Thyroid Cancer Risk Factors and Prognosis

Type Desription:

This dataset contains medical and lifestyle factors associated with thyroid cancer risk, diagnosis, and prognosis. It includes patient demographics, clinical history, genetic predispositions, lifestyle habits, tumor characteristics, treatment details, and survival outcomes.

The dataset is structured to support predictive modeling, statistical analysis, and machine learning applications for risk assessment, early detection, and treatment outcome predictions.

Potential Features:

Patient Information: Age, Gender, BMI

Medical History: Family History of Thyroid Cancer, Previous Cancers, Autoimmune Disorders (e.g., Hashimoto’s Thyroiditis)

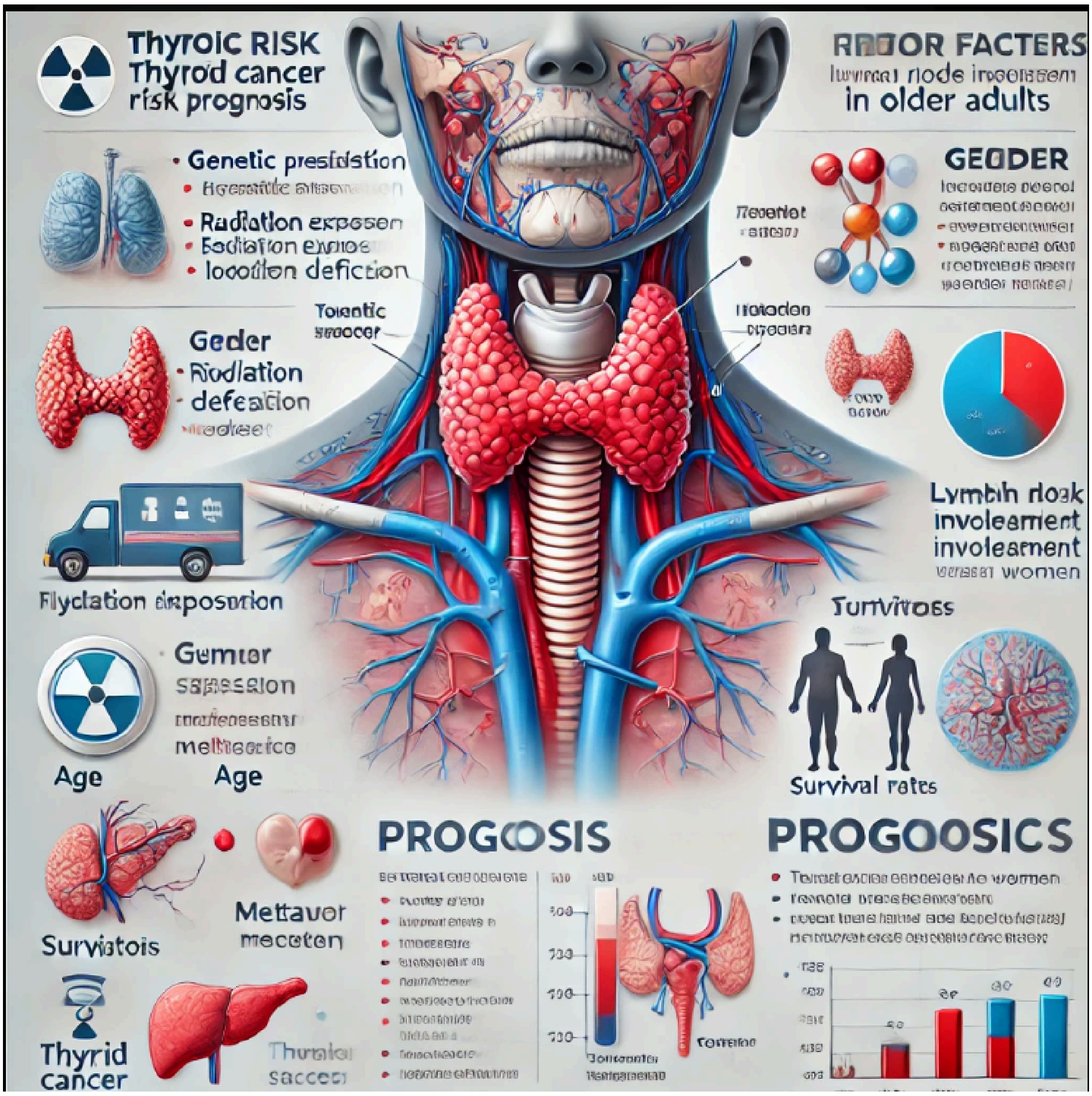
Genetic Factors: Presence of RET/PTC, BRAF, and other relevant mutations

Lifestyle Factors: Smoking Status, Alcohol Consumption, Iodine Intake, Radiation Exposure

Clinical Presentation: Tumor Size, Tumor Type (Papillary, Follicular, Medullary, Anaplastic), Nodule Characteristics, Symptom Severity

Diagnosis Details: Fine-Needle Aspiration Biopsy (FNAB) Results, Ultrasound Findings, Diagnosis Delay in Days

Treatment & Prognosis: Surgery Type, Radioactive Iodine Therapy, Chemotherapy, Recurrence Rate, Survival Years After Diagnosis



Import Libraries

In [5]:
```python
import seaborn as sns
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

Import Dataset

In [8]:
```python
df = pd.read_csv(r"C:\Users\chitt\Downloads\thyroid_cancer_risk_data.csv")
```

In [10]:
```python
df
```

Out[10]:

	Patient_ID	Age	Gender	Country	Ethnicity	Family_History	Radiation_Exposu
0	1	66	Male	Russia	Caucasian	No	
1	2	29	Male	Germany	Hispanic	No	
2	3	86	Male	Nigeria	Caucasian	No	
3	4	75	Female	India	Asian	No	
4	5	35	Female	Germany	African	Yes	
...	...	...	...	...	...	...	
212686	212687	58	Female	India	Asian	No	
212687	212688	89	Male	Japan	Middle Eastern	No	
212688	212689	72	Female	Nigeria	Hispanic	No	
212689	212690	85	Female	Brazil	Middle Eastern	No	
212690	212691	46	Female	Japan	Middle Eastern	No	

212691 rows × 17 columns

In [12]:

```
df.shape
```

Out[12]:

```
(212691, 17)
```

In [14]:

```
df.head()
```

Out[14]:

	Patient_ID	Age	Gender	Country	Ethnicity	Family_History	Radiation_Exposure	Ic
0	1	66	Male	Russia	Caucasian	No	Yes	
1	2	29	Male	Germany	Hispanic	No	Yes	
2	3	86	Male	Nigeria	Caucasian	No	No	
3	4	75	Female	India	Asian	No	No	
4	5	35	Female	Germany	African	Yes	Yes	

In [16]:

```
df.tail()
```

Out[16]:

	Patient_ID	Age	Gender	Country	Ethnicity	Family_History	Radiation_Exposu
212686	212687	58	Female	India	Asian	No	N
212687	212688	89	Male	Japan	Middle Eastern	No	N
212688	212689	72	Female	Nigeria	Hispanic	No	N
212689	212690	85	Female	Brazil	Middle Eastern	No	N
212690	212691	46	Female	Japan	Middle Eastern	No	N

In [20]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 212691 entries, 0 to 212690
Data columns (total 17 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   Patient_ID           212691 non-null  int64  
 1   Age                  212691 non-null  int64  
 2   Gender               212691 non-null  object 
 3   Country              212691 non-null  object 
 4   Ethnicity            212691 non-null  object 
 5   Family_History       212691 non-null  object 
 6   Radiation_Exposure   212691 non-null  object 
 7   Iodine_Deficiency    212691 non-null  object 
 8   Smoking              212691 non-null  object 
 9   Obesity              212691 non-null  object 
 10  Diabetes             212691 non-null  object 
 11  TSH_Level            212691 non-null  float64
 12  T3_Level             212691 non-null  float64
 13  T4_Level             212691 non-null  float64
 14  Nodule_Size          212691 non-null  float64
 15  Thyroid_Cancer_Risk  212691 non-null  object 
 16  Diagnosis            212691 non-null  object 
dtypes: float64(4), int64(2), object(11)
memory usage: 27.6+ MB
```

In [22]:

```
df.describe()
```

Out[22]:

	Patient_ID	Age	TSH_Level	T3_Level	T4_Level	No
count	212691.00000	212691.000000	212691.000000	212691.000000	212691.000000	2126
mean	106346.00000	51.918497	5.045102	2.001727	8.246204	
std	61398.74739	21.632815	2.860264	0.866248	2.164188	
min	1.00000	15.000000	0.100000	0.500000	4.500000	
25%	53173.50000	33.000000	2.570000	1.250000	6.370000	
50%	106346.00000	52.000000	5.040000	2.000000	8.240000	
75%	159518.50000	71.000000	7.520000	2.750000	10.120000	
max	212691.00000	89.000000	10.000000	3.500000	12.000000	

In [24]:

```
df.corr
```

Out[24]:
```
<bound method DataFrame.corr of         Patient_ID  Age  Gender  Country
Ethnicity Family_History  \
0                1   66    Male   Russia       Caucasian             No
1                2   29    Male  Germany        Hispanic             No
2                3   86    Male  Nigeria       Caucasian             No
3                4   75  Female    India           Asian             No
4                5   35  Female  Germany         African            Yes
...            ...  ...     ...      ...             ...            ...
212686      212687   58  Female    India           Asian             No
212687      212688   89    Male    Japan  Middle Eastern             No
212688      212689   72  Female  Nigeria        Hispanic             No
212689      212690   85  Female   Brazil  Middle Eastern             No
212690      212691   46  Female    Japan  Middle Eastern             No

       Radiation_Exposure Iodine_Deficiency Smoking Obesity Diabetes  \
0                     Yes                No      No      No       No
1                     Yes                No      No      No       No
2                      No                No      No      No       No
3                      No                No      No      No       No
4                     Yes                No      No      No       No
...                   ...               ...     ...     ...      ...
212686                 No                No      No     Yes       No
212687                 No                No      No     Yes       No
212688                 No                No      No      No      Yes
212689                 No                No      No      No      Yes
212690                 No                No     Yes      No       No

        TSH_Level  T3_Level  T4_Level  Nodule_Size Thyroid_Cancer_Risk  \
0            9.37      1.67      6.16         1.08                 Low
1            1.83      1.73     10.54         4.05                 Low
2            6.26      2.59     10.57         4.61                 Low
3            4.10      2.62     11.04         2.46              Medium
4            9.10      2.11     10.71         2.11                High
...           ...       ...       ...          ...                 ...
212686       2.00      0.64     11.92         1.48                 Low
212687       9.77      3.25      7.30         4.46              Medium
212688       7.72      2.44      8.71         2.36              Medium
212689       5.62      2.53      9.62         1.54              Medium
212690       5.60      2.73     10.59         2.53                 Low

        Diagnosis
0          Benign
1          Benign
2          Benign
3          Benign
4          Benign
...           ...
212686     Benign
212687     Benign
212688     Benign
212689     Benign
212690  Malignant

[212691 rows x 17 columns]>
```

In [26]:
```
df.isnull().sum()
```

Out[26]:
```
Patient_ID            0
Age                   0
Gender                0
Country               0
Ethnicity             0
Family_History        0
Radiation_Exposure    0
Iodine_Deficiency     0
Smoking               0
Obesity               0
Diabetes              0
TSH_Level             0
T3_Level              0
T4_Level              0
Nodule_Size           0
Thyroid_Cancer_Risk   0
Diagnosis             0
dtype: int64
```

In [28]:
```
df.duplicated()
```

Out[28]:
```
0         False
1         False
2         False
3         False
4         False
          ...
212686    False
212687    False
212688    False
212689    False
212690    False
Length: 212691, dtype: bool
```

In [30]:
```
df.columns
```

Out[30]:
```
Index(['Patient_ID', 'Age', 'Gender', 'Country', 'Ethnicity', 'Family_History',
       'Radiation_Exposure', 'Iodine_Deficiency', 'Smoking', 'Obesity',
       'Diabetes', 'TSH_Level', 'T3_Level', 'T4_Level', 'Nodule_Size',
       'Thyroid_Cancer_Risk', 'Diagnosis'],
      dtype='object')
```

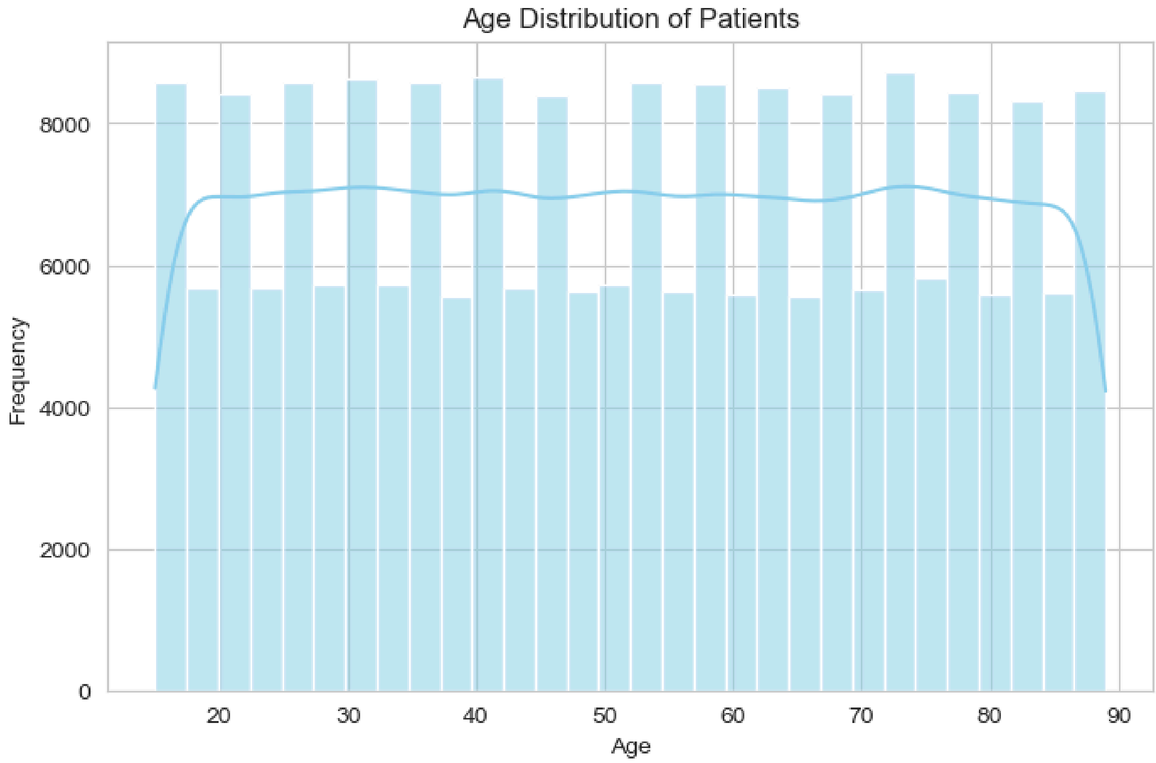
In [32]:
```
df["Family_History"] = df["Family_History"].map({"Yes": 1, "No": 0})  # Adjust b
df["Thyroid_Cancer_Risk"] = pd.to_numeric(df["Thyroid_Cancer_Risk"], errors="coe
```

Data Visualizations

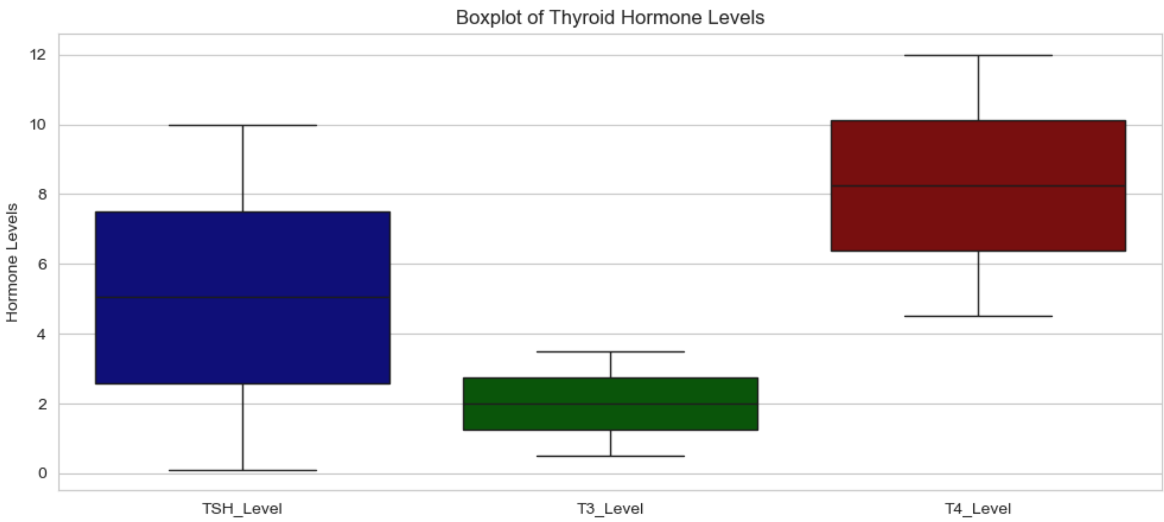
In [35]:
```
# Set style
sns.set_style("whitegrid")
plt.rcParams["figure.figsize"] = (10, 6)
```

In [37]:
```
# 1. Age Distribution
plt.figure(figsize=(8, 5))
sns.histplot(df["Age"], kde=True, bins=30, color="skyblue")
plt.title("Age Distribution of Patients")
plt.xlabel("Age")
```

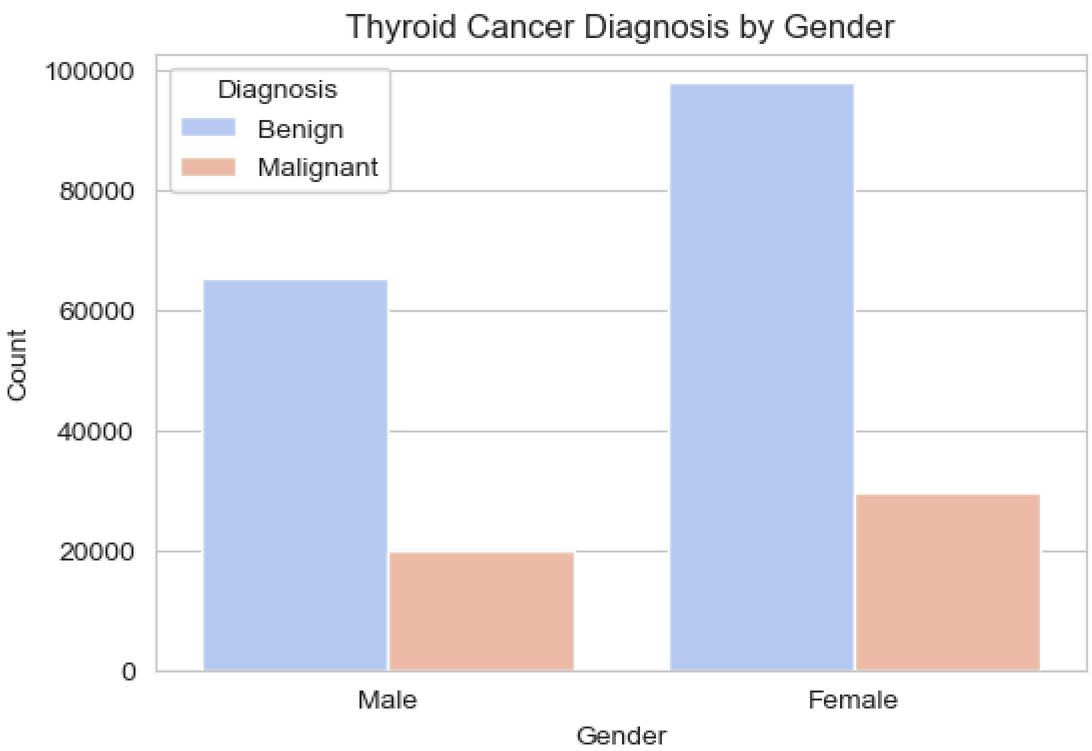
```
plt.ylabel("Frequency")
plt.show()
```



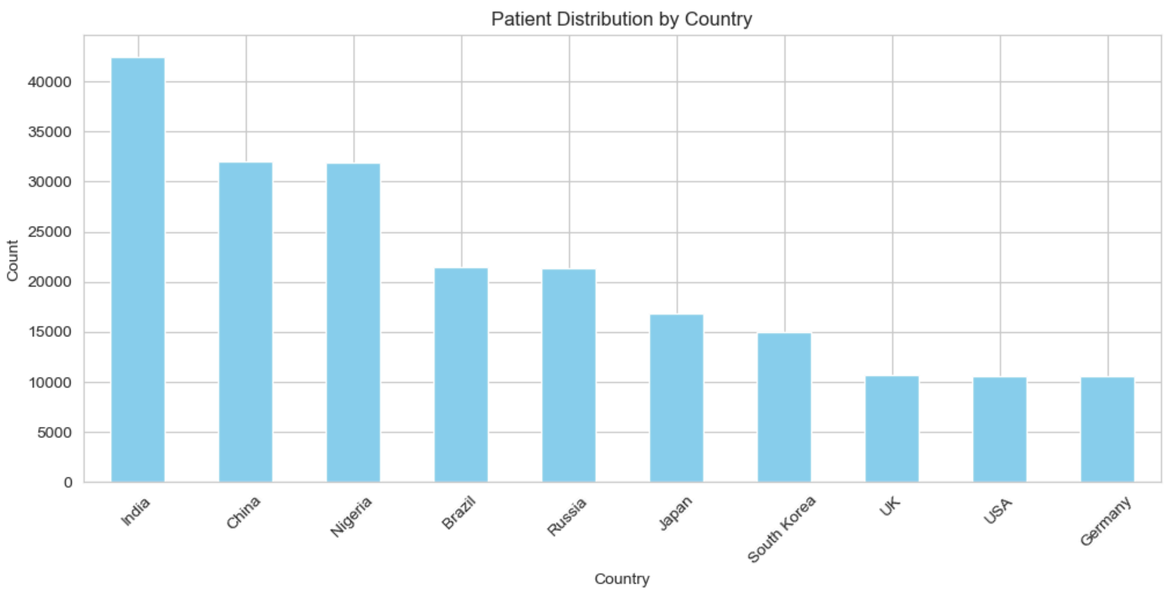
In [39]:
```
# 2. Boxplot of TSH, T3, and T4 Levels
plt.figure(figsize=(12, 5))
sns.boxplot(data=df[["TSH_Level", "T3_Level", "T4_Level"]], palette=["darkblue",
plt.title("Boxplot of Thyroid Hormone Levels")
plt.ylabel("Hormone Levels")
plt.show()
```



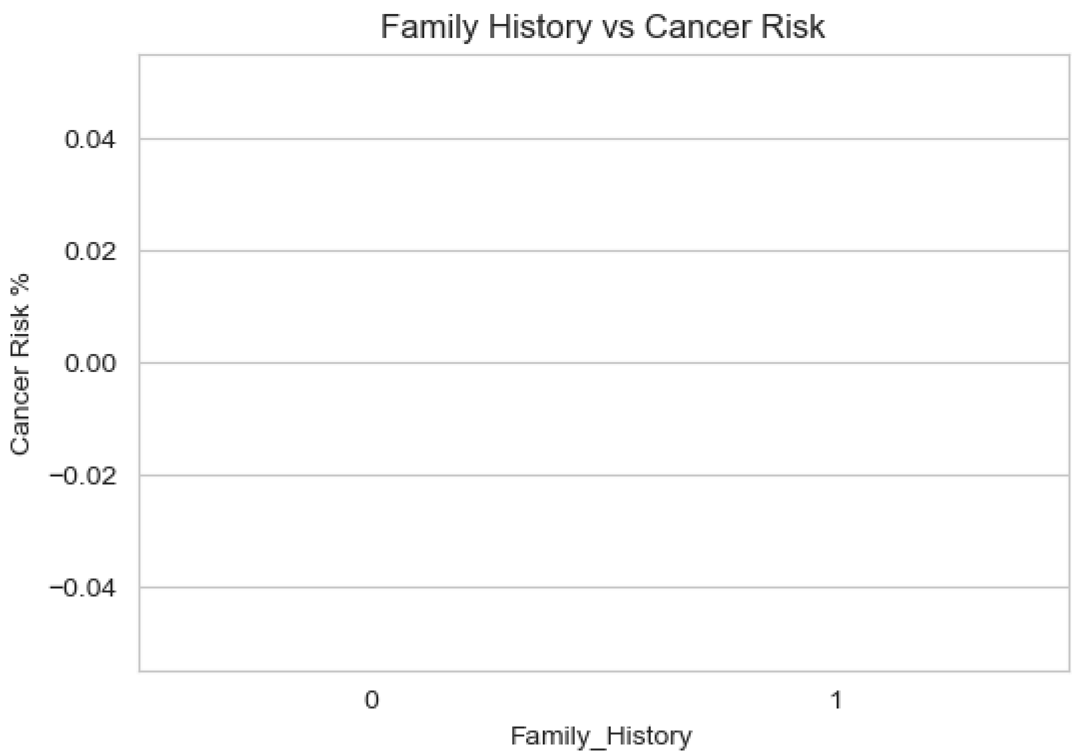
In [41]:
```
# 3. Gender vs Diagnosis
plt.figure(figsize=(6, 4))
sns.countplot(x="Gender", hue="Diagnosis", data=df, palette="coolwarm")
plt.title("Thyroid Cancer Diagnosis by Gender")
plt.xlabel("Gender")
plt.ylabel("Count")
plt.show()
```



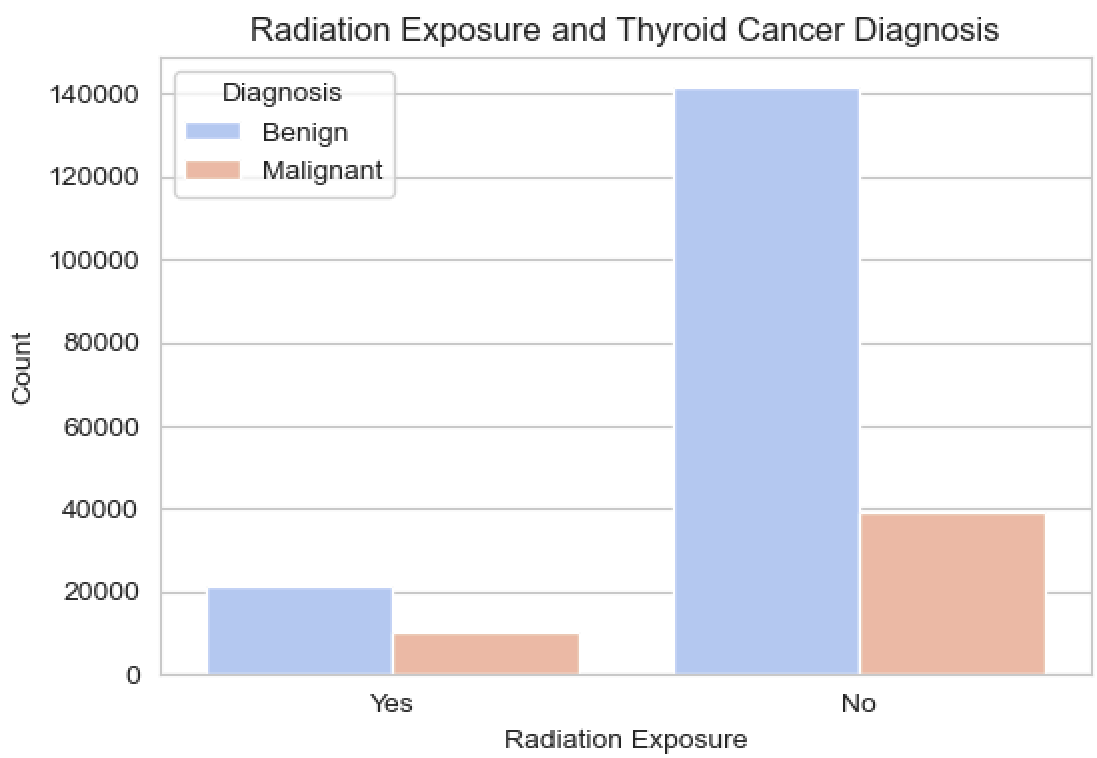
In [43]:
```
# 4. Country-wise Distribution
plt.figure(figsize=(12, 5))
df["Country"].value_counts().plot(kind="bar", color="skyblue")
plt.title("Patient Distribution by Country")
plt.xlabel("Country")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.show()
```



In [45]:
```
# 5. Family History vs Thyroid Cancer Risk
plt.figure(figsize=(6, 4))
sns.barplot(x="Family_History", y="Thyroid_Cancer_Risk", data=df, palette="Blues
plt.title("Family History vs Cancer Risk")
plt.ylabel("Cancer Risk %")
plt.show()
```

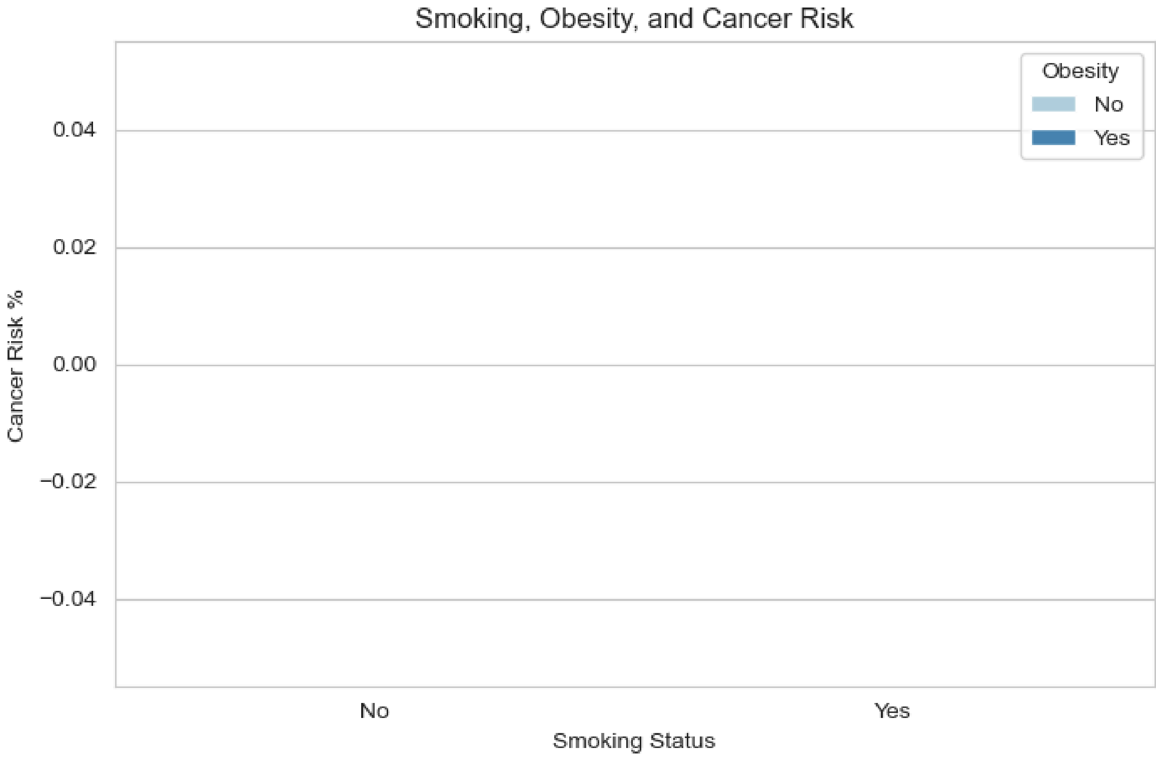


In [47]:
```
# 6. Radiation Exposure & Cancer Diagnosis
plt.figure(figsize=(6, 4))
sns.countplot(x="Radiation_Exposure", hue="Diagnosis", data=df, palette="coolwar
plt.title("Radiation Exposure and Thyroid Cancer Diagnosis")
plt.xlabel("Radiation Exposure")
plt.ylabel("Count")
plt.show()
```



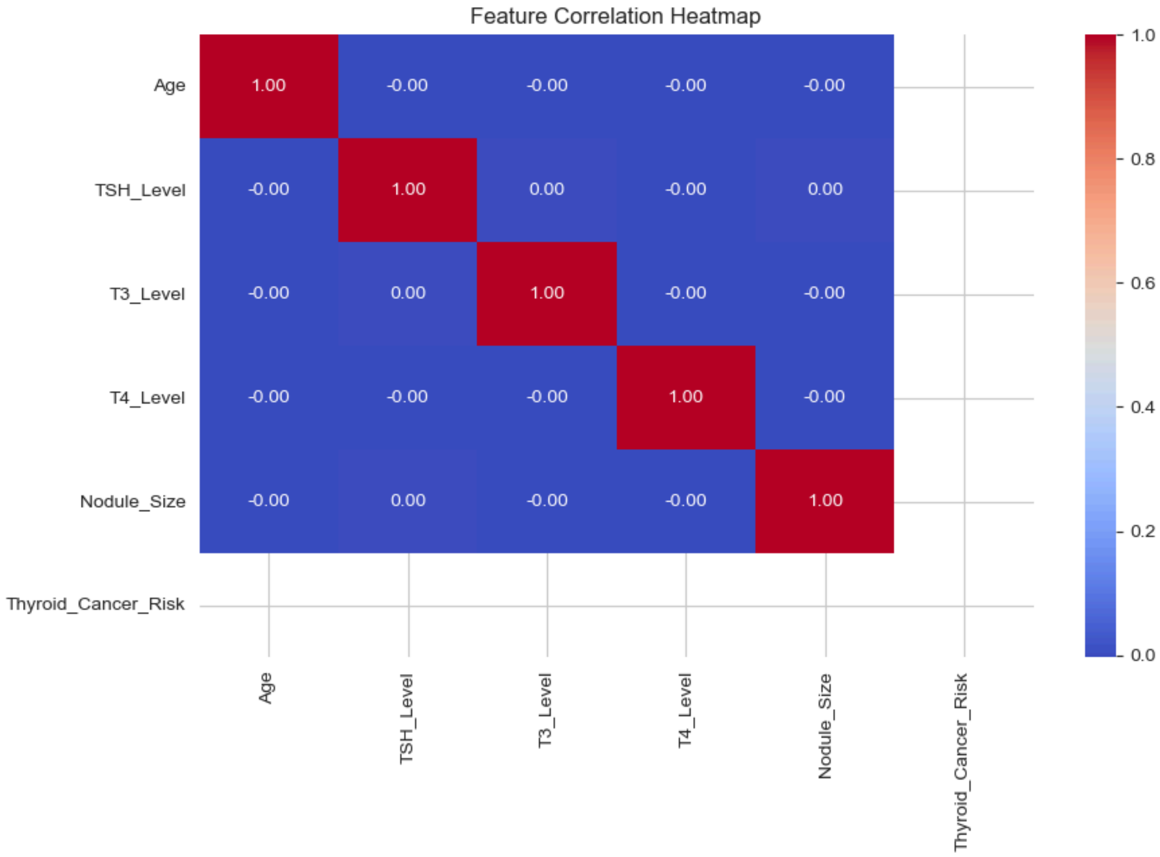
In [49]:
```
# 7. Smoking & Obesity vs Cancer Risk
plt.figure(figsize=(8, 5))
sns.barplot(x="Smoking", y="Thyroid_Cancer_Risk", hue="Obesity", data=df, palett
plt.title("Smoking, Obesity, and Cancer Risk")
```

```
plt.xlabel("Smoking Status")
plt.ylabel("Cancer Risk %")
plt.show()
```

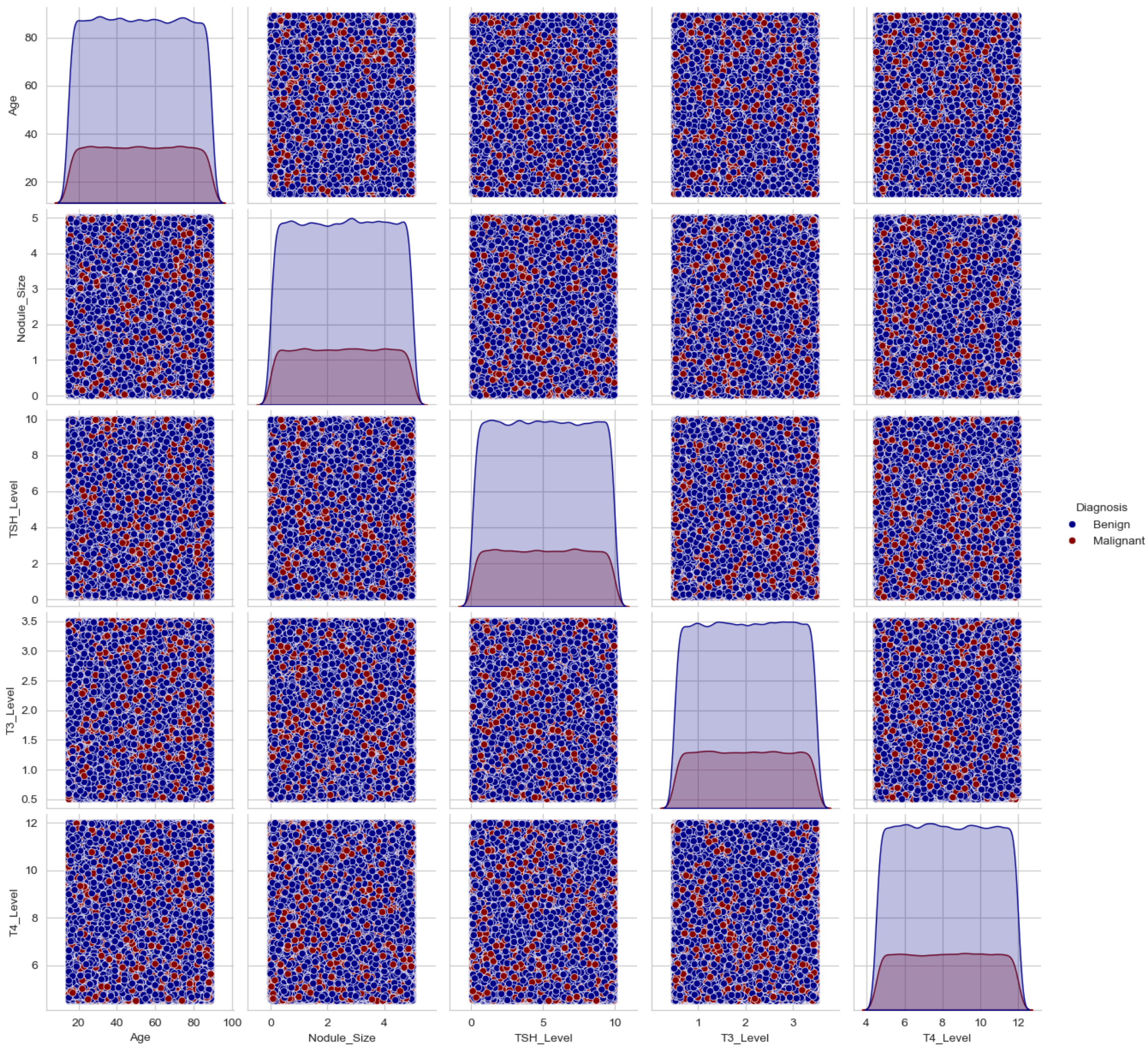


In [51]:

```
# 8. Correlation Heatmap
plt.figure(figsize=(10, 6))
corr = df[["Age", "TSH_Level", "T3_Level", "T4_Level", "Nodule_Size", "Thyroid_C
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()
```



In [53]:
```
# 9. Pairplot for Key Features
sns.pairplot(df, vars=["Age", "Nodule_Size", "TSH_Level", "T3_Level", "T4_Level"
plt.show()
```



ML Algorithms

In [56]:
```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import accuracy_score, classification_report
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier
```

In [60]:
```
from sklearn.preprocessing import LabelEncoder
df_categorical = df.select_dtypes(include='object')
label_encoders = {}
for column in df_categorical.columns:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le
print(df.head())
```

```
   Patient_ID  Age  Gender  Country  Ethnicity  Family_History  \
0           1   66       1        6          2               0   
1           2   29       1        2          3               0   
2           3   86       1        5          2               0   
3           4   75       0        3          1               0   
4           5   35       0        2          0               1   

   Radiation_Exposure  Iodine_Deficiency  Smoking  Obesity  Diabetes  \
0                   1                  0        0        0         0   
1                   1                  0        0        0         0   
2                   0                  0        0        0         0   
3                   0                  0        0        0         0   
4                   1                  0        0        0         0   

   TSH_Level  T3_Level  T4_Level  Nodule_Size  Thyroid_Cancer_Risk  Diagnosis  
0       9.37      1.67      6.16         1.08                  NaN          0  
1       1.83      1.73     10.54         4.05                  NaN          0  
2       6.26      2.59     10.57         4.61                  NaN          0  
3       4.10      2.62     11.04         2.46                  NaN          0  
4       9.10      2.11     10.71         2.11                  NaN          0  
```

In [62]:
```
from sklearn.preprocessing import StandardScaler

x = df.drop(columns=["Patient_ID", "Diagnosis"])
y = df["Diagnosis"]
x_scaled = StandardScaler().fit_transform(x)
```

In [64]:
```
from sklearn.model_selection import train_test_split
```

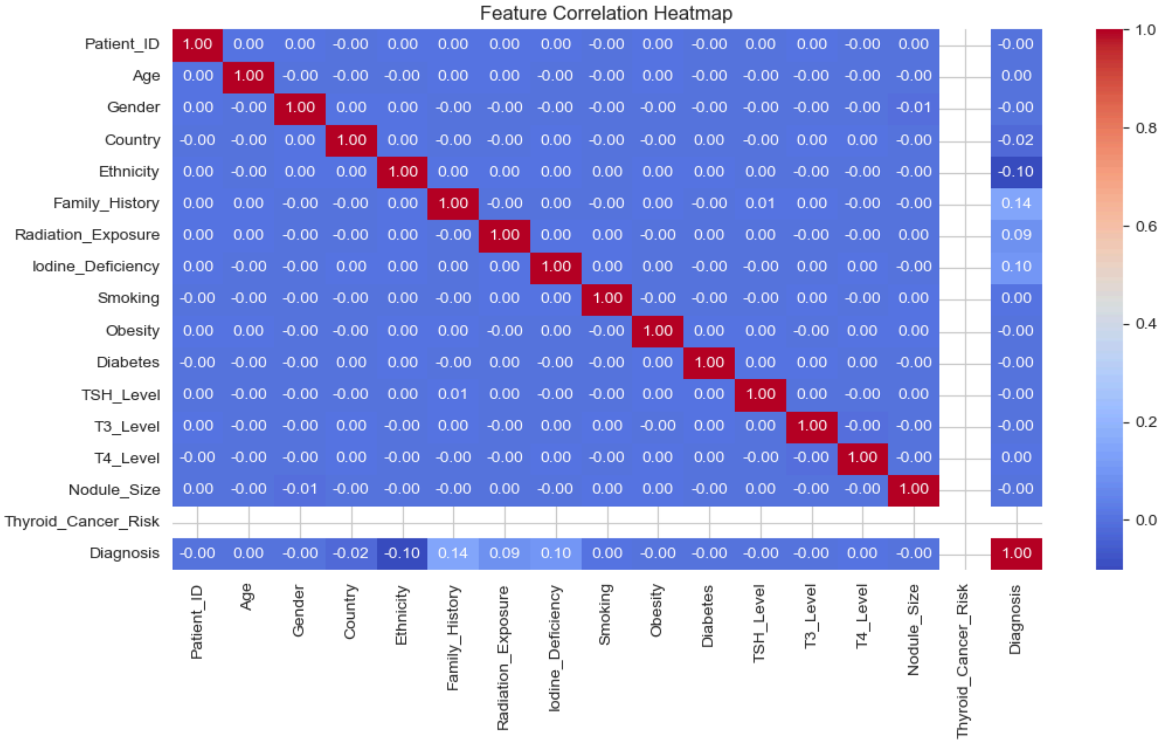
In [66]:
```
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y, test_size=0.2,
```

In [68]:
```
for col in df.columns:
    if df[col].dtype == "object":
        df[col].fillna(df[col].mode()[0], inplace=True)
    else:
        df[col].fillna(df[col].median(), inplace=True)
```

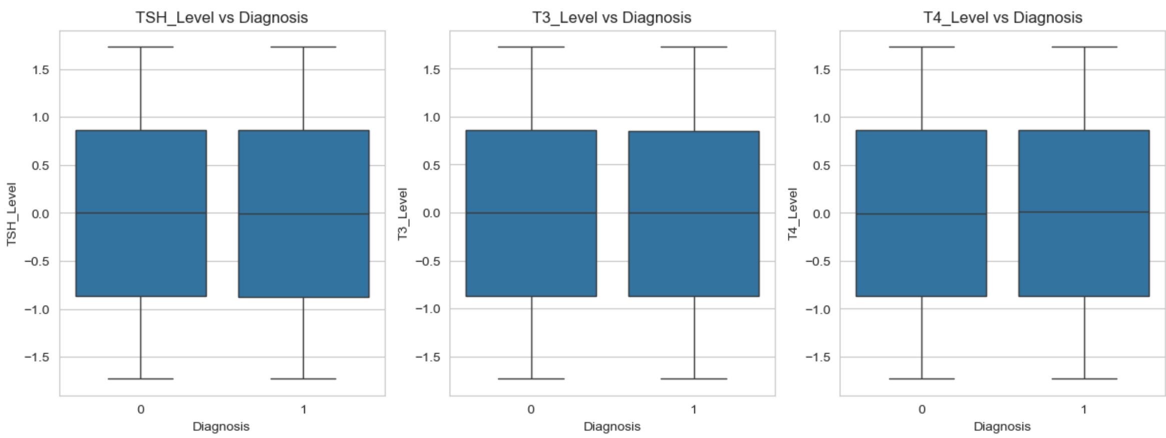
In [70]:
```
categorical_cols = ['Gender', 'Country', 'Ethnicity', 'Family_History',
                    'Radiation_Exposure', 'Iodine_Deficiency', 'Smoking',
                    'Obesity', 'Diabetes']
encoder = LabelEncoder()
for col in categorical_cols:
    df[col] = encoder.fit_transform(df[col])
```

In [72]:
```
scaler = StandardScaler()
num_cols = ['Age', 'TSH_Level', 'T3_Level', 'T4_Level', 'Nodule_Size']
df[num_cols] = scaler.fit_transform(df[num_cols])
```

In [74]:
```
plt.figure(figsize=(12, 6))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()
```



In [76]:
```
plt.figure(figsize=(15, 5))
for i, col in enumerate(['TSH_Level', 'T3_Level', 'T4_Level'], 1):
    plt.subplot(1, 3, i)
    sns.boxplot(x=y, y=df[col])
    plt.title(f"{col} vs Diagnosis")
plt.show()
```



In [78]:
```
print(df.isnull().sum())
```

```
Patient_ID                0
Age                       0
Gender                    0
Country                   0
Ethnicity                 0
Family_History            0
Radiation_Exposure        0
Iodine_Deficiency         0
Smoking                   0
Obesity                   0
Diabetes                  0
TSH_Level                 0
T3_Level                  0
T4_Level                  0
Nodule_Size               0
Thyroid_Cancer_Risk  212691
Diagnosis                 0
dtype: int64
```

In [80]:
```
df['Thyroid_Cancer_Risk'].fillna(df['Thyroid_Cancer_Risk'].median(), inplace=Tru
```

In [82]:
```
print(df.isnull().sum())
```

```
Patient_ID                0
Age                       0
Gender                    0
Country                   0
Ethnicity                 0
Family_History            0
Radiation_Exposure        0
Iodine_Deficiency         0
Smoking                   0
Obesity                   0
Diabetes                  0
TSH_Level                 0
T3_Level                  0
T4_Level                  0
Nodule_Size               0
Thyroid_Cancer_Risk  212691
Diagnosis                 0
dtype: int64
```

In [84]:
```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy='mean')  # or 'median', 'most_frequent'
x_train = imputer.fit_transform(x_train)
x_test = imputer.transform(x_test)
```

In [86]:
```
print("Original X columns:", df.drop(columns=["Patient_ID", "Diagnosis"]).shape[
print("Scaled X columns:", x_scaled.shape[1])
```

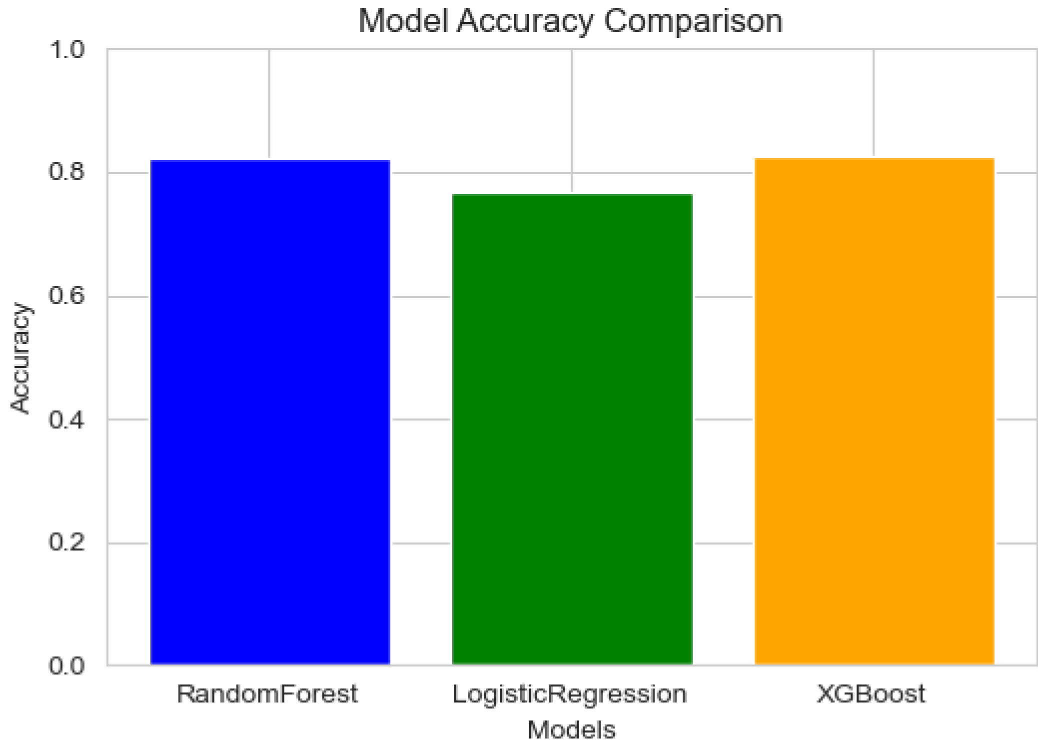
```
Original X columns: 15
Scaled X columns: 15
```

In [88]:
```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy="mean")  # or "median", "most_frequent"
X_imputed = imputer.fit_transform(df.drop(columns=["Patient_ID", "Diagnosis"]))

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_imputed)
```

In [90]:
```
feature_names = df.drop(columns=["Patient_ID", "Diagnosis"]).columns[:x_train.sh
x_train = pd.DataFrame(x_train, columns=feature_names)
x_test = pd.DataFrame(x_test, columns=feature_names)
```

In [92]:
```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt

models_params = {
    'RandomForest': (RandomForestClassifier(random_state=42), {
        'n_estimators': [50, 100],
        'max_depth': [None, 10],
    }),
    'LogisticRegression': (LogisticRegression(random_state=42, max_iter=500), {
        'C': [0.01, 1, 100],
    }),
    'XGBoost': (XGBClassifier(random_state=42, use_label_encoder=False, eval_met
        'n_estimators': [50, 100],
        'learning_rate': [0.01, 0.1],
    })
}
results = {}
for name, (model, param_dist) in models_params.items():
    print(f"Training {name}...")

    search = RandomizedSearchCV(
        estimator=model,
        param_distributions=param_dist,
        n_iter=3,
        scoring='accuracy',
        cv=3,
        n_jobs=-1,
        random_state=42,
        verbose=0
    )
    search.fit(x_train, y_train)

    best_model = search.best_estimator_
    y_pred = best_model.predict(x_test)

    results[name] = accuracy_score(y_test, y_pred)
    print(f"{name} Accuracy: {results[name]:.4f}")

# Visualization
plt.figure(figsize=(6, 4))
plt.bar(results.keys(), results.values(), color=['blue', 'green', 'orange'])
plt.xlabel("Models")
plt.ylabel("Accuracy")
plt.title("Model Accuracy Comparison")
plt.ylim(0, 1)
plt.show()
```

```
Training RandomForest...
RandomForest Accuracy: 0.8228
Training LogisticRegression...
LogisticRegression Accuracy: 0.7680
Training XGBoost...
XGBoost Accuracy: 0.8249
```



Completed