```
# Step 1: Import Libraries and Load Data
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.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Load your dataset (replace 'cancer_data.csv' with your dataset file)
data = pd.read_csv('cancer_data.csv')

# Step 2: Problem Description
print("Cancer Detection Mini-Project")
print("------------------------------")
print("Problem Description:")
print("We aim to detect cancer based on patient data.")

# Step 3: Data Description
print("\nData Description:")
print("Number of Samples:", data.shape[0])
print("Number of Features:", data.shape[1])
print("\nSample Data:")
print(data.head())

# Step 4: Exploratory Data Analysis (EDA)
# Visualize data to understand its distribution and relationships
plt.figure(figsize=(10, 6))
sns.countplot(data['target'])
plt.title('Distribution of Target Variable')
plt.xlabel('Target')
plt.ylabel('Count')
plt.show()

# Step 5: Data Preprocessing
# Preprocess the data (feature scaling, encoding, train-test split, etc.)
X = data.drop('target', axis=1)
y = data['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Step 6: Model Building
# Build a logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Step 7: Model Evaluation
# Evaluate the model on the testing dataset
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
```

```
print("\nModel Evaluation:")
print("Accuracy:", accuracy)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
print("\nConfusion Matrix:")
conf_matrix = confusion_matrix(y_test, y_pred)
print(conf_matrix)

# Step 8: Model Optimization
# Fine-tune hyperparameters to improve model performance (if needed)

# Step 9: Results
# Present the results and insights here

# Step 10: Conclusion and Discussion
# Summarize the project and discuss findings and limitations

# Step 11: Save and Share
# Save the Jupyter notebook with all code and explanations

# Step 9: Results
# Present the results and insights here

# Example Results (using synthetic data)
print("\nExample Results:")
print("----------------")
print("Accuracy:", accuracy)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
print("\nConfusion Matrix:")
conf_matrix = confusion_matrix(y_test, y_pred)
print(conf_matrix)

# Interpretation of Results (using synthetic data)
print("\nInterpretation of Results:")
print("--------------------------")
print("The model achieved an accuracy of {:.2f}% on the test data.".format(accuracy * 100))
print("Precision for class 0 (No Cancer): {:.2f}".format(conf_matrix[0, 0] / (conf_matrix[0, 0] + conf_matrix[1
, 0])))
print("Recall for class 0 (No Cancer): {:.2f}".format(conf_matrix[0, 0] / (conf_matrix[0, 0] + conf_matrix[0, 1
])))
print("Precision for class 1 (Cancer): {:.2f}".format(conf_matrix[1, 1] / (conf_matrix[1, 1] + conf_matrix[0, 1]
)))
print("Recall for class 1 (Cancer): {:.2f}".format(conf_matrix[1, 1] / (conf_matrix[1, 1] + conf_matrix[1, 0])))

# Step 10: Conclusion and Discussion
# Summarize the project and discuss findings and limitations

# Step 11: Save and Share
# Save the Jupyter notebook with all code and explanations
```