```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

# Load dataset
df = pd.read_csv("tested.csv")

# ---------------------------
# 🔍 Exploratory Data Analysis
# ---------------------------
print("🔹 Dataset Info:")
print(df.info())
print("\n🔹 Dataset Description:")
print(df.describe(include='all'))
print("\n🔹 Missing Values:")
print(df.isnull().sum())

# Plot Distributions
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
sns.histplot(df['Age'].dropna(), kde=True, color='skyblue')
plt.title('Age Distribution')

plt.subplot(1, 2, 2)
sns.histplot(df['Fare'].dropna(), kde=True, color='salmon')
plt.title('Fare Distribution')
plt.tight_layout()
plt.show()
```

```
# Boxplots for Outliers
plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
sns.boxplot(y=df['Age'], color='lightblue')
plt.title('Boxplot of Age')

plt.subplot(1, 2, 2)
sns.boxplot(y=df['Fare'], color='lightcoral')
plt.title('Boxplot of Fare')
plt.tight_layout()
plt.show()

# Countplots for Categorical
plt.figure(figsize=(18, 5))
plt.subplot(1, 3, 1)
sns.countplot(x='Sex', data=df, palette='pastel')
plt.title('Sex Distribution')

plt.subplot(1, 3, 2)
sns.countplot(x='Pclass', data=df, palette='Set2')
plt.title('Passenger Class Distribution')

plt.subplot(1, 3, 3)
sns.countplot(x='Embarked', data=df, palette='Set3')
plt.title('Port of Embarkation Distribution')
plt.tight_layout()
plt.show()

# Correlation Heatmap
numeric_cols = df.select_dtypes(include=['int64', 'float64']).drop(columns=['PassengerId'])
```

```
plt.figure(figsize=(10, 6))
sns.heatmap(numeric_cols.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()

# ---------------------------
# 🧩 Imputation & Feature Engineering
# ---------------------------

# Handle missing values
df['Age'].fillna(df['Age'].median(), inplace=True)
df['Fare'].fillna(df['Fare'].median(), inplace=True)
df.drop(columns=['Cabin'], inplace=True)

# Extract Title
df['Title'] = df['Name'].str.extract(' ([A-Za-z]+)\.', expand=False)
df['Title'] = df['Title'].replace(['Lady', 'Countess', 'Capt', 'Col', 'Don', 'Dr',
                                   'Major', 'Rev', 'Sir', 'Jonkheer', 'Dona'], 'Rare')
df['Title'] = df['Title'].replace({'Mlle': 'Miss', 'Ms': 'Miss', 'Mme': 'Mrs'})

# Create FamilySize & IsAlone
df['FamilySize'] = df['SibSp'] + df['Parch'] + 1
df['IsAlone'] = (df['FamilySize'] == 1).astype(int)

# One-hot encode categorical columns
df = pd.get_dummies(df, columns=['Sex', 'Embarked', 'Title'], drop_first=True)

# Drop unneeded columns
df.drop(columns=['Name', 'Ticket'], inplace=True)

# ---------------------------
```

```
# 🤖 Logistic Regression Modeling
# ----------------------------

# Prepare data
X = df.drop(['Survived', 'PassengerId'], axis=1)
y = df['Survived']

# Train/Test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
print("\n 🔹 Model Accuracy:", accuracy_score(y_test, y_pred))
print("\n 🔹 Classification Report:\n", classification_report(y_test, y_pred))
```