**Assignment: Time Series Prediction using RNNs with Keras**

**Objective**: Train a Recurrent Neural Network (RNN) using Keras to predict future values of the Airline Passenger dataset. Submit your predictions in a CSV file along with a Jupyter notebook detailing your exploratory data analysis (EDA), architecture selection, training process, and evaluation.

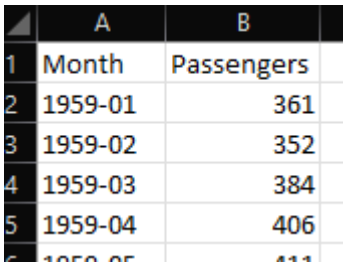
**Dataset**: You will be provided with the Airline Passenger dataset, which contains monthly totals of international airline passengers from January 1949 to December 1960.

**Columns**:

- **Month**: The month and year of the record (e.g., 1949-01, 1949-02, ...).
- **Passengers**: The number of international airline passengers in thousands.

**Tasks**:

1. **Exploratory Data Analysis (EDA)**:
    - Load the dataset using Pandas.
    - Visualize the time series data to understand its patterns, trends, and seasonality.
    - Check for any missing values and decide on a strategy to handle them.
2. **Data Preprocessing**:
    - Normalize or standardize the data using tools like **MinMaxScaler** or **StandardScaler** from Scikit-learn.
    - Convert the time series into a supervised learning problem. For instance, use the past 10 values to predict the next one.
    - Split the data into input sequences (**X**) and their corresponding targets (**y**).
    - Split the data into training and test sets. Use data from January 1949 to December 1958 for training and data from January 1959 to December 1960 for testing.
3. **Architecture Selection using Keras**:
    - Design an RNN architecture using Keras layers like **SimpleRNN**.
    - Decide on the number of hidden units, the type of activation functions, and any regularization techniques (like dropout).
    - Choose an appropriate loss function (e.g., Mean Squared Error) and optimization algorithm.
    - Compile the model using Keras' **compile** method.
4. **Training**:
    - Train your RNN model on the training data using Keras' **fit** method.
    - Monitor the training
    - Implement early stopping or model checkpoints using Keras callbacks to save the best model.

5. **Evaluation**:
    - Use the trained model to predict the values in the test set using Keras' **predict** method.
    - Calculate performance metrics like MSE or MAE to evaluate the model's accuracy on the test set.
    - Visualize the actual vs. predicted values for the test data.
6. **Submission**:
    - Save your predictions for the test set in a CSV file. The file should have two columns: one for the actual values and one for the predicted values.
    - CSV file should contain 2 columns: Month and Passengers!
    - Number of passengers is an INTEGER NUMBER!
    - From January 1959 to December 1960: 24 values!
    - Attached is an example.
    - 

| | A | B |
|---|---|---|
| 1 | Month | Passengers |
| 2 | 1959-01 | 361 |
| 3 | 1959-02 | 352 |
| 4 | 1959-03 | 384 |
| 5 | 1959-04 | 406 |

    - Submit a Jupyter notebook that contains all the steps: EDA, architecture selection, training, and evaluation. Ensure your code is well-commented, and include markdown cells to explain your decisions and observations.

**Evaluation Criteria**:

- Quality of EDA and the insights derived.
- Appropriateness of data preprocessing steps.
- Rationale behind the chosen RNN architecture and its parameters.
- Training strategy and the model's performance on the training data.
- Accuracy of predictions on the test set.
- Clarity and organization of the submitted Jupyter notebook.