**Phase-3 Submission Template**

**Student Name:** HARISH P

**Register Number:** 732323104013

**Institution:** SSM COLLEGE OF ENGINEERING

**Department:** COMPUTER SCIENCE AND ENGINEERING

**Date of Submission:** 25-04-2025

**Github Repository Link:** [https://github.com/Divya231105/group-12/blob/main/Divya.py]

### **1. Problem Statement**

Transforming healthcare with Al-powered disease prediction based on patient data give the model building

### **2. Abstract**

*This project addresses the critical challenge of delayed diagnosis in healthcare by developing an AI-powered system to predict the onset of diseases using structured patient data. The objective is to leverage machine learning to identify individuals at high risk for conditions such as heart disease, diabetes, and chronic kidney disease—enabling earlier intervention and better health outcomes. The approach involves collecting and preprocessing patient data, selecting relevant features, and training classification models like Random Forest, XGBoost, and logistic regression. Hyperparameter tuning and cross-validation are used to enhance model robustness and reduce overfitting. The outcome is a predictive tool that can support clinical decision-making, improve patient care, and optimize healthcare resources. Ultimately, this AI-driven solution contributes to a more proactive and personalized healthcare system.*

### **System Requirements**

***Hardware****:* ***Development & Model Training Environment***

* + ***Processor (CPU)****: Intel i7 or AMD Ryzen 7 (minimum), Intel Xeon or AMD Threadripper (recommended for large datasets)*
  + ***Memory (RAM)****: Minimum 16 GB (32 GB or more recommended for handling large datasets)*
  + ***Storage****: At least 512 GB SSD (1 TB+ recommended if dealing with extensive patient records or high-resolution medical imaging)*
  + ***GPU*** *(Optional but recommended for deep learning or large-scale models):*
  + *NVIDIA RTX 3060 or higher (e.g., RTX 3080 / A100 for high-end deep learning)*
  + *CUDA support if using TensorFlow or PyTorch*
  + ***Deployment Environment (for real-time prediction systems)***
  + ***Cloud-based server*** *(AWS EC2, GCP Compute Engine, Azure VMs)*
  + ***Edge Device*** *(optional): For hospital systems using localized predictions*

***Software****:*

***Operating System***

* *Windows 10/11, macOS, or Linux (Ubuntu 20.04+ preferred for ML workflows)*

***Programming Language***

* *Python 3.8 or later*

***Libraries & Frameworks***

* ***Data Handling****: pandas, numpy*
* ***Machine Learning****: scikit-learn, xgboost, lightgbm*
* ***Deep Learning (optional)****: TensorFlow, Keras, or PyTorch*
* ***Visualization****: matplotlib, seaborn, plotly*
* ***Model Explainability****: SHAP, lime*
* ***Hyperparameter Tuning****: GridSearchCV, Optuna, scikit-optimize*

***Environment Management***

* *virtualenv or conda*

***Version Control***

* *Git with GitHub/GitLab/Bitbucket*

***Database (optional for production)***

* *PostgreSQL, MySQL, or MongoDB*
* *Alternatively, cloud-based data storage (e.g., AWS S3, Google Cloud Storage)*

***API/Deployment Tools (if building a web-based interface)***

* *Flask or FastAPI for REST API*
* *Docker for containerization*
* *Streamlit or Dash for front-end visualization (optional)*

**4. Objectives**

1. **Develop an AI-Based Predictive Model**

* + Build a machine learning model capable of accurately predicting the likelihood of specific diseases using patient data (e.g., EHRs, medical history, lab results).

1. **Enhance Early Diagnosis**
   * Enable early detection of high-risk diseases such as diabetes, cardiovascular diseases, or cancer, improving treatment outcomes and reducing healthcare costs.
2. **Personalize Patient Care**
   * Leverage AI insights to support personalized treatment plans based on individual risk profiles and historical health data.
3. **Improve Clinical Decision-Making**
   * Provide physicians with intelligent decision support tools that can suggest probable diagnoses or highlight at-risk patients.
4. **Integrate with Healthcare Systems**
   * Ensure the AI model can be integrated seamlessly with existing Electronic Health Record (EHR) systems for real-time predictions and alerts.

**5.Flowchart of Project Workflow**

### Start

### ↓

### 1. Data Collection

### - Patient demographics

### - Medical history

### - Lab reports

### - Imaging data (optional)

### - Wearable device data (if available)

### ↓

### 2. Data Preprocessing

### - Data cleaning (missing values, outliers)

### - Data normalization/scaling

### - Feature extraction/selection

### - Encoding categorical variables

### ↓

### 3. Exploratory Data Analysis (EDA)

### - Statistical summaries

### - Correlation analysis

### - Visualization of data distributions and trends

### ↓

### 4. Model Selection

### Choose suitable algorithms (e.g., Random Forest, XGBoost, Neural Networks)

### - Define performance metrics (Accuracy, AUC, Precision, Recall, F1)

### ↓

### 5. Model Training

### - Train model(s) using training data

### - Perform cross-validation

### - Tune hyperparameter

### ↓

### 6. Model Evaluation

### - Test on validation/test dataset

### - Analyze confusion matrix, ROC curve, etc.

### - Compare model performance

### ↓

### 7. Model Interpretation

### - Use SHAP, LIME, or feature importance plots

### - Validate predictions with domain experts

### ↓

### 8. Deployment

### - Integrate into clinical decision support system (CDSS)

### - Web/mobile interface for healthcare providers.

### **↓**

### 9. Monitoring & Maintenance

### - Continuous performance tracking

- Retraining with new data

### **6.Dataset Description**

### To build an effective AI-powered disease prediction system, the model will leverage a diverse and rich set of patient data. The data sources will be structured and unstructured and can be categorized as follows:

### 1.Demographic Information

### Fields: Age, Gender, Ethnicity, Race, Location, Socioeconomic status

### Purpose: Understand disease prevalence in specific populations and personalize predictions.

### 2. Electronic Health Records (EHRs)

### Fields: Medical history, previous diagnoses, prescribed medications, allergies, clinical notes

### Purpose: Serve as the primary source of longitudinal patient data for disease trend analysis and risk prediction.

### 3. Lab Test Results

### Fields: Blood test results (CBC, HbA1c, cholesterol, etc.), urine tests, liver/kidney function tests, etc.

### Purpose: Identify abnormal clinical indicators that may signal disease onset or progression.

### 4. Vital Signs

### Fields: Blood pressure, heart rate, respiratory rate, body temperature, BMI

### Purpose: Track physiological parameters to detect early signs of acute or chronic conditions.

### 5. Medical Imaging Data *(Optional, if incorporated)*

### Fields: X-rays, CT scans, MRIs, ultrasound images

### Purpose: Enhance prediction accuracy through image-based diagnostic insights (requires deep learning/CNNs).

### 6. Genetic and Genomic Data *(Optional, for personalized medicine)*

### Fields: DNA sequence, family history of disease, gene expression profiles

### Purpose: Support precision medicine approaches by identifying genetic predispositions to diseases.

### 7. Lifestyle and Behavioral Data

### Fields: Smoking status, alcohol use, diet, physical activity, sleep patterns

### Purpose: Account for modifiable risk factors that influence disease development.

### 8. Healthcare Utilization Data

### Fields: Number of hospital visits, emergency room usage, frequency of consultations

### Purpose: Analyze patient engagement with healthcare services to identify at-risk individuals.

### 9. Insurance and Billing Data *(For economic analysis, if relevant)*

### Fields: Type of insurance, cost of treatment, billing codes

### Purpose: Evaluate the financial impact of predictive interventions on healthcare systems.

### **7. Data Preprocessing**

*Preprocessing is critical in ensuring high-quality, reliable input for AI models. Here's a detailed step-by-step guide for preprocessing patient data:*

***1. Data Collection***

* *Source data from Electronic Health Records (EHRs), lab test results, wearable devices, imaging reports, genomics, and patient history.*
* *Ensure diverse, representative data across age, gender, ethnicity, and medical conditions.*

***2. Data Cleaning***

* ***Handle missing values:*** *Impute (mean/median/mode), remove, or flag incomplete entries.*
* ***Remove duplicates:*** *Ensure each patient record is unique.*
* ***Correct errors:*** *Fix inconsistencies in dates, units, or misentered values (e.g., "Mlae" → "Male").*

***3. Data Normalization/Standardization***

* *Normalize continuous variables (e.g., glucose level, blood pressure) using Min-Max scaling or Z-score normalization.*
* *Standardize units (e.g., mg/dL vs. mmol/L).*

***4. Categorical Encoding***

* *Convert categorical variables (e.g., gender, diagnosis codes) into numerical values using:*
  + ***One-hot encoding*** *for non-ordinal categories.*
  + ***Label encoding*** *for ordinal variables.*

***5. Outlier Detection and Treatment***

* *Use statistical methods (IQR, Z-score) or visualization (box plots) to detect outliers.*
* *Treat based on domain knowledge—either remove or cap/floor them.*

***6. Feature Engineering***

* *Derive new features like* ***BMI****,* ***disease risk scores****,* ***time since last diagnosis****, etc.*
* *Aggregate historical patient visits or lab results for trend analysis.*

***7. Dimensionality Reduction (if needed)***

* *Use PCA, t-SNE, or feature selection techniques to reduce dimensionality without losing important information.*

***8. Data Splitting***

* *Divide data into* ***training****,* ***validation****, and* ***test*** *sets (e.g., 70/15/15).*
* *Use stratified sampling if working with imbalanced disease classes.*

***9. Handling Imbalanced Data***

* *If disease cases are rare, use techniques like:*
  + ***SMOTE (Synthetic Minority Oversampling Technique)***
  + ***Class weighting*** *during model training*
  + ***Undersampling*** *majority class*

***10. Ensure Privacy & Security***

* *De-identify or anonymize data.*
* *Apply data encryption and comply with regulations (e.g., HIPAA, GDPR).*

### **8.Exploratory Data Analysis (EDA)**

**1. Understand the Dataset**

* **Common Features in Healthcare Data**:
  + Patient ID
  + Age, Gender
  + Vitals: Blood pressure, Heart rate, BMI
  + Lab Results: Glucose, Cholesterol, Hemoglobin, etc.
  + Lifestyle: Smoking, Alcohol, Exercise
  + History: Diagnosed conditions, Medications
  + Outcome variable: Presence of disease (e.g., Diabetes, Heart Disease)

📌 **Goal**: Predict disease presence (classification problem) using patient attributes.

**2. Data Overview**

python

df.shape

df.info()

df.describe()

* Check for:
  + Missing values
  + Data types (numeric, categorical, date)
  + Unique values per column

**3. Missing Values & Data Quality**

python

df.isnull().sum().sort\_values(ascending=False)

* Visualize with heatmap or bar plot
* Strategies:
  + Impute with mean/median/mode
  + Use domain knowledge
  + Drop if not critical

**4. Univariate Analysis**

**🧬 Numerical Features**

python

df['age'].hist()

df['bmi'].plot(kind='box')

* Look for outliers and skewed distributions

**🧬 Categorical Features**

python

df['gender'].value\_counts().plot(kind='bar')

* Distribution of categories (e.g., gender, smoker)

**5. Bivariate Analysis**

**📈 Numeric vs Outcome**

python

sns.boxplot(x='disease', y='age', data=df)

* See how features differ by disease status

**📊 Categorical vs Outcome**

python

pd.crosstab(df['smoker'], df['disease']).plot(kind='bar', stacked=True)

* Association between lifestyle habits and disease

**6. Correlation Analysis**

python

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

* Detect multicollinearity
* Identify features most correlated with disease

**7. Dimensionality Reduction (Optional)**

* PCA or t-SNE for visual clustering
* Useful for high-dimensional datasets

**8. Target Distribution**

python

df['disease'].value\_counts(normalize=True).plot(kind='bar')

* Check for class imbalance
* May need resampling techniques (SMOTE, etc.)

**9. Initial Insights**

* Summarize:
  + Risk factors correlated with disease
  + Feature distributions and patterns
  + Data quality issues to address before modeling

**9.Feature Engineering**

***1. Handling Missing Values***

* ***Numerical columns*** *→ Impute with* ***median*** *or* ***mean***
* ***Categorical columns*** *→ Impute with* ***mode*** *or create* ***"Unknown"*** *category*

*python*

*df['bmi'].fillna(df['bmi'].median(), inplace=True)*

*df['smoking\_status'].fillna('Unknown', inplace=True)*

***2. Encoding Categorical Variables***

* *Label encode binary categories (e.g., gender)*
* *One-hot encode multi-category features (e.g., smoking status)*

*python*

*df['gender'] = df['gender'].map({'Male': 0, 'Female': 1})*

*df = pd.get\_dummies(df, columns=['smoking\_status'], drop\_first=True)*

***3. Create New Features***

*Here’s where AI can shine through* ***domain-informed features****:*

| ***New Feature*** | ***Logic*** | ***Reason*** |
| --- | --- | --- |
| *age\_group* | *Binned age into decades* | *Risk varies by age* |
| *bmi\_category* | *Underweight, Normal, Overweight, Obese* | *Links to heart/diabetes* |
| *risk\_score* | *Composite of glucose, cholesterol, BMI* | *Early risk assessment* |
| *has\_family\_history* | *From patient history* | *Strong predictor* |

*python*

*df['age\_group'] = pd.cut(df['age'], bins=[0, 30, 45, 60, 75, 100], labels=['<30', '30-45', '45-60', '60-75', '75+'])*

*df['bmi\_category'] = pd.cut(df['bmi'], bins=[0, 18.5, 25, 30, 100], labels=['Underweight', 'Normal', 'Overweight', 'Obese'])*

*df['risk\_score'] = df['glucose']\*0.3 + df['cholesterol']\*0.3 + df['bmi']\*0.4*

***4. Outlier Detection & Treatment***

* *Remove or cap extreme outliers (important for tree-based or linear models)*

*python*

*Q1 = df['glucose'].quantile(0.25)*

*Q3 = df['glucose'].quantile(0.75)*

*IQR = Q3 - Q1*

*df = df[(df['glucose'] >= Q1 - 1.5 \* IQR) & (df['glucose'] <= Q3 + 1.5 \* IQR)]*

***5. Feature Scaling (if needed)***

* *For algorithms like* ***Logistic Regression, SVM, KNN****, apply scaling.*

*python*

*from sklearn.preprocessing import StandardScaler*

*num\_cols = ['age', 'glucose', 'bmi', 'cholesterol']*

*df[num\_cols] = StandardScaler().fit\_transform(df[num\_cols])*

***✅ Final Steps Before Modeling***

* *Define* ***X (features)*** *and* ***y (target variable)****:*

*python*

*X = df.drop('disease', axis=1)*

*y = df['disease']*

### **10.Model Building**

### you can use various ML models depending on complexity and interpretability

### Baseline Models:

### Python

### from sklearn.linear\_model import LogisticRegression

### from sklearn.tree import DecisionTreeClassifier

### from sklearn.ensemble import RandomForestClassifier

### model = RandomForestClassifier()

### model.fit(X\_train, y\_train)

### Advanced Models:

### Gradient Boosting (XGBoost, LightGBM)

### Deep Learning (for time series, EHR, imaging, etc.)

### AutoML frameworks (e.g., AutoKeras, H2O)

### **11. Model Evaluation**

* *Show evaluation metrics: accuracy, F1-score, ROC, RMSE, etc.*
* *Visuals: Confusion matrix, ROC curve, etc.*
* *Error analysis or model comparison table*
* *Include all screenshots of outputs*

### **Deployment**

* *Deploy using a free platform:*
  + *Streamlit Cloud*
  + *Gradio + Hugging Face Spaces*
  + *Flask API on Render or Deta*
* *Include:*
  + *Deployment method*
  + *Public link*
  + *UI Screenshot*
  + *Sample prediction output*

1. Source code:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

# Load your patient data CSV

data = pd.read\_csv("patient\_data.csv") # Replace with your actual dataset

# Example: columns = ['age', 'sex', 'blood\_pressure', 'cholesterol', ..., 'disease']

# Split features and target

X = data.drop("disease", axis=1) # Independent variables

y = data["disease"] # Target variable

# Preprocessing

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Model: Random Forest

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train\_scaled, y\_train)

# Predictions

y\_pred = model.predict(X\_test\_scaled)

# Evaluation

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

1. Future code:

1. Integration with Genomics and Precision Health

* With decreasing costs in genome sequencing, AI will leverage genomic data for predicting genetic disorders or cancer susceptibility.
* Can assist in identifying biomarkers for rare diseases.

2. Real-Time Clinical Decision Support

* AI will provide decision support at the point of care, suggesting possible diagnoses and actions to clinicians in real time.

3. Ethical AI and Explainability

* Future systems will focus more on transparency, explainability of predictions, and bias mitigation, ensuring trust among healthcare providers and patients.

4. Global Health Accessibility

* AI tools can enable disease prediction in low-resource settings, bridging gaps in healthcare access, especially in developing countries.

5. Regulatory Approvals and Standardization

* Regulatory bodies (like FDA, EMA) are increasingly working to approve and standardize AI tools, which will pave the way for broader adoption in clinical settings.

1. **Team Members and Roles**

*1.G.DIVYA DHARSHINI=>(EDA,MODEL DEVELOPMENT,SOURCE CODE)*

*2.V.VARUN=>(FEATURE ENGINEERING,REPORTING)*

*3.S.YALARASU=>(DOCUMENTATION,DATA CLEANING)*

*4.P.HARISH=>(FLOW CHART,TECHNOLOGIES USED REPORT)*