`load_data_from_s3`

1. Define the function `load_data_from_s3`.
2. Download the heart disease dataset from an S3 bucket to a local file.
3. Read the CSV file into a Spark DataFrame.
4. Return the DataFrame.

`clean_data`

1. Define the function `clean_data` with input `df` (the DataFrame to clean).
2. Create a list of columns to retain from the original DataFrame.
3. Select only those columns from the DataFrame.
4. Calculate the mode of the 'painloc' column and fill missing values in the 'painloc' column with the mode.
5. Calculate the mode of the 'painexer' column and fill missing values in the 'painexer' column with the mode.
6. Impute values in the 'trestbps' column (values less than 100 are replaced with 100).
7. Impute 'oldpeak' values: replace values less than 0 with 0 and those greater than 4 with 4.
8. Impute missing values in 'thaldur' and 'thalach' using their respective means.
9. For columns like 'fbs', 'prop', 'nitr', 'pro', and 'diuretic', calculate the mode and fill missing values. Also, clip values greater than 1 to 1.
10. Calculate the mode for the 'exang' and 'slope' columns and fill missing values with the mode.
11. Impute missing values for the 'age', 'sex', 'cp', 'trestbps', and 'target' columns with the mode.
12. Impute missing 'oldpeak' values with the mean.
13. Limit the dataset to the first 899 rows, as rows after this are not in the correct format.
14. Return the cleaned DataFrame.

`impute_smoking_1`

1. Define the function `impute_smoking_1` with input `url` and `df_cleaned`.
2. Send an HTTP request to the provided URL.
3. If the request is successful (status code 200):
    - Parse the HTML content of the page using BeautifulSoup.
    - Find the relevant table containing smoking data.
    - Extract smoking rate data from the table for each age group.
    - Store the smoking rate data in a dictionary (`smoking_rate_by_age`).
4. Add a new column `smoke_source_1` to the DataFrame, initializing it with values from the 'smoke' column.

5. Modify the 'smoke_source_1' column based on the smoking rate data (e.g., multiply or replace values).
6. Return the updated DataFrame.

`impute_smoking_2(url, df_cleaned)`

1. Define the function `impute_smoking_2(url, df_cleaned)`.
2. Send an HTTP GET request to `url`.
3. Parse the HTML page using BeautifulSoup.
4. Extract the relevant table containing smoking rate data.
5. Identify smoking rate values by age group and store them in a dictionary, converting percentages to decimal values.
6. Add a new column `smoke_source_2` to the DataFrame `df_cleaned`.
7. Match each row with the appropriate smoking rate from the dictionary using the `age` column.
8. Fill in missing smoking data based on the corresponding age group's rate.
9. Return the updated DataFrame.

`train_heart_disease_model(df_cleaned)`

1. Define the function `train_heart_disease_model(df_cleaned)`.
2. Define the feature columns for model training.
3. Assemble the features into a feature vector using `VectorAssembler`.
4. Split the data into 90-10 stratified training and test sets.
5. Initialize the classification models to be used (e.g., Random Forest, Logistic Regression).
6. Set up hyperparameter tuning using `ParamGridBuilder`.
7. Perform 5-fold cross-validation using `CrossValidator`.
8. Evaluate the model using `BinaryClassificationEvaluator`.
9. Select the best model based on the evaluation metrics.
10. Save the trained model.
11. Return the trained model.

`main()`

1. Define the function `main()`.
2. Call `load_data_from_s3()` to load the dataset into a Spark DataFrame.
3. Call `clean_data(df)` to clean the data.
4. Call `impute_smoking_1(url, df_cleaned)` to fill the `smoke_source_1` column.

5. Call `impute_smoking_2(url, df_cleaned)` to fill the `smoke_source_2` column.
6. Call `train_heart_disease_model(df_cleaned)` to train the model.
7. Optionally, save the trained model.
8. Return the final trained model.