**Use case Diagram:**

**Fig: Use Case Diagram**

A use case diagram is a visual representation used in software engineering to depict the interactions between system actors and the system itself. It captures the dynamic behavior of a system by illustrating its use cases and the roles that interact with them. These diagrams are essential in specifying the system’s functional requirements and understanding how users will interact with the system.

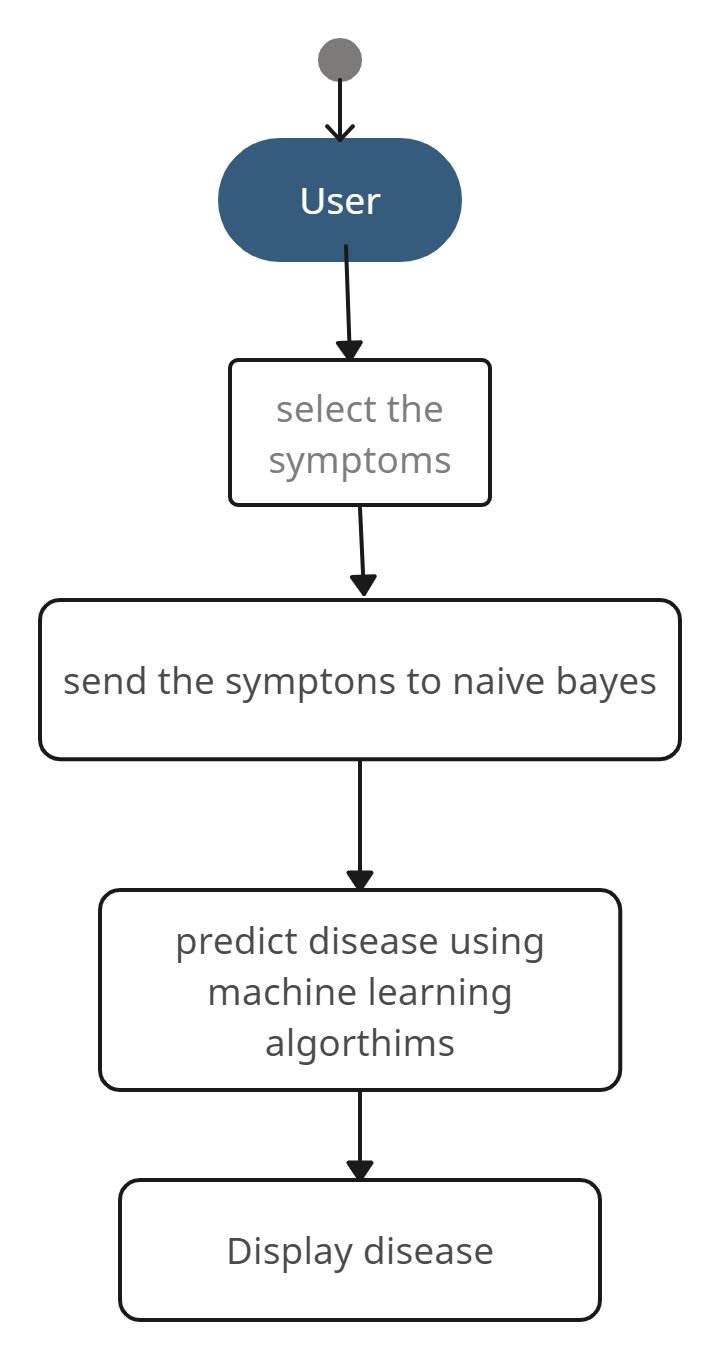
**Actors:**

* User (U): Represents the user interacting with the system to upload symptoms, preprocess them, and detect diseases using machine learning.

**Use Cases**:

* Prepare dataset: This use case represents the action where the process of preparing raw data such that it is suitable for further processing and analysis which include collecting, cleaning, and labeling raw data into a form suitable
* Import Modules: The system preprocesses the uploaded dataset to prepare it for feature extraction and model training.
* Predict disease: In this use case, the system extracts relevant features from the preprocessed dataset, which are essential for disease prediction.
* Train Model: The system uses the extracted features to train a machine learning model for prediction of disease.
* **Interactions**:
* The admin and user interact with various system components:
* Preparing dataset: The admin prepares the dataset for further processing and evaluation required.
* Import: The admin imports the required modules for predicting diseases.
* Predict: The admin after up0loading the symptoms predicts the disease using algorithms.
* Train: The user adjusts parameters for model training.
* Detect: The user views the results of disease prediction
* **Relationships:**
* Association: Each use case is associated with the admin, indicating that the user interacts with these use cases to perform specific actions related to disease prediction and user is associated to uploading symptoms and viewing results.
* **Flow of Operations:**
* The user starts by uploading a symptom.
* The system preprocesses the uploaded symptoms.
* The predicted disease is displayed.

**Activity Diagram:**



**Fig: Activity Diagram**

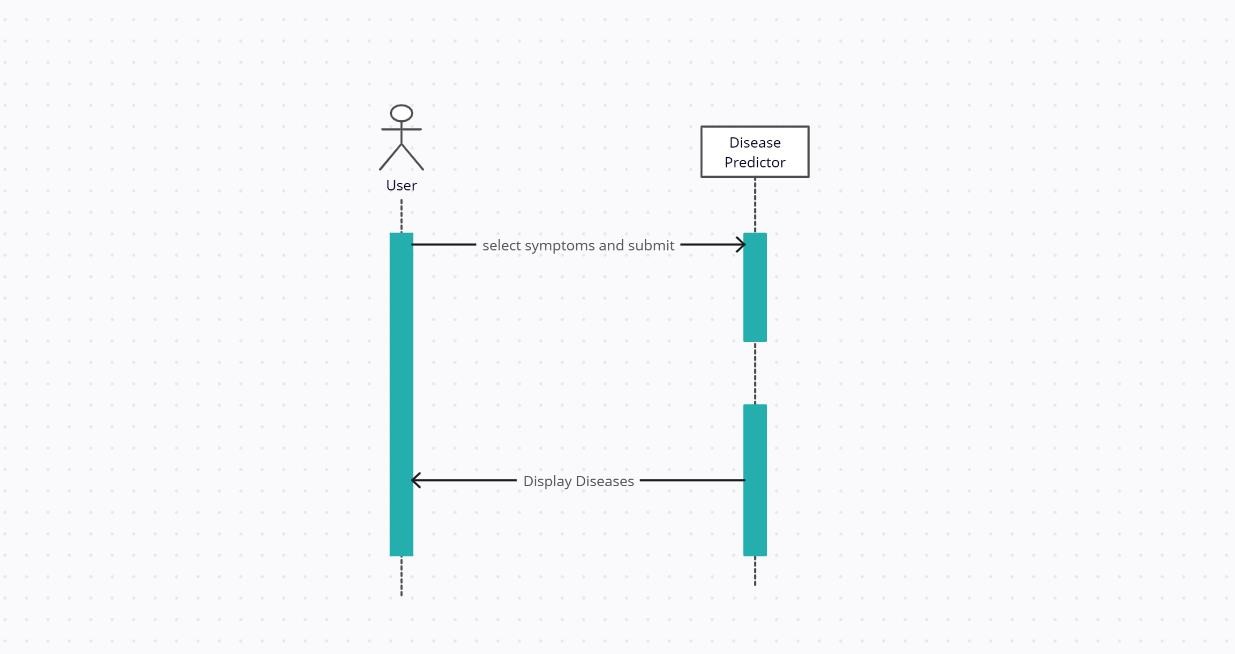
User Input: The user interacts with the system by providing input, which includes entering their symptoms into the interface.

Machine Learning Model: The pre-processed data is fed into the machine learning model. The model performs classification based on the trained algorithms to predict potential diseases associated with the given symptoms.

Prediction Output: The system generates a prediction output, indicating the likely disease(s) based on the input symptoms.

Result Presentation: The predicted results are presented to the user through the system's interface. This could include a list of potential diseases ranked by likelihood.

**Sequence Diagram**



**Fig: Sequence Diagram**

In this sequence diagram:

1.The user provides symptoms data to the Disease Prediction System.

2.The Disease Prediction System interacts with a Data Source to retrieve historical symptom data for training.

3.The Disease Prediction System employs a Machine Learning Algorithm to train a prediction model using the historical data.

4.The Machine Learning Algorithm returns the trained model to the Disease Prediction System. When new symptoms data is provided, the Disease Prediction System requests a prediction from the Machine Learning Algorithm.

5.The Machine Learning Algorithm predicts the likelihood of disease occurrence based on the provided symptoms data.

Finally, the Disease Prediction System presents the disease prediction results to the user.

This sequence diagram illustrates the flow of interactions between the user, the disease prediction system, the data source, and the machine learning algorithm in the context of predicting diseases based on symptoms.

**Class Diagram:**

A diagram of a computer

Description automatically generated

**Fig: Class Diagram**

Class diagrams are one of the most useful types of diagrams in UML as they clearly map out the structure of a particular system by modelling its classes, attributes, operations, and relationships between objects. It is a static diagram that represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The class shape itself consists of a rectangle with three rows. The top row contains the name of the class, the middle row contains the attributes of the class, and the bottom section expresses the methods or operations that the class may use. Classes and subclasses are grouped together to show the static relationship between each object.

**Class Name:**

The name of the class appears in the first partition.

**Class Attributes:**

Attributes are shown in the second partition. The attribute type is shown after the colon. Attributes map onto member variables (data members) in code.

It explains the classes used in the disease prediction system.

1. Symptoms Reader: Reads the user input and creates the list of symptoms

2.Analyzer: Based on symptoms parameter, it displays the result.

3.Calculate Values: Calculates the probabilistic model of the diseases.