

Programming for Artificial Intelligence

Semester Project

**Submitted By:** Group 9

**Members:**

Dur Muhammad Khan FA23-BAI-012

Hassan Afzal FA23-BAI-019

**Submission Date:** 13-06-2025

**Submitted to:** Dr Usman

**DEPARTMENT OF COMPUTER SCIENCE**

**Project Title**

**Disease Predictor using Naive Bayes and Flutter**

**Objective**

To build a machine learning-powered Android application that predicts the most likely disease based on the user’s symptoms using a trained Naive Bayes model. The app utilizes a Flask backend and is deployed on PythonAnywhere.

**Tools & Technologies**

* **Backend**:
  + Python
  + Pandas
  + Scikit-learn (BernoulliNB & MultinomialNB)
  + Flask
  + Joblib
  + PythonAnywhere (Deployment)
* **Frontend**:
  + Flutter (for Android)
  + Dart
  + HTTP (Dio package for API calls)
* **Others**:
  + VS Code (Development)
  + Kaggle

**Workflow Summary**

1. **Data Preparation**:
   * We picked the dataset from Kaggle.
2. **Model Training**:
   * Two models were trained using:
     + **Bernoulli Naive Bayes** (best for binary features).
     + **Multinomial Naive Bayes** (best for frequency-based features).
   * After evaluation, both models were stored using joblib.
3. **Model Deployment**:
   * A **Flask** web application was created to serve the models.
   * The API accepts a list of symptoms in JSON format and returns:
     + Predicted disease
     + Confidence score (probability)
     + Results from both models
   * The result with the **highest confidence score** is shown to the user.
   * Flask app is hosted on **PythonAnywhere** for public access.
4. **Flutter App (Frontend)**:
   * Users select their symptoms from a list.
   * The app sends the symptom list to the Flask backend using a **POST** request.
   * The returned prediction is parsed and shown to the user with the disease name.

**API Design**

* **Endpoint**: https://yourusername.pythonanywhere.com/predict
* **Method**: POST

**Input (JSON)**:

{

"headache": 1,

"fever": 1,

"cough": 0,

...

}

**Output (JSON)**:

{

"predicted\_disease": "Flu",

"confidence": 0.87,

"model\_used": "BernoulliNB",

"all\_model\_scores": {

"BernoulliNB": 0.87,

"MultinomialNB": 0.75

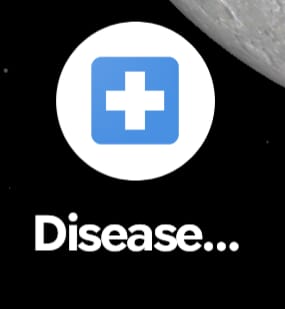
}

}

**Results**

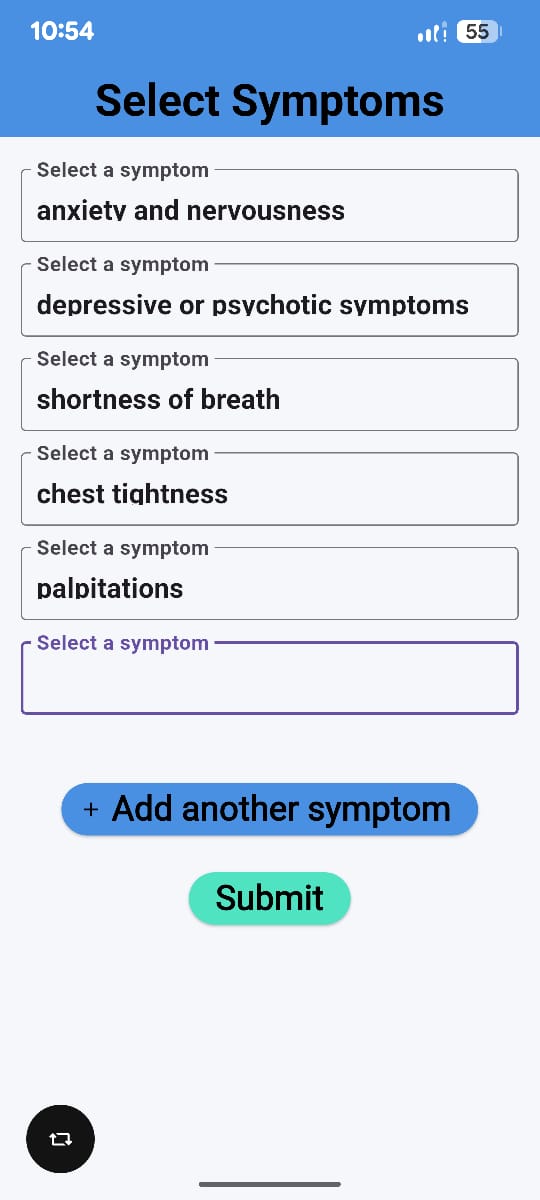
* The app successfully predicts diseases with reasonable accuracy using both BernoulliNB and MultinomialNB.
* Prediction confidence is dynamically compared between models.
* Only the most confident prediction is shown in the user interface for simplicity.

**GUI and Output example :**

**App icon:**

**Splash Screen:**

****

**Symptom Selection page:**

**Output Screen:**

**9. Future Improvements**

* **Include more diseases and symptoms for broader coverage.**
* **Use ensemble models or deep learning for better accuracy.**
* **Add user account system to save prediction history.**
* **Improve UI and add multi-language support.**