Python Program Creation Definition

1. Programming Language

- Python: Chosen for its simplicity, vast ecosystem of libraries, and strong support for data science and machine learning.

2. Modules and Libraries Used

- pandas: For data manipulation and analysis.
- numpy: For numerical operations.
- matplotlib, seaborn: For data visualization (e.g., histograms, boxplots, ROC curves).
- scikit-learn: For machine learning models and utilities like train-test splitting, preprocessing, and evaluation.
- shap: For explaining model predictions using SHAP values.
- fpdf: For exporting the Python program to PDF format.

3. Core Program Components

a. Data Collection

- The dataset is loaded using pandas.read_csv().

b. Data Preprocessing

- Handling Missing Values: Replacing or removing nulls using median imputation.
- Removing Duplicates: Ensures data quality.
- Data Type Consistency: Ensures numerical and datetime features are in the correct format.

c. Encoding Categorical Variables

- Uses LabelEncoder to convert non-numeric features into numeric format.

d. Feature Scaling

- Uses StandardScaler to normalize numerical features, improving model performance.

e. Model Building

- Model Chosen: RandomForestClassifier - suitable for classification tasks and robust to overfitting.
- The model is trained using .fit() on training data.

f. Model Evaluation

- Confusion Matrix: To assess classification performance.
- Classification Report: Precision, recall, F1-score.
- ROC Curve: Graphical representation of true positive vs. false positive rate.
- AUC Score: Evaluates the model's discrimination capacity.

g. Model Interpretation

- SHAP: Visual tool for understanding how each feature contributes to predictions.

4. Deployment (Optional)

- Tools like Streamlit or Flask can be used to create a web-based interface for real-time predictions.

5. Output

- The code is saved and exported to a PDF using the fpdf module.