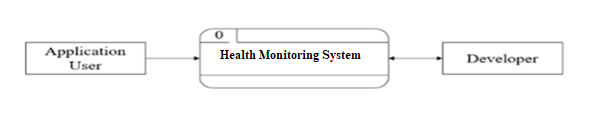
HEALTH PREDICTION

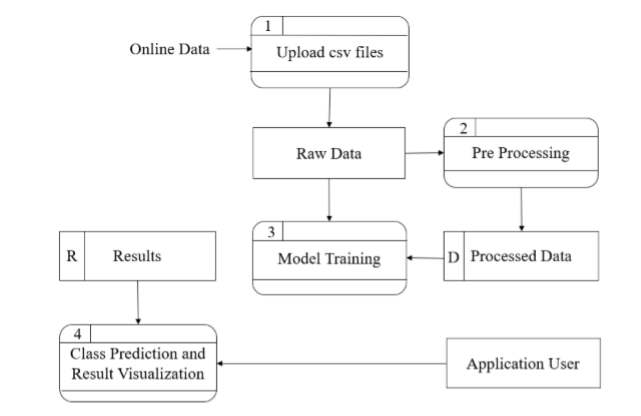
**Dataflow Diagram:**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail which can later be elaborated. DFDs can also be used for the visualization of data processing(structured design).

**Level-0**



**Level-1**

****

**Objectives :**

Objective of this project is to use the methods of machine learning in healthcare to monitor the patient health status. Goals of this project is as global general diagnosis are increasing day by day, the diagnosis from hospital centric to home centric is needed.

**Methodology:**

Once OpenCV is installed, the OPENCV\_BUILD\install directory will be populated with three types of files:

* **Header files**: These are located in the OPENCV\_BUILD\install\includesubdirectory and are used to develop new projects with OpenCV.
* **Library binaries**: These are static or dynamic libraries (depending on the option selected with CMake) with the functionality of each of the OpenCV modules. They are located in the bin subdirectory (for example, x64\mingw\bin when the GNU compiler is used).
* **Sample binaries**: These are executables with examples that use the libraries. The sources for these samples can be found in the source package.

**Input:**

• Patient data (features like):

• Age

• Gender

• Height, Weight, BMI

• Blood pressure

• Blood sugar levels

• Cholesterol levels

• Heart rate

• Medical history (e.g., past diseases, surgeries)

• Lifestyle factors (e.g., smoking, exercise habits)

• Genetic information (optional, if available)

• Symptoms (if you’re predicting diseases)

**Output:**

• Predicted health condition (target label), such as:

• Disease risk (e.g., “Diabetes: High risk” or “Heart disease: Low risk”)

• Health score (a number from 0 to 100)

• Diagnosis prediction (e.g., “Likely flu” or “Possible hypertension”)

• Recommendation (optional, like “Consult a cardiologist”)

**Example:**

| Input Features | Output |
| --- | --- |
| Age=45, BMI=30, BP=140/90, Smoker=Yes | Risk of Heart Disease = High |

**Modules Used**

• Pandas — to load and prepare data

• NumPy — to work with numbers

• Matplotlib/Seaborn — to draw charts

• Scikit-learn — to build and test ML models

• TensorFlow/Keras — for deep learning models (if needed)

• Flask/Streamlit — to make a web app for users

**Process Logic**

* Collect Data — get patient health data
* Clean Data — fix missing or wrong values
* Split Data — into training and testing sets
* Train Model — teach the model using training data
* Test Model — check accuracy using testing data
* Predict — use the model to predict health risks
* Deploy — make it live with a simple app

**Limitations**

The implementation phase of software development is concerned with translating design

specifications into source code. The primary goal of implementation is to write source

code and internal documentation so that conformance of the code to its specifications can

be easily verified and so that debugging testing and modification are eased. This goal can

be achieved by making the source code as clear and straightforward as possible. Simplicity

clarity and elegance are the hallmarks of good programs and these characteristics have

been implemented in each program module.

**The goals of implementation are as follows:-**

• Minimize the memory required.

• Maximize output readability.

• Maximize source text readability.

• Minimize the number of source statements.

* Minimize development time.

**Tools, Platforms, languages used**

* **System Components:**

• Frontend: User interface for displaying predictions and health insights.

• Backend: Machine learning model for generating predictions, API for data handling, and processing.

• Database: Secure storage of patient data and prediction results.

* **Technologies Used: Programming Languages:**

• Frontend: HTML, CSS, JavaScript (React, Angular)

* Backend: Python (Flask, Django), JavaScript (Node.js)
* Database: SQL (MySQL, PostgreSQL)
* **Frameworks:**

• Frontend: React, Angular

* Backend: Flask, Django, FastAPI
* Database: MySQL, PostgreSQL, MongoDB
* **Neural Network:**

• A neural network in a health prediction project learns patterns from medical data to accurately predict health risks and outcomes.