habibullah Farahmand | - Perform literature review to identify gaps and justify the project.  - Write the data gathering and preprocessing section for the report.  - Assist with writing the frontend design section for the report.  - Write the backend and deployment section for the report.  - Assist with working on backend of project.  - Document the progress of the capstone project and compile all assignments/reports.  - Write the introduction, related work, and documentation sections for the report.  - Coordinate final report and presentation submission to meet deadlines. |

**Additional Responsibilities for Collaborative Tasks**

In addition to their individual responsibilities, all members of the capstone project team contribute collectively to the following tasks:

1. **Project Planning and Coordination**
   * Regular team meetings to review progress, discuss challenges, and refine the project plan.
   * Collaboration in defining project goals, milestones, and deliverables.
2. **Literature Review and Research**
   * Joint efforts in researching existing methodologies, frameworks, and best practices relevant to air quality monitoring and time series analysis.
   * Contribution to identifying gaps in the current research and positioning the project's innovation.
3. **Code Review and Debugging**
   * Peer reviewing code to ensure quality, consistency, and adherence to best practices.
   * Collaborative debugging sessions to resolve issues encountered in model development or deployment.
4. **Model Validation and Testing**
   * Joint testing of the prediction and classification models to validate their performance on diverse datasets.
   * Discussion and consensus on the metrics used to evaluate model success.
5. **Documentation and Reporting**
   * Contributing to the final report, ensuring it accurately reflects all aspects of the project.
   * Reviewing documentation for clarity, accuracy, and comprehensiveness.
6. **Presentation Preparation**
   * Collaborative creation of presentation slides and demonstration materials for the final project presentation.
   * Rehearsing the presentation as a team to ensure smooth delivery and coverage of all key points.
7. **Frontend and Backend Integration**
   * Joint efforts to integrate the frontend interface with backend models and APIs for seamless functionality.
   * Testing and debugging the integrated system to ensure usability and reliability.
8. **Data Quality Assurance**
   * Verifying the accuracy, completeness, and consistency of the datasets used.
   * Collaborating on methods to handle missing data, outliers, and noise.
9. **Ethical Review**
   * Ensuring that the project adheres to ethical guidelines, including data privacy and fairness.
   * Discussing potential biases in models and implementing strategies to mitigate them.
10. **Continuous Learning and Skill Sharing**
    * Sharing knowledge and skills within the team, such as tutorials on specific tools or techniques.
    * Supporting each other in learning new technologies or frameworks required for the project.

# Gantt Chart:

#### Here is the Gantt chart for our capstone project. It visually represents the timeline of tasks and their durations, ensuring all milestones and deliverables are clearly outlined.

#### 

# 3. Milestones

* **Milestone 1: Data Collection and Preprocessing (End of Week 2)**  
  The completion of gathering, cleaning, and preparing the necessary data for the project, including extracting temporal features and assigning region IDs.
* **Milestone 2: Initial Model Implementation (End of Week 4)**  
  Developing and testing basic prediction and classification models like ARIMA, XGBoost, and initial CNN/LSTM implementations.
* **Milestone 3: Model Selection and Evaluation (End of Week 5)**  
  Finalizing the best-performing models based on evaluation metrics such as MAE, RMSE, and MAPE, and visualizing their results for presentation.
* **Milestone 4: Web Application Development and Deployment (End of Week 7)**  
  Completion of the web application, including frontend and backend development, containerization with Docker, and integration with real-time data APIs.
* **Milestone 5: Cloud Deployment and Monitoring Setup (End of Week 8)**  
  Deployment of the web application on cloud infrastructure (e.g., AWS, GCP), ensuring scalability, performance testing, and setting up continuous monitoring for model drift.

# 4. Challenges and Mitigations

* **Data Quality**
  + **Challenge:** Missing values, outliers, and noisy time series data may degrade model performance.
  + **Mitigation:** Employ advanced imputation techniques (e.g., interpolation, forward/backward fill) and robust anomaly detection methods to handle inconsistencies effectively.
* **Model Performance**
  + **Challenge:** Overfitting or inadequate generalization of time series models may lead to poor results.
  + **Mitigation:** Utilize cross-validation techniques and test on out-of-sample data. Adopt hybrid or ensemble modeling approaches to enhance robustness.
* **Technical Constraints**
  + **Challenge:** Handling large-scale data and computationally expensive deep learning models.
  + **Mitigation:** Leverage cloud-based infrastructure for scalable computing resources, batch processing for training, and distributed systems for efficiency.

# 5. Ethical Considerations

* **Data Privacy**:  
  The project must ensure that any personal or sensitive information from air quality datasets is anonymized and handled according to data privacy laws and best practices.
* **Bias in Models**:  
  Models should be carefully trained to avoid bias, particularly if certain regions or timeframes are over- or under-represented in the dataset. This ensures fair and unbiased predictions.
* **Community Impact**:  
  As the project targets air quality monitoring, the results and tools developed must be accessible, user-friendly, and aimed at empowering communities to take actionable steps toward improving public health.

# 6. References

1. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). **Scikit-learn: Machine Learning in Python**. *Journal of Machine Learning Research*.
2. Chollet, F. (2015). **Keras: Deep Learning for Humans**.
3. NEPA (National Environmental Protection Agency). (Accessed for data collection and contextual information).
4. OpenWeatherMap API Documentation. (Accessed for real-time data integration).
5. Goodfellow, I., Bengio, Y., & Courville, A. (2016). **Deep Learning**. *MIT Press*.
6. Doshi-Velez, F., & Kim, B. (2017). **Towards a Rigorous Science of Interpretable Machine Learning**. *arXiv preprint arXiv:1702.08608*.