AI Healthcare Assistant – Project Report

AI Solution (Relevance to Theme)

Our solution is an AI Healthcare Assistant that predicts diseases from patient symptoms. It is relevant to the theme “AI Solution for Industries” because the healthcare sector is facing critical doctor shortages. This solution uses AI classification models to provide early detection support, improve patient self-awareness, and assist healthcare workers in decision-making.

Problem Definition

Healthcare systems in many communities, particularly in developing regions, face the persistent problem of not having enough doctors to serve the growing population. Patients often wait long periods before being diagnosed, which delays treatment and worsens health outcomes. The problem is the shortage of qualified medical practitioners, resulting in long queues, late interventions, and avoidable complications. An AI-powered solution that predicts possible diseases from symptoms can reduce this gap. Patients can input their symptoms into a system, receive predictions, definitions of potential diseases, and recommended precautions. This does not replace doctors but helps in early awareness and better decision-making. For local municipalities, this solution reduces pressure on clinics and hospitals, improves community health education, and provides quick, cost-effective support where medical staff are limited.

Business Objectives

- Build an AI system that predicts diseases based on symptoms.
- Provide patients with definitions and precautionary advice.
- Support healthcare workers with faster pre-diagnosis.
- Reduce waiting times and improve access to medical information.

Requirements, Constraints, Risks

- Requirements: Symptom–Disease datasets, Python libraries (Pandas, scikit-learn, Streamlit), GitHub for project management.
- Constraints: Limited dataset size, limited time, need for anonymised health data.
- Risks: Low accuracy (mitigation: try multiple models), usability issues in GUI (mitigation: iterative testing).

Tools and Techniques

- Tools: Python, Jupyter, GitHub, Streamlit.
- Techniques: Data cleaning, One-Hot Encoding, Label Encoding, Classification (Decision Trees, Random Forest), Model evaluation (accuracy, precision, recall, F1), GUI development.

Machine Learning Approach

We use classification algorithms because symptoms must be mapped to discrete disease categories.
- Baseline models: Logistic Regression, Decision Tree, Naïve Bayes.
- Advanced models: Random Forest, Gradient Boosting (XGBoost).
This ensures a balance between interpretability and performance.

Data

The data collected is directly relevant to the problem and forms the foundation of the AI system.

1. Symptoms–Disease Dataset → training the model. Example: Flu – fever, cough, headache.
2. Disease Definitions Dataset → for explanations in GUI. Example: Flu – viral infection affecting respiratory system.
3. Precautions Dataset → preventive advice. Example: Flu – rest, fluids, consult doctor if severe.
4. Weight vs Disease Dataset → extra analysis. Example: Obesity – linked with diabetes, heart disease.

Together, these datasets provide both the training data for machine learning and the contextual information needed for real-world application.

Model

The model will be evaluated using:
- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix
This ensures reliable predictions beyond raw accuracy.

Solution Techniques

- Data cleaning, encoding, and classification.
- Hyperparameter tuning for Random Forest and Gradient Boosting to improve accuracy.
- Using cross-validation to reduce overfitting.

Practical Solution

The practical outcome is a Streamlit app where users select symptoms, and the AI system predicts the most likely disease, provides definitions, and suggests precautions.