Synopsis – Health Predictor 💊

Team – HealthyLife-AI

Health Predictor: Disease Risk Prediction & Health Insights

Objectives

● **Predict Disease Risks**

Predict the likelihood of diseases such as **diabetes** and **heart disease** using input medical features from users.

● **Explainable Predictions**

Provide users with **transparent predictions** using **SHAP values** or **feature importance**, highlighting which inputs influenced results the most.

● **Cold-Start Handling**

Handle missing or partial user data with **mean/mode imputation** and provide **fallback logic** based on known population trends.

● **User Segmentation**

Automatically classify users into **Low**, **Medium**, or **High Risk** categories based on model predictions.

● **Analytics Dashboard**

Deliver **real-time insights**, **risk distribution charts**, and **feature impact visuals** through an interactive **Streamlit dashboard**.

Project Description

Health Predictor is a smart disease risk detection platform that uses **machine learning models** to evaluate the risk of conditions like diabetes and heart disease from medical inputs such as age, BMI, glucose, and blood pressure.

The system consists of robust **data preprocessing**, model training pipelines, and an intuitive **user-facing dashboard**. It offers predictions, **risk segmentation**, and **explainable outputs**, enabling users and health professionals to make data-driven wellness decisions.

Deliverables

🔹 Data Ingestion & ETL

- Use public datasets like:
  – *Pima Indians Diabetes Dataset*
  – *UCI Heart Disease Dataset*

- Preprocessing tasks:
  – Handle **missing values** using imputation
  – Normalize numerical columns (z-score or MinMax scaling)
  – Label encode categorical fields (e.g., gender, chest pain type)

🔹 Exploratory Data Analysis (EDA)

- Visualize class balance (Healthy vs Diseased)
- Correlation heatmaps among features
- Relationship plots (e.g., Glucose Level vs Diabetes Probability)
- Identify key trends like age/BMI correlations with disease risk

---

🔹 Machine Learning Models

- Train and compare the following models:
  – Logistic Regression
  – Random Forest
  – XGBoost (optional)

- Model evaluation via:
  – Accuracy
  – Precision
  – Recall
  – F1-Score
  – ROC-AUC Curve

- Apply **k-fold Cross Validation** for performance robustness

---

🔹 Cold-Start Strategies

- Impute missing data using **mean (numerical)** or **mode (categorical)**
- Provide default predictions based on **average risk profiles**
- Suggest most **influential features** to collect next

🔹 Evaluation Metrics

- Use classification metrics for effectiveness:
  – Confusion Matrix
  – ROC Curve
  – Precision-Recall Tradeoff

● Display detailed metrics on the dashboard

🔹 **Interactive Dashboard**

Built using Streamlit:

● Real-time **input form** for user health parameters
● Instant prediction display:
→ e.g., "High Risk of Diabetes"
● Visualization components:
– **Pie chart** of risk segment distribution
– **Top 5 feature influences** using SHAP
– User summary cards and historical logs

🔹 **Model Serving & API**

Backend via Flask / FastAPI:

● Input:
– JSON format user data
● Output:
– Prediction score
– Risk level (Low / Medium / High)
– Feature influence (SHAP values)

Data storage: SQLite / CSV

Deployment Ready: Docker-compatible API and UI

Advanced Features

● **SHAP-Based Explainability:** Use SHAP plots to show why a particular prediction was made (e.g., "High Glucose + High BMI → High Risk").

● **PDF Report Generator:** Allow users to download a **personalized health report** summarizing input, prediction, risk level, and advice.

● **User History Tracker:** Track and store past predictions with **timestamp**, view **prediction timeline**.

● **Multiple Disease Prediction (Multi-label) :**Extend to predict risk for **more than one disease** using multi-output classification models.

Tech Stack

Frontend

- **Streamlit** (Interactive UI)
- **Matplotlib / Seaborn / SHAP** for visualizations

Backend

- **Python 3.8+**
- **Flask or FastAPI**
- **SQLite / CSV** (for storage)
- **Docker** (for containerized deployment)

Data & ML

- **pandas, NumPy, scikit-learn**
- **XGBoost (optional)**
- **SHAP** for explainability
- **Jupyter Notebook** for experimentation

Visualization & Dashboard

- **Streamlit**
- **matplotlib, seaborn**
- **SHAP visualizer**