1. import pandas as pd: Imports the pandas library for data manipulation and analysis.
2. import numpy as np: Imports NumPy for numerical operations and linear algebra.
3. from sklearn.multioutput import MultiOutputRegressor: Imports a wrapper to fit multiple regression models.
4. from sklearn.ensemble import RandomForestRegressor: Imports the Random Forest algorithm for regression.
5. from sklearn.model\_selection import train\_test\_split: Imports function to split data into training and test sets.
6. from sklearn.metrics import mean\_squared\_error, r2\_score: Imports metrics to evaluate model performance.
7. df = pd.read\_csv('PB\_All\_2000\_2021.csv', sep=';'): Loads the dataset from a CSV file using semicolon as separator.
8. df.info(): Displays summary information about the DataFrame.
9. df.shape: Returns the number of rows and columns in the DataFrame.
10. df.describe().T: Shows descriptive statistics for all numeric columns (transposed).
11. df.isnull().sum(): Shows the count of missing values in each column.
12. df['date'] = pd.to\_datetime(df['date'], format='%d.%m.%Y'): Converts the 'date' column to datetime format.
13. df = df.sort\_values(by=['id', 'date']): Sorts the DataFrame by station ID and date.
14. df['year'] = df['date'].dt.year: Extracts the year from the date and adds it as a new column.
15. df['month'] = df['date'].dt.month: Extracts the month from the date and adds it as a new column.
16. pollutants = [...]: Defines the target pollutant columns to be predicted.
17. df = df.dropna(subset=pollutants): Drops rows with missing values in pollutant columns.
18. X = df[['id', 'year']]: Selects input features (id and year).
19. y = df[pollutants]: Selects target/output features (pollutants).
20. X\_encoded = pd.get\_dummies(X, columns=['id'], drop\_first=True): Converts 'id' to dummy variables (one-hot encoding).
21. X\_train, X\_test, y\_train, y\_test = train\_test\_split(...): Splits data into training and test sets.
22. model = MultiOutputRegressor(RandomForestRegressor(...)): Initializes a multi-output model using random forest regressors.
23. model.fit(X\_train, y\_train): Trains the model on the training data.
24. y\_pred = model.predict(X\_test): Predicts pollutant levels for the test data.
25. mean\_squared\_error(...): Computes the mean squared error for a predicted pollutant.
26. r2\_score(...): Computes the R² score to assess model accuracy.
27. input\_data = pd.DataFrame({'year': [year\_input], 'id': [station\_id]}): Creates input data for prediction.
28. input\_encoded = pd.get\_dummies(input\_data, columns=['id']): One-hot encodes the input station ID.
29. missing\_cols = set(X\_encoded.columns) - set(input\_encoded.columns): Identifies missing columns in the input.
30. for col in missing\_cols: input\_encoded[col] = 0: Adds missing columns with 0 values to match training data.
31. input\_encoded = input\_encoded[X\_encoded.columns]: Reorders columns to match training set structure.
32. predicted\_pollutants = model.predict(input\_encoded)[0]: Predicts pollutants for the given input data.
33. joblib.dump(model, 'pollution\_model.pkl'): Saves the trained model to a file.
34. joblib.dump(X\_encoded.columns.tolist(), "model\_columns.pkl"): Saves the column structure used in training.
35. print('Model and cols structure are saved!'): Prints confirmation that the model and columns are saved.
36. import pickle  
      
    with open('serialized.pkl', 'rb') as f:    data = pickle.load(f)