PROJECT A: Remaining Useful Life Prediction for Turbofan Engines

I. **Project Overview**

The engine is a heart of an airplane. With the emergence of action cycles, system faults and the deterioration process would inevitably occur. To improve the system's performance and dependability, it is also important to determine how frequently such breakdowns will occur. Prognostics and Health Management (PHM) is a framework for health system management that provides integrated yet individualized approaches. The most complex and technically demanding comprehensive technique in prognosis and health management is aircraft engine prognostics, as it determines an engine's actual health. *The length from the current time to the failure of a system is characterized as remaining useful life (RUL).*

A turbofan engine has many components. A few important components are: Fan, Low Pressure Compressor (LPC), High Pressure Compressor (HPC), Low Pressure Turbine (LPT), High Pressure Turbine (HPT), etc. During operation, degradation occurs in each of the components. If degradation level in any component exceeds a threshold, the engine is said to have failed. We don't want jet engines to fail mid-air. Therefore, jet engines are inspected before every take off. This is a form of periodic maintenance that is not cost effective. But given the critical nature of operation (considering human lives involved), this form of maintenance strategy is justified. Even then, we need a system that can given us early warning if something is going to fail. An early warning, in many cases, may help us prepare for the problem, if not prevent it altogether. A sufficiently early warning will enable us to prevent the disaster. But if an early warning is too conservative (i.e., too early), it will lead to unnecessary waste of money. So, aim of a predictive maintenance system is to predict the RUL as accurately as possible such that it is neither too early nor too late.

II. **Dataset**

The CMAPSS dataset will be used to train the ML models needed for this system. CMAPSS stands for Commercial Modular Aero-Propulsion System Simulation. It is a system developed by NASA to study engine degradation.

Each data set is divided into a train and a test set. Each time-series data set is a data set obtained from another engine. A dataset is data from the same type of engine. The four data sets are multivariate time-series data obtained with different operational conditions (ONE, SIX) and fault modes (ONE, TWO), respectively. There are three operational settings that affect engine performance. For each test data, RUL (Remaining Useful Life) values are provided. In the training set, the degradation increases until it reaches the predefined threshold, which is deemed bad for operating the engine. In the test set, the time series data is terminated before it is completely degraded. For each data set, there is a column of 26 sensors.

For this project, use all the four train and test sets.

III. **Functional Requirements**

We list the functional requirements that the project must satisfy below.

1. The user must be able to send sensor data during the operation of the asset and get an estimate of the remaining useful life.
2. The user must be able to specify the training set to be used to train the model. By default, this should include the entire training set. In addition to this default behavior, the user must be able to specify specific assets to train on by providing unique identifiers for the assets.

3. The user must be able to maintain different versions of the model(s) and specify the version of the model(s) to be used for prediction.

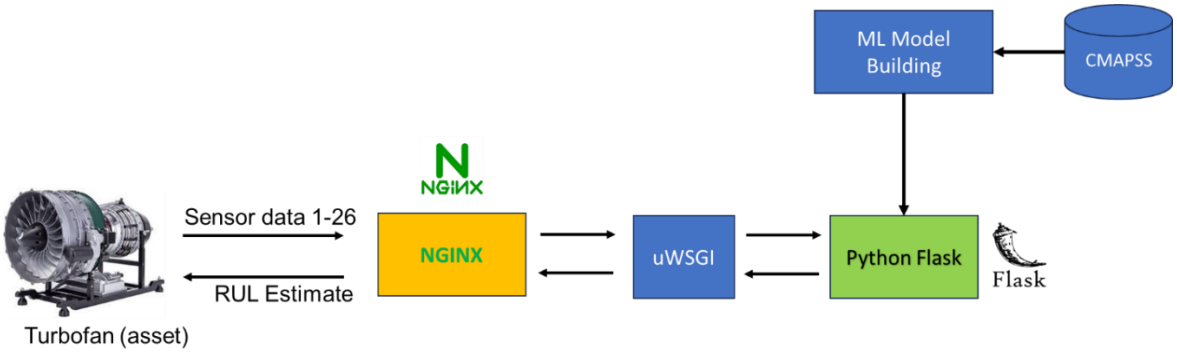
IV. Technical Requirements

From the technical point of view, we will setup a web app to serve ML models learned from the CMAPSS dataset, as shown in the figure below.

1. The models must be built using all the four training datasets.

2. The models must be served using a Flask-based Web app. It can be stood up locally for this project.

3. The client application must be able to send sensor data to receive the RUL prediction.



V. Deliverables

1. *Developed prediction models and test results*: Code to pre-process, train and test the RUL prediction model(s).
2. *Microservice API*: This is the actual web-based REST API developed and stood up locally to serve
3. *Postman environment*: This is the client application that packs the requests and sends a REST call to the web API.
4. *Swagger Documentation*: Document your API using Swagger and deliver the documentation.