# Correcting the code error by completing the function call for PDF output
from fpdf import FPDF
# Create instance of FPDF class
pdf = FPDF()
pdf.add\_page()
# Set title and metadata
pdf.set\_title("Heart Disease Risk Prediction Project")
pdf.set\_author("Generated by ChatGPT")
# Title
pdf.set\_font("Arial", 'B', 16)
pdf.cell(0, 10, "Heart Disease Risk Prediction Project", ln=True, align='C')
pdf.ln(10)
# Section: Aim and Description
pdf.set\_font("Arial", 'B', 12)
pdf.cell(0, 10, "Aim:", ln=True)
pdf.set\_font("Arial", '', 12)
pdf.multi\_cell(0, 10, "Build a UI allowing users to input health metrics. Develop a machine learning model to predict the risk of heart disease.")
pdf.set\_font("Arial", 'B', 12)
pdf.cell(0, 10, "Description:", ln=True)
pdf.set\_font("Arial", '', 12)
pdf.multi\_cell(0, 10, "Create a user-friendly interface for inputting health data and implement a model (e.g., Random Forest) for risk assessment.")
pdf.ln(5)
# Section: Technologies Used
pdf.set\_font("Arial", 'B', 12)
pdf.cell(0, 10, "Technologies Used:", ln=True)
pdf.set\_font("Arial", '', 12)
pdf.multi\_cell(0, 10, "- Python\n- Flask or Streamlit for UI\n- Scikit-Learn for machine learning\n- Pandas & NumPy for data manipulation\n- Joblib for saving/loading model")
pdf.ln(5)
# Section: Project Repository Structure
pdf.set\_font("Arial", 'B', 12)
pdf.cell(0, 10, "Project Repository Structure:", ln=True)
pdf.set\_font("Arial", '', 12)
pdf.multi\_cell(0, 10, """
heart-disease-prediction/
- app.py : Main application file (Flask or Streamlit)
- model.py : Machine learning model training script
- requirements.txt : Required dependencies
- heart.csv : Dataset (ensure you have a valid heart dataset)
- README.md : Project documentation
- templates/ : HTML files (if using Flask)
- index.html
- static/ : Static files (e.g., CSS, images) (optional)
""")
pdf.ln(5)
# Section: Code Files
code\_files = {
"model.py": """
import pandas as pd
from sklearn.model\_selection import train\_test\_split
from sklearn.ensemble import RandomForestClassifier
from [sklearn.me](sklearn.me#sklearn.me)trics import accuracy\_score
import joblib
df = pd.read\_csv('heart.csv')
X = df.drop('target', axis=1)
y = df['target']
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)
model = RandomForestClassifier(n\_estimators=100, random\_state=42)
model.fit(X\_train, y\_train)
y\_pred = model.predict(X\_test)
accuracy = accuracy\_score(y\_test, y\_pred)
print(f"Model Accuracy: {accuracy:.2f}")
joblib.dump(model, 'heart\_disease\_model.pkl')
""",
"app.py (Flask version)": """
from flask import Flask, request, render\_template
import joblib
import numpy as np
app = Flask(\_\_name\_\_)
model = joblib.load('heart\_disease\_model.pkl')
@app.route('/')
def index():
return render\_template('index.html')
@app.route('/predict', methods=['POST'])
def predict():
input\_features = [float(x) for x in request.form.values()]
input\_array = np.array([input\_features])
prediction = model.predict(input\_array)
output = 'High Risk' if prediction[0] == 1 else 'Low Risk'
return render\_template('index.html', prediction\_text=f'Heart Disease Risk: {output}')
if \_\_name\_\_ == '\_\_main\_\_':
app.run(debug=True)
""",
"app.py (Streamlit version)": """
import streamlit as st
import joblib
import numpy as np
model = joblib.load('heart\_disease\_model.pkl')
st.title('Heart Disease Risk Prediction')
age = st.number\_input('Age')
sex = st.selectbox('Sex', (0, 1))
cp = st.selectbox('Chest Pain Type', (0, 1, 2, 3))
trestbps = st.number\_input('Resting Blood Pressure')
chol = st.number\_input('Cholesterol Level')
fbs = st.selectbox('Fasting Blood Sugar > 120 mg/dl', (0, 1))
restecg = st.selectbox('Resting ECG Results', (0, 1, 2))
thalach = st.number\_input('Max Heart Rate Achieved')
exang = st.selectbox('Exercise Induced Angina', (0, 1))
oldpeak = st.number\_input('ST Depression Induced by Exercise')
slope = st.selectbox('Slope of the Peak Exercise ST Segment', (0, 1, 2))
ca = st.selectbox('Number of Major Vessels Colored by Fluoroscopy', (0, 1, 2, 3))
thal = st.selectbox('Thalassemia', (0, 1, 2, 3))
if st.button('Predict'):
input\_features = np.array([[age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal]])
prediction = model.predict(input\_features)
output = 'High Risk' if prediction[0] == 1 else 'Low Risk'
st.success(f'Heart Disease Risk: {output}')
""",
"index.html": """
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Heart Disease Prediction</title>
</head>
<body>
<h2>Heart Disease Risk Prediction</h2>
<form action="/predict" method="post">
Age: <input type="number" name="age"><br>
Sex: <input type="number" name="sex"><br>
Chest Pain Type: <input type="number" name="cp"><br>
Resting Blood Pressure: <input type="number" name="trestbps"><br>
Cholesterol Level: <input type="number" name="chol"><br>
Fasting Blood Sugar: <input type="number" name="fbs"><br>
Resting ECG: <input type="number" name="restecg"><br>
Max Heart Rate Achieved: <input type="number" name="thalach"><br>
Exercise Induced Angina: <input type="number" name="exang"><br>
ST Depression: <input type="number" name="oldpeak"><br>
Slope of ST Segment: <input type="number" name="slope"><br>
Number of Major Vessels: <input type="number" name="ca"><br>
Thalassemia: <input type="number" name="thal"><br>
<input type="submit" value="Predict">
</form>
{% if prediction\_text %}
<h3>{{ prediction\_text }}</h3>
{% endif %}
</body>
</html>
""",
"requirements.txt": """
Flask==2.1.2
streamlit==1.6.0
scikit-learn==1.0.2
pandas==1.3.5
numpy==1.21.5
joblib==1.1.0
""",
"README.md": """
# Heart Disease Risk Prediction
This project aims to build a user-friendly web application to predict the risk of heart disease using a machine-learning model.
## Technologies Used
- \*\*Python\*\*: Programming language
- \*\*Flask\*\* or \*\*Streamlit\*\*: For building the web user interface (UI)
- \*\*Scikit-Learn\*\*: For developing the machine learning model (Random Forest)
- \*\*Pandas & NumPy\*\*: For data manipulation and analysis
- \*\*Joblib\*\*: For saving and loading the machine learning model
## How to Run the Project
1. \*\*Install dependencies\*\*:
```bash
pip install -r requirements.txt
```
2. \*\*Train the model\*\*:
```bash
python model.py
```
3. \*\*Run the application\*\*:
- For \*\*Flask\*\*:
```bash
python app.py
```
- For \*\*Streamlit\*\*:
```bash
streamlit run app.py
```
## What You Learn
- UI design for health applications
- Understanding cardiovascular risk factors
- Building and deploying a classification model
## Dataset
Ensure you have the dataset (`heart.csv`) in the project directory.
"""
}
# Add code files to the PDF
for filename, code in code\_files.items():
pdf.add\_page()
pdf.set\_font("Arial", 'B', 12)
pdf.cell(0, 10, filename, ln=True)
pdf.set\_font("Courier", '', 8)
pdf.multi\_cell(0, 5, code.strip())
# Output the PDF file
pdf\_file\_path = "/mnt/data/Heart\_Disease\_Risk\_Prediction\_Project.pdf"
pdf.output