Colab Code for ML Model:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

# ---------------------------------------------------------
# 1. SETUP & DATA ENTRY
# ---------------------------------------------------------

# Standard Yield Strengths (MPa)
yield_strengths = {
    "Ti-6Al-4V": 880,
    "Co-Cr Alloy": 570,
    "316 Stainless": 290
}

# --- IMPORTANT: SAFETY FACTOR ---
# Hum Safety Factor 2.0 use kar rahe hain taaki ML model ko
# "Safe" aur "Unsafe" dono tarah ka data mile.
# Limit = Yield Strength / 2.0
safety_factor = 2.0

# --- DATASETS (Accurate Data from Ansys Images) ---

# 1. BEARING DATA
bearing_data = [
    ["DP 0", 30, 20, 100, 5000, "Ti-6Al-4V", 66.58],
    ["DP 1", 32, 22, 105, 8000, "Ti-6Al-4V", 72.10],
    ["DP 2", 28, 18, 95, 4000, "Ti-6Al-4V", 54.858],
    ["DP 3", 35, 25, 110, 12000, "Ti-6Al-4V", 110.20],
    ["DP 4", 30, 15, 100, 6000, "Ti-6Al-4V", 117.63],
    ["DP 5", 25, 20, 98, 10000, "Ti-6Al-4V", 80.381],
    ["DP 6", 34, 24, 108, 15000, "Ti-6Al-4V", 125.40],
    ["DP 7", 30, 20, 100, 18000, "Co-Cr Alloy", 145.20],
    ["DP 8", 32, 22, 105, 10000, "Co-Cr Alloy", 98.50],
    ["DP 9", 28, 18, 95, 25000, "Co-Cr Alloy", 190.80],
    ["DP 10", 35, 25, 110, 30000, "Co-Cr Alloy", 210.50],
    ["DP 11", 30, 15, 100, 8000, "Co-Cr Alloy", 75.30],
    ["DP 12", 25, 20, 98, 22000, "Co-Cr Alloy", 160.10],
    ["DP 13", 34, 24, 108, 15000, "Co-Cr Alloy", 120.40],
    ["DP 14", 30, 20, 100, 8000, "316 Stainless", 70.20],
    ["DP 15", 32, 22, 105, 5000, "316 Stainless", 58.40],
    ["DP 16", 28, 18, 95, 15000, "316 Stainless", 130.50],
    ["DP 17", 35, 25, 110, 9000, "316 Stainless", 95.60],
    ["DP 18", 30, 15, 100, 4000, "316 Stainless", 48.90],
    ["DP 19", 25, 20, 98, 11000, "316 Stainless", 105.30],
    ["DP 20", 34, 24, 108, 15000, "316 Stainless", 125.40]
]
```

```
df_bearing = pd.DataFrame(bearing_data, columns=["Name", "P1", "P2", "P3",
"Force", "Material", "Stress"])
df_bearing['Part'] = 'Bearing'

# 2. GEAR DATA
gear_data = [
    ["DP 0", 30, 20, 100, 5000, "Ti-6Al-4V", 66.58],
    ["DP 1", 28, 18, 96, 8000, "Ti-6Al-4V", 118.51],
    ["DP 2", 28, 18, 95, 4000, "Ti-6Al-4V", 54.858],
    ["DP 3", 26, 16, 94, 12000, "Ti-6Al-4V", 178.91],
    ["DP 4", 30, 15, 100, 6000, "Ti-6Al-4V", 117.63],
    ["DP 5", 25, 20, 98, 10000, "Ti-6Al-4V", 80.381],
    ["DP 6", 29, 19, 98, 15000, "Ti-6Al-4V", 191.48],
    ["DP 7", 30, 20, 100, 5000, "Co-Cr Alloy", 66.932],
    ["DP 8", 28, 18, 96, 10000, "Co-Cr Alloy", 149.92],
    ["DP 9", 25, 15, 90, 25000, "Co-Cr Alloy", 481.97],
    ["DP 10", 32, 19, 98, 30000, "Co-Cr Alloy", 432.62],
    ["DP 11", 26, 16, 94, 8000, "Co-Cr Alloy", 119.9],
    ["DP 12", 29, 20, 99, 22000, "Co-Cr Alloy", 305.08],
    ["DP 13", 27, 17, 92, 15000, "Co-Cr Alloy", 187.96],
    ["DP 14", 30, 20, 100, 5000, "316 Stainless", 67.221],
    ["DP 15", 30, 20, 100, 8000, "316 Stainless", 107.55],
    ["DP 16", 28, 18, 95, 5000, "316 Stainless", 70.072],
    ["DP 17", 32, 19, 98, 15000, "316 Stainless", 219.22],
    ["DP 18", 25, 15, 92, 9000, "316 Stainless", 182.36],
    ["DP 19", 29, 16, 96, 4000, "316 Stainless", 70.906],
    ["DP 20", 26, 14, 94, 11000, "316 Stainless", 275.63]
]
df_gear = pd.DataFrame(gear_data, columns=["Name", "P1", "P2", "P3", "Force",
"Material", "Stress"])
df_gear['Part'] = 'Gear'

# 3. NUT DATA
nut_data = [
    ["DP 0", 20, 24, 22, 10000, "Ti-6Al-4V", 126.35],
    ["DP 1", 18, 22, 22, 15000, "Ti-6Al-4V", 184.84],
    ["DP 2", 15, 26, 22, 8000, "Ti-6Al-4V", 156.09],
    ["DP 3", 22, 24, 24, 20000, "Ti-6Al-4V", 210.8],
    ["DP 4", 16, 20, 20, 12000, "Ti-6Al-4V", 198.88],
    ["DP 5", 14, 28, 24, 5000, "Ti-6Al-4V", 105.68],
    ["DP 6", 25, 24, 22, 25000, "Ti-6Al-4V", 236.89],
    ["DP 7", 20, 24, 22, 18000, "Co-Cr Alloy", 224.83],
    ["DP 8", 18, 25, 21, 10000, "Co-Cr Alloy", 150.14],
    ["DP 9", 24, 22, 25, 30000, "Co-Cr Alloy", 274],
    ["DP 10", 12, 28, 22, 6000, "Co-Cr Alloy", 156.56],
    ["DP 11", 20, 24, 23, 22000, "Co-Cr Alloy", 268.31],
    ["DP 12", 15, 20, 18, 8000, "Co-Cr Alloy", 151.2],
    ["DP 13", 22, 26, 24, 25000, "Co-Cr Alloy", 266.73],
    ["DP 14", 20, 24, 22, 18000, "316 Stainless", 223.42],
    ["DP 15", 20, 24, 22, 8000, "316 Stainless", 99.299],
    ["DP 16", 18, 22, 20, 5000, "316 Stainless", 72.974],
    ["DP 17", 25, 26, 24, 15000, "316 Stainless", 134.14],
    ["DP 18", 16, 25, 22, 9000, "316 Stainless", 154.63],
    ["DP 19", 14, 28, 25, 4000, "316 Stainless", 81.363],
    ["DP 20", 12, 24, 22, 11000, "316 Stainless", 278.76]
]
df_nut = pd.DataFrame(nut_data, columns=["Name", "P1", "P2", "P3", "Force",
"Material", "Stress"])
df_nut['Part'] = 'Nut'

# 4. FLANGE DATA
flange_data = [
    ["DP 0", 220, 22, -5000, "Ti-6Al-4V", 5.3162],
    ["DP 1", 210, 15, -8000, "Ti-6Al-4V", 15.497],
    ["DP 2", 200, 12, -12000, "Ti-6Al-4V", 35.626],
    ["DP 3", 230, 25, -15000, "Ti-6Al-4V", 8.7244],
    ["DP 4", 215, 18, -20000, "Ti-6Al-4V", 29.588],
    ["DP 5", 225, 20, -25000, "Ti-6Al-4V", 32.721],
    ["DP 6", 205, 10, -5000, "Ti-6Al-4V", 20.357],
    ["DP 7", 195, 8, -4000, "Ti-6Al-4V", 25.88],
    ["DP 8", 220, 20, -5000, "Co-Cr Alloy", 6.4966],
    ["DP 9", 220, 10, -6000, "Co-Cr Alloy", 24.42],
    ["DP 10", 210, 8, -9000, "Co-Cr Alloy", 58.888],
    ["DP 11", 200, 6, -4000, "Co-Cr Alloy", 45.503],
    ["DP 12", 240, 15, -30000, "Co-Cr Alloy", 62.222],
    ["DP 13", 218, 9, -18000, "Co-Cr Alloy", 92.367],
    ["DP 14", 222, 11, -22000, "Co-Cr Alloy", 73.221],
    ["DP 15", 195, 7, -10000, "Co-Cr Alloy", 83.82],
    ["DP 16", 220, 20, -5000, "316 Stainless", 6.6357],
    ["DP 17", 220, 10, -5000, "316 Stainless", 21.368],
    ["DP 18", 215, 9, -11000, "316 Stainless", 57.867],
    ["DP 19", 205, 8, -16000, "316 Stainless", 107.84],
    ["DP 20", 235, 14, -28000, "316 Stainless", 67.509],
    ["DP 21", 200, 6, -8000, "316 Stainless", 95.793],
    ["DP 22", 225, 12, -24000, "316 Stainless", 76.117],
    ["DP 23", 210, 18, -3500, "316 Stainless", 5.4793]
]
df_flange = pd.DataFrame(flange_data, columns=["Name", "P1", "P2",
"Force_Raw", "Material", "Stress"])
df_flange['Force'] = df_flange['Force_Raw'].abs()
df_flange['P3'] = 0
df_flange['Part'] = 'Flange'
df_flange = df_flange[["Name", "P1", "P2", "P3", "Force", "Material",
"Stress", "Part"]]

# Combine All
df_all = pd.concat([df_bearing, df_gear, df_nut, df_flange],
ignore_index=True)

# ------------------------------------------------------------
# 2. DATA PROCESSING & LABELING (With Safety Factor)
# ------------------------------------------------------------

def check_safety(row):
    # Limit is now half of the Yield Strength (Safety Factor 2.0)
    limit = yield_strengths[row['Material']] / safety_factor
    if row['Stress'] < limit:
        return 1 # Safe
    else:
        return 0 # Unsafe

df_all['Label'] = df_all.apply(check_safety, axis=1)
df_all['Label_Text'] = df_all['Label'].map({1: 'Safe', 0: 'Unsafe'})

# Encode for ML
df_all['Material_Code'] = df_all['Material'].astype('category').cat.codes
df_all['Part_Code'] = df_all['Part'].astype('category').cat.codes

print("--- Data Summary (After Safety Factor 2.0) ---")
print(df_all.groupby(['Part', 'Label_Text']).size())

# ---------------------------------------------------------
# 3. MACHINE LEARNING (Random Forest)
# ---------------------------------------------------------

# Features: Dimensions (P1, P2), Force, Material
X = df_all[['P1', 'P2', 'Force', 'Material_Code', 'Part_Code']]
# Note: Hum "Stress" ko input nahi de rahe, kyunke Stress to result hai.
# Hum model ko Dimension aur Force se predict karwana chahte hain.
y = df_all['Label']

# Train/Test Split
# Stratify ensures both Safe and Unsafe labels are in the test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42, stratify=y)

clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

print("\n--- Machine Learning Classification Report ---")
try:
    print(classification_report(y_test, y_pred, target_names=['Unsafe',
'Safe']))
except ValueError:
    print("Warning: Test set might still not have enough classes, but data
processing shows mix.")

# ---------------------------------------------------------
# 4. ANSWERING "KAUNSA SAFE/UNSAFE AUR KYUN?"
# ---------------------------------------------------------

print("\n" + "="*50)
print("   DETAILED FAILURE ANALYSIS (List of Unsafe Designs)")
print("="*50)
unsafe_df = df_all[df_all['Label'] == 0][['Part', 'Name', 'Material',
'Force', 'Stress']]
```

```
print(unsafe_df.to_string())
print("\nNote: These failed because Stress > (Yield Strength / 2.0)")

# ---------------------------------------------------------
# 5. VISUALIZATIONS (Graphs)
# ---------------------------------------------------------

sns.set_style("whitegrid")
plt.rcParams.update({'font.size': 12})

# GRAPH 1: FEATURE IMPORTANCE (The "Why" Graph)
plt.figure(figsize=(10, 6))
importances = clf.feature_importances_
feature_names = X.columns
indices = np.argsort(importances)[::-1]

sns.barplot(x=[feature_names[i] for i in indices], y=importances[indices],
palette="viridis")
plt.title('Why did it fail? (Feature Importance)', fontsize=16,
fontweight='bold')
plt.xlabel('Design Parameters')
plt.ylabel('Importance Score')
plt.show()

# GRAPH 2: 3D SCATTER PLOT
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, projection='3d')
colors = df_all['Label_Text'].map({'Safe': 'green', 'Unsafe': 'red'})
sc = ax.scatter(df_all['P1'], df_all['Force'], df_all['Stress'], c=colors,
s=60, edgecolors='k', alpha=0.8)
ax.set_xlabel('Dimension P1 (mm)')
ax.set_ylabel('Force (N)')
ax.set_zlabel('Stress (MPa)')
ax.set_title('3D View: Safe (Green) vs Unsafe (Red)', fontsize=14,
fontweight='bold')
plt.show()

# GRAPH 3: SAFETY COUNT
plt.figure(figsize=(8, 5))
sns.countplot(x='Part', hue='Label_Text', data=df_all, palette={'Safe':
'green', 'Unsafe': 'red'})
plt.title('Count of Safe vs Unsafe Designs per Part (FoS=2.0)')
plt.show()
```