**FINAL REPORT**

|  |  |
| --- | --- |
| **Team ID** | **LTVIP2026TMIDS65771** |
| **Project Name** | **Heart Disease Analysis** |

**1. INTRODUCTION**

1.1 Project Overview

Heart Disease Analysis is a healthcare analytics project designed to predict the probability of heart disease using machine learning techniques. The system analyzes patient medical attributes such as age, cholesterol, blood pressure, ECG results, heart rate, and other clinical indicators. By applying data preprocessing and predictive modeling, the system generates accurate insights and visual dashboards. This helps healthcare professionals make faster decisions and enables users to understand their risk level through data-driven analysis.

**1.2 Purpose**

* To develop an intelligent system for early detection of heart disease risk.
* To reduce dependency on manual medical data analysis.
* To improve diagnostic support using machine learning.
* To visualize healthcare data using dashboards and graphs.
* To provide an easy-to-use platform for prediction and analysis.

**2. IDEATION PHASE**

**2.1 Problem Statement**

****Heart disease continues to be one of the major causes of death worldwide. Doctors and healthcare staff handle large volumes of patient data, making it difficult to quickly identify high-risk patients. Existing systems lack clear visualization and predictive analysis. This project aims to create a smart solution that uses machine learning to analyze patient data and provide accurate heart disease predictions.

**2.2 Empathy Map Canvas**

**Think & Feel**

* Worried about heart health.
* Need early diagnosis.
* Fear of wrong diagnosis.

**See**

* Complex medical reports.
* Multiple health parameters.
* Lack of simple analysis tools.

**Hear**

* Advice from doctors for regular monitoring.
* Importance of lifestyle changes.

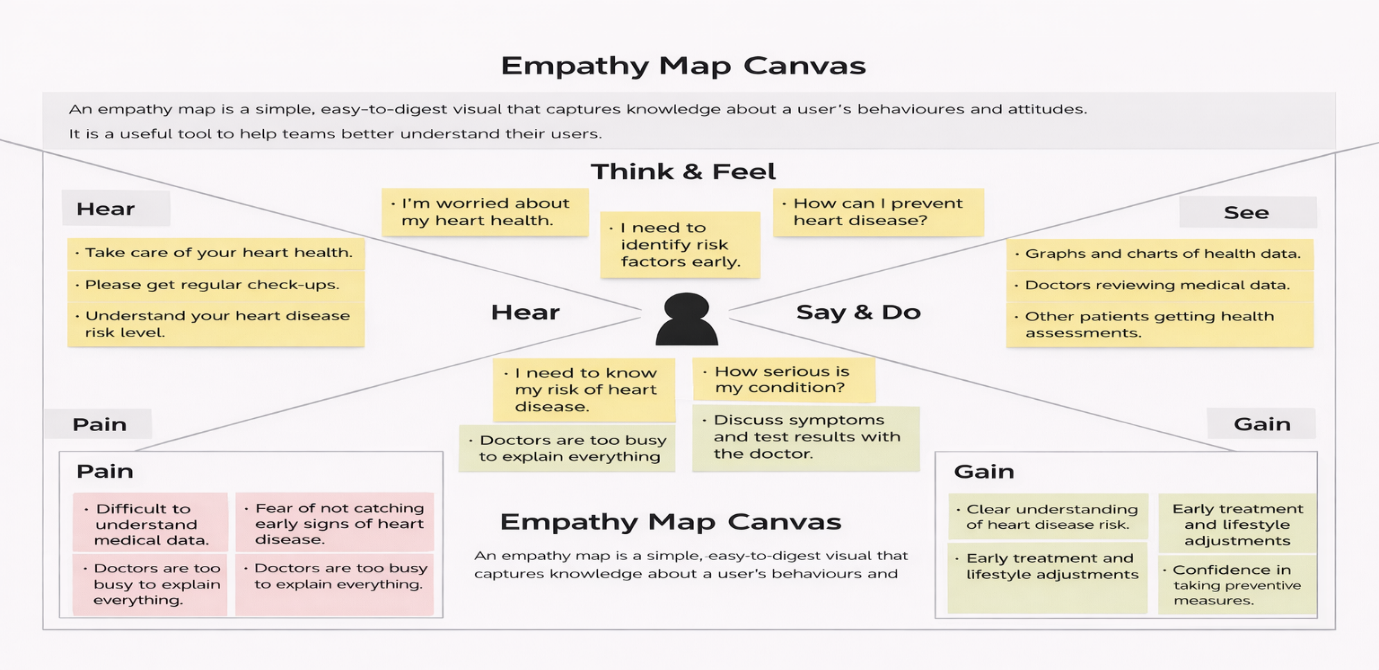
**Say & Do**

* Search for health information online.
* Visit hospitals for diagnosis.
* Want quick and understandable results.

**Pain**

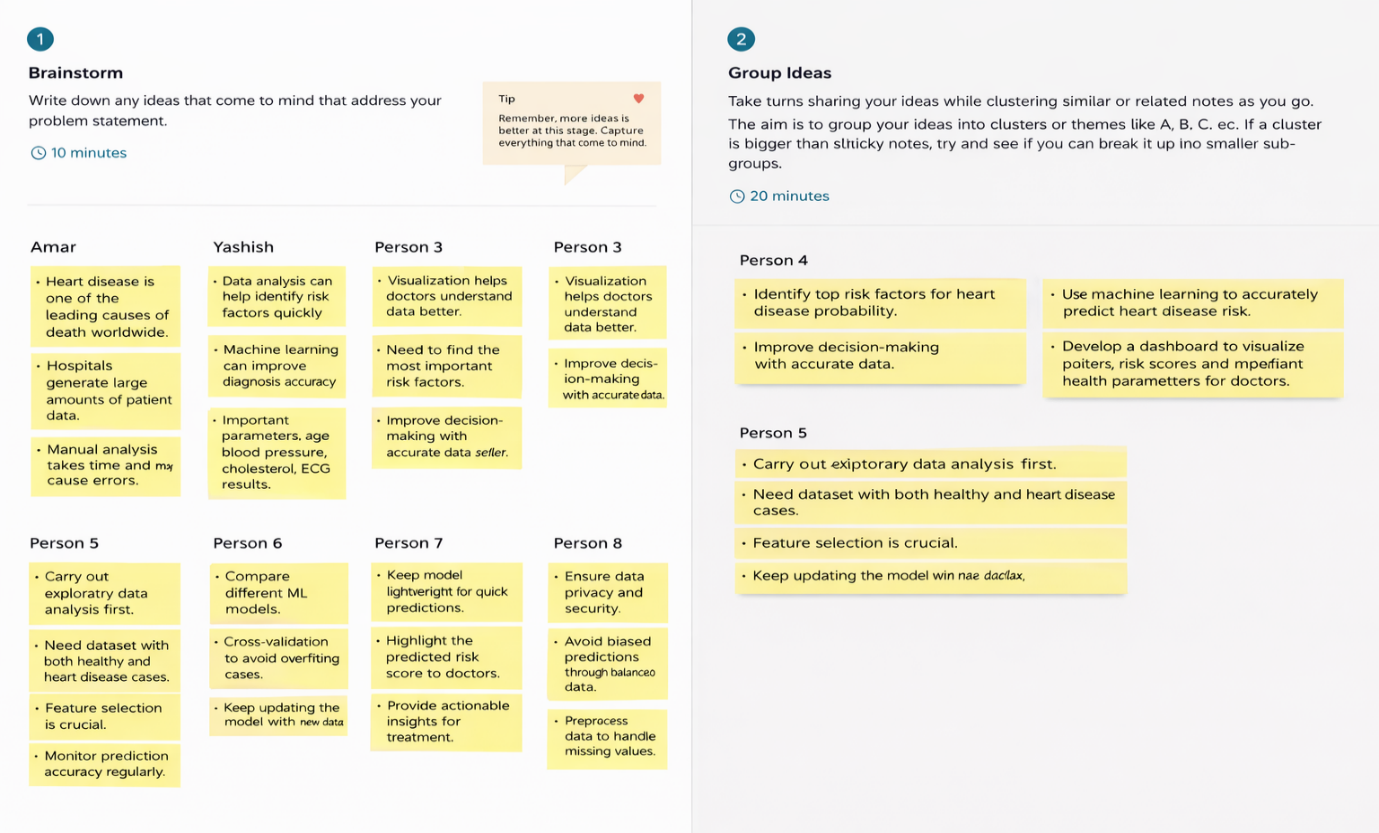
* Difficult to understand reports.
* Time-consuming diagnosis process.

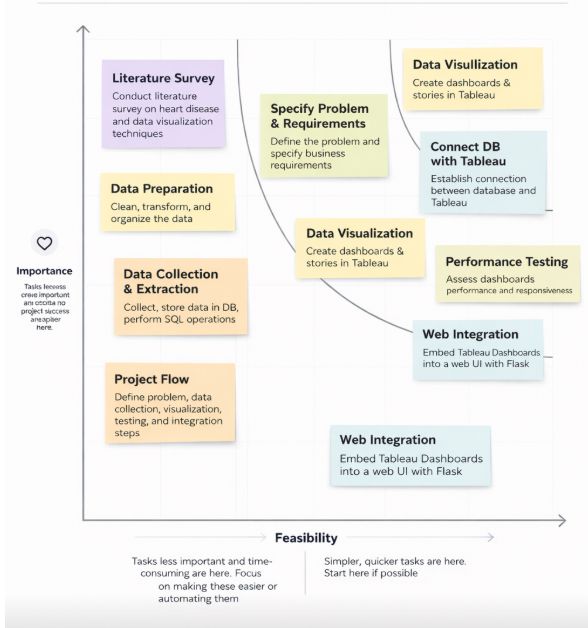
**Gain**

* Early prediction saves lives.
* Easy visualization improves understanding.

**2.3 Brainstorming**

* Use machine learning classification models.
* Build interactive dashboards.
* Enable data filtering and analysis.
* Generate automated reports.
* Compare multiple ML models for accuracy.
* Allow users to upload patient datasets.

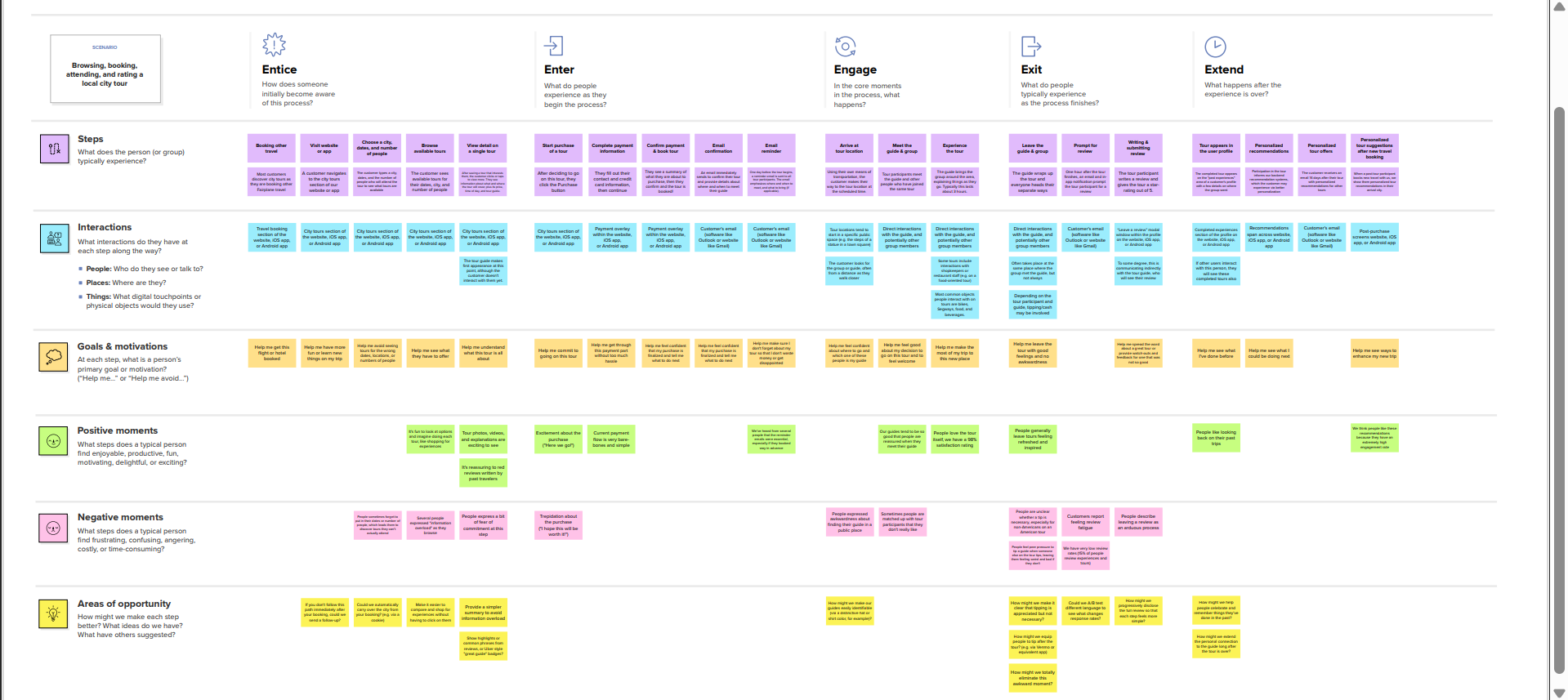


****

**3. REQUIREMENT ANALYSIS**

**3.1 Customer Journey Map**

1. User opens application.
2. Registration/Login process.
3. Uploads medical data.
4. Data preprocessing occurs.
5. Machine learning model predicts risk.
6. Results shown on dashboard.
7. User downloads report or views insights.



**3.2 Solution Requirement**

| FR No. | Functional Requirement (Epic) | Sub Requirement (Story / Sub-Task) |
| --- | --- | --- |
| FR-1 | User Registration | Registration through Form, Registration through Gmail, Registration through LinkedIn |
| FR-2 | User Confirmation | Confirmation via Email, Confirmation via OTP |
| FR-3 | User Login & Authentication | Login using Email & Password, Forgot Password, Session Management |
| FR-4 | Data Upload & Management | Upload patient health data, Validate input data, Store data securely |
| FR-5 | Data Preprocessing | Handle missing values, Feature scaling, Data normalization |

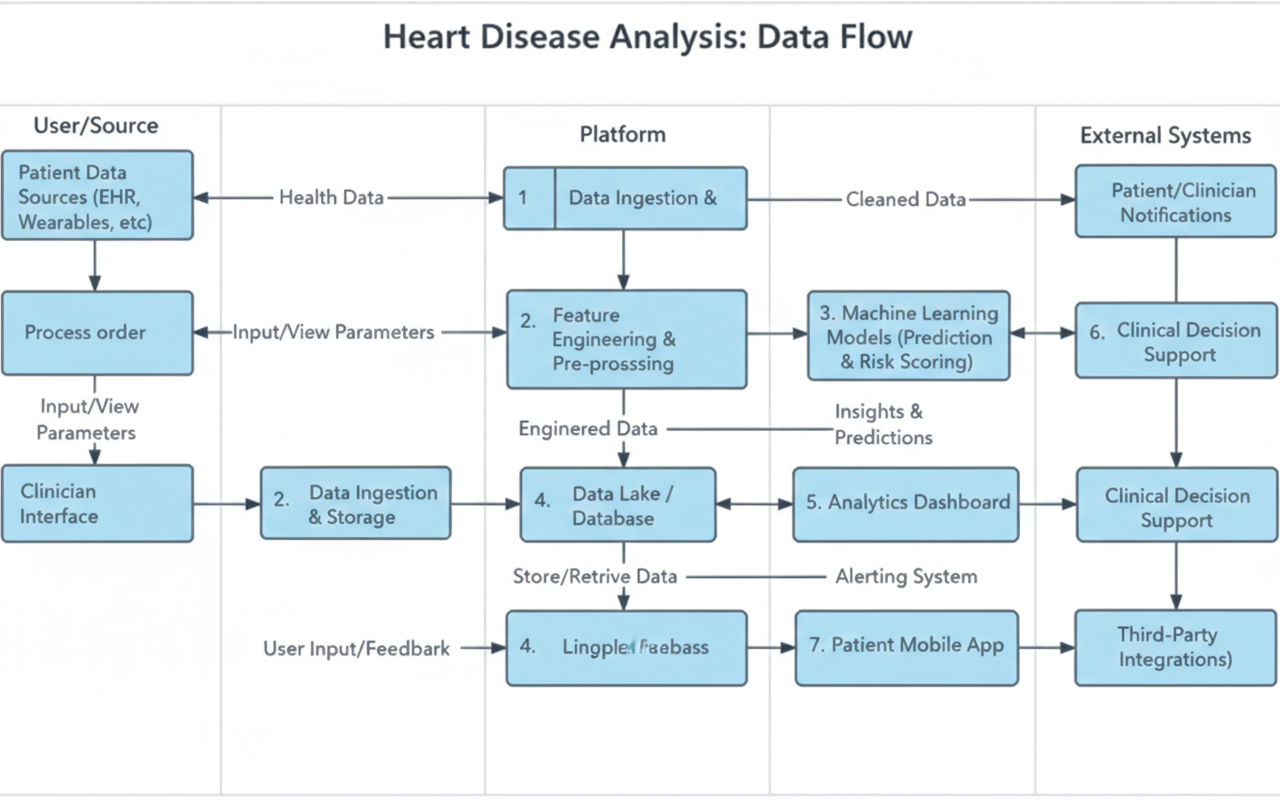
**Non-functional Requirements:**

Following are the non-functional requirements of the proposed solution.

| **FR No.** | **Non-Functional Requirement** | **Description** |
| --- | --- | --- |
| NFR-1 | Usability | The system should be easy to use with a simple and intuitive interface for users and doctors. |
| NFR-2 | Security | User data and medical information must be securely stored with authentication and authorization mechanisms. |
| NFR-3 | Reliability | The system should provide accurate and consistent prediction results with minimal failures. |
| NFR-4 | Performance | The system should process data and display predictions quickly with minimal response time. |
| NFR-5 | Availability | The application should be accessible whenever required with minimal downtime. |
| NFR-6 | Scalability | The system should support increased users and larger datasets without performance degradation. |

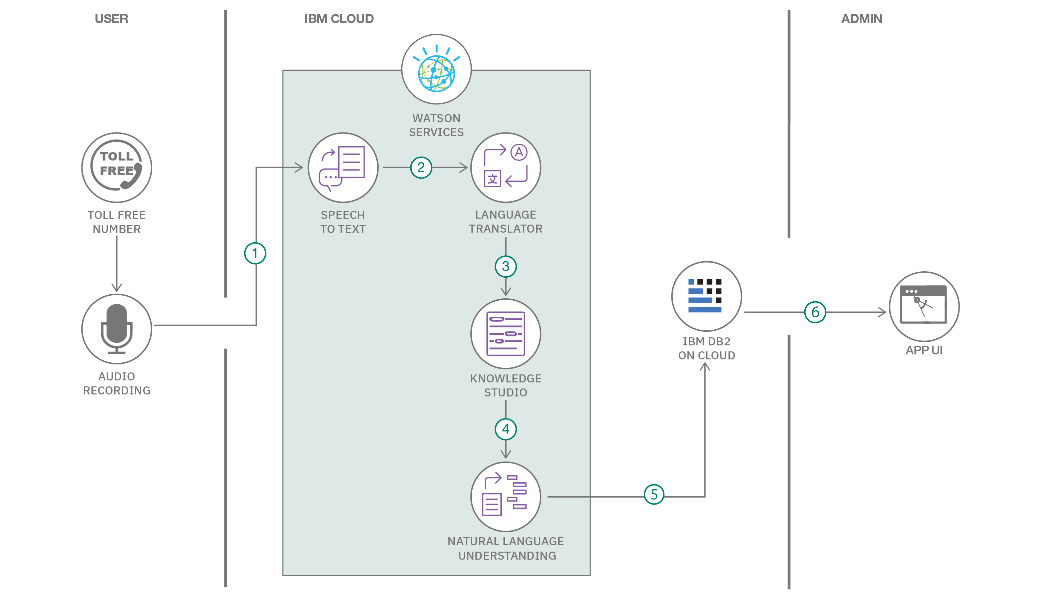
**3.3 Data Flow Diagram**

* Data Input → Data Cleaning → Feature Selection → Model Processing → Prediction Output → Dashboard Visualization → Report Generation.



**3.4 Technology Stack**

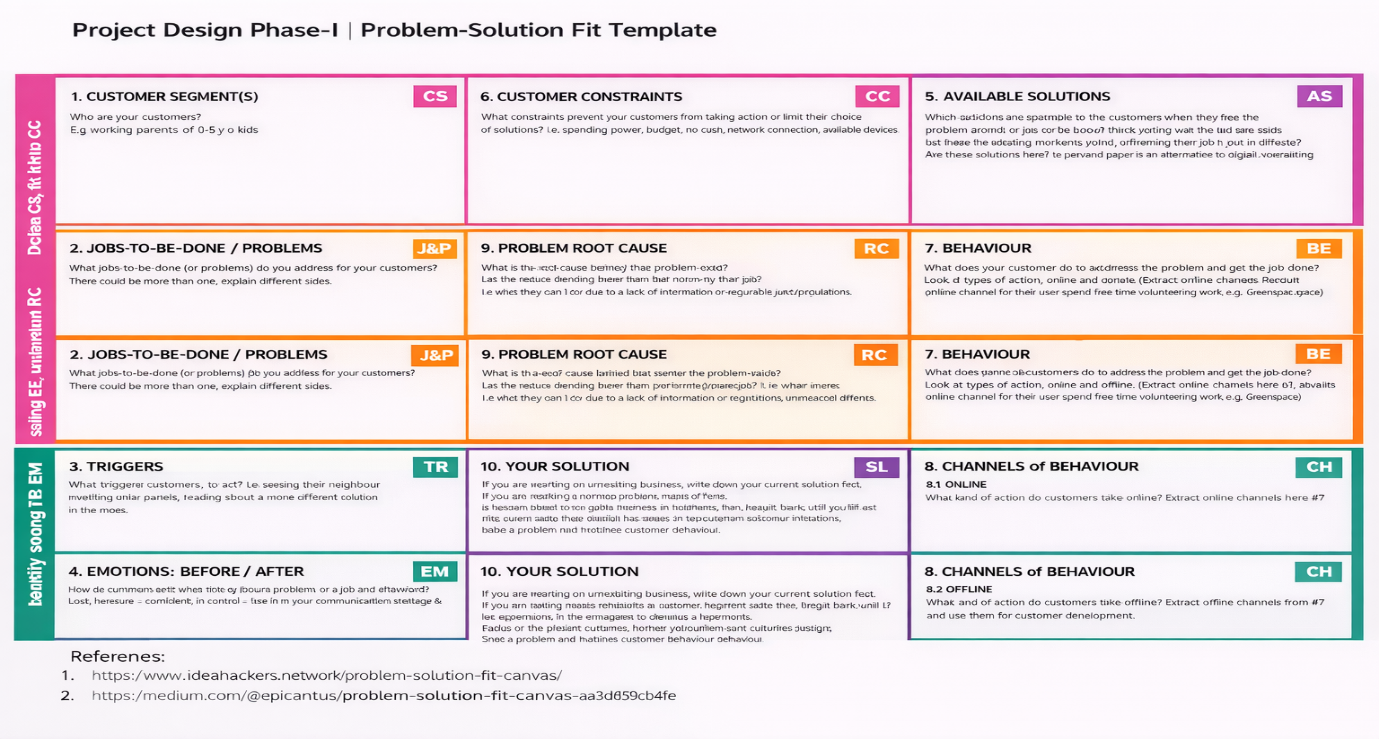
* **Programming Language:** Python
* **Libraries:** Pandas, NumPy, Scikit-learn, Matplotlib, Seaborn
* **Frontend:** HTML, CSS / Dashboard tools
* **Backend:** Python Flask / Django (if used)
* **Database:** CSV / Dataset files
* **IDE:** Jupyter Notebook / VS Code
* **Version Control:** GitHub



**4. PROJECT DESIGN**

**4.1 Problem Solution Fit**

The project identifies healthcare professionals and patients as primary users. The system solves the challenge of slow and manual data analysis by providing automated prediction and visualization. The solution aligns with user needs by giving quick, understandable insights.



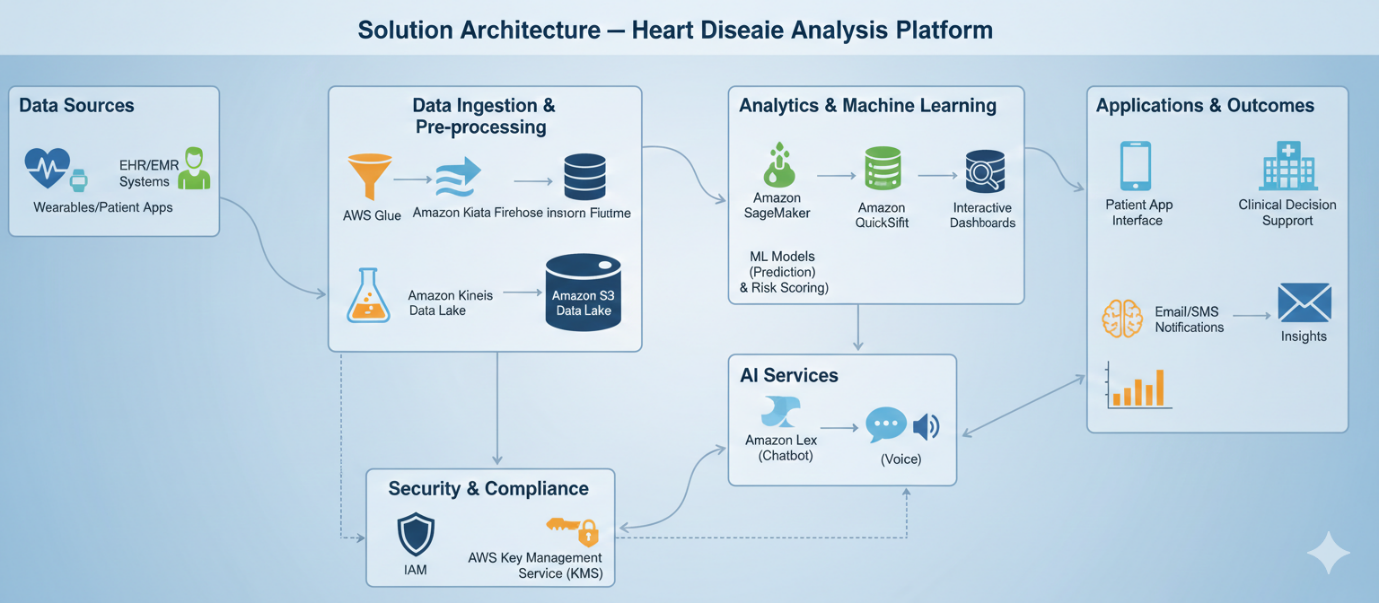
**4.2 Proposed Solution**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
|  | Problem Statement (Problem to be solved) | Heart diseases are one of the leading causes of death worldwide. Early detection is difficult because symptoms may appear late, and manual analysis of medical data can be time-consuming and error-prone. There is a need for a reliable system that can analyze heart-related data and assist in early diagnosis. |
|  | Idea / Solution description | The proposed solution is an intelligent heart analysis system that uses patient medical data such as ECG results, blood pressure, cholesterol levels, age, and other health parameters to predict the risk of heart disease. The system applies machine learning techniques to analyze data quickly and provide accurate predictions, helping doctors in decision-making. |
|  | Novelty / Uniqueness | The uniqueness of this solution lies in combining multiple health parameters into a single predictive model that provides fast and data-driven insights. It reduces human error, supports early risk prediction, and offers an easy-to-use interface for both healthcare professionals and patients. |
|  | Social Impact / Customer Satisfaction | This solution helps in early detection of heart problems, which can save lives and reduce healthcare costs. Patients gain awareness about their heart health, while doctors receive reliable analytical support, leading to better treatment outcomes and higher satisfaction. |
|  | Business Model (Revenue Model) | The system can generate revenue through hospital subscriptions, diagnostic center licensing, and healthcare partnerships. Additional income can be generated through premium analytics features, cloud-based access, and maintenance services for healthcare institutions. |
|  | Scalability of the Solution | The solution can be easily scaled by integrating with hospitals, clinics, and telemedicine platforms. It can handle large datasets and be expanded to include wearable device data, enabling real-time heart monitoring and wider adoption across different regions. |

The proposed solution includes:

* Data collection from healthcare datasets.
* Data cleaning and normalization.
* Machine learning model training.
* Risk prediction generation.
* Dashboard visualization with graphs.
* Report generation for users.

**4.3 Solution Architecture**

****

**Layers:**

1. Data Collection Layer
2. Preprocessing Layer
3. Feature Engineering Layer
4. Machine Learning Model Layer
5. Prediction Output Layer
6. Visualization Dashboard Layer
7. User Interface Layer

**5. PROJECT PLANNING & SCHEDULING**

**5.1 Project Planning**

* Agile development model followed.
* Work divided into multiple sprints.
* Tasks assigned based on modules:
  + User authentication
  + Data processing
  + Model development
  + Dashboard design
  + Testing & evaluation
* Regular sprint reviews performed.

**6. FUNCTIONAL AND PERFORMANCE TESTING**

**6.1 Performance Testing**

* Data loading performance checked.
* Preprocessing execution speed tested.
* Model accuracy evaluated.
* Dashboard loading time tested.
* Multiple dataset sizes tested for stability.
* Prediction latency measured.

**7. RESULTS**

**7.1 Output Screenshots**

* Registration Page
* Login Interface
* Dataset Upload Screen
* Preprocessing Output
* Prediction Result Page
* Dashboard with graphs
* Accuracy and confusion matrix output

**8. ADVANTAGES & DISADVANTAGES**

**Advantages**

* Early detection of heart disease risk.
* Reduced manual analysis effort.
* Improved healthcare decision support.
* Interactive visualization dashboard.
* Supports large data analysis.

**Disadvantages**

* Prediction depends on data quality.
* Requires correct patient data input.
* Model accuracy may vary with dataset changes.

**9. CONCLUSION**

The Heart Disease Analysis system successfully demonstrates how machine learning can be applied in healthcare analytics. The project provides accurate prediction results, interactive dashboards, and meaningful insights that help doctors and users make informed decisions. It shows the importance of data-driven healthcare solutions.

**10. FUTURE SCOPE**

* Integration with hospital live databases.
* Mobile application development.
* AI-based real-time monitoring.
* Deep learning models for improved accuracy.
* Cloud deployment for scalability.
* Multi-disease prediction platform.

**11. APPENDIX**

* **Source Code**

# Heart Disease Prediction Project

* from flask import Flask, jsonify, request, send\_from\_directory
* from flask\_cors import CORS
* import os
* from database import init\_db, db
* from models import HeartData
* from sqlalchemy import func
* # ✅ Create ONLY ONE Flask app
* BASE\_DIR = os.path.abspath(os.path.dirname(\_\_file\_\_))
* app = Flask(
* \_\_name\_\_,
* static\_folder=os.path.join(BASE\_DIR, "../frontend/dist"),
* static\_url\_path="/"
* )
* CORS(app)
* # Initialize Database
* init\_db(app)
* # ---------------- API ROUTES ---------------- #
* @app.route('/api/project-info', methods=['GET'])
* def get\_project\_stats():
* try:
* total\_records = HeartData.query.count()
* disease\_count = HeartData.query.filter\_by(heart\_disease='Yes').count()
* avg\_bmi = db.session.query(func.avg(HeartData.bmi)).scalar()
* return jsonify({
* "title": "Heart Disease Indicators",
* "description": "Analysis of key personal health indicators.",
* "stats": {
* "dataset\_size": f"{total\_records} Records",
* "avg\_bmi": f"{round(avg\_bmi, 1)}" if avg\_bmi else "N/A",
* "patients\_with\_disease": disease\_count,
* "visualizations": "8 Unique Charts"
* },
* "status": "Database Connected"
* })
* except Exception as e:
* return jsonify({"error": str(e), "status": "DB Connection Failed"}), 500
* @app.route('/api/data', methods=['GET'])
* def get\_data():
* page = request.args.get('page', 1, type=int)
* per\_page = 10
* pagination = HeartData.query.paginate(page=page, per\_page=per\_page, error\_out=False)
* data = [item.to\_dict() for item in pagination.items]
* return jsonify({
* "data": data,
* "total": pagination.total,
* "pages": pagination.pages,
* "current\_page": page
* })
* @app.route('/api/performance', methods=['GET'])
* def get\_performance():
* try:
* row\_count = HeartData.query.count()
* db\_status = "Connected"
* except:
* row\_count = 0
* db\_status = "Disconnected"
* import random
* render\_time = round(random.uniform(0.8, 1.4), 2)
* return jsonify({
* "db\_rows": f"{row\_count} Rows",
* "filters": "5 Context Filters",
* "calc\_fields": "12 Fields",
* "charts": "8 Unique Charts",
* "render\_time": f"< {render\_time}s",
* "status": db\_status
* })
* # ---------------- SERVE REACT FRONTEND ---------------- #
* @app.route("/", defaults={"path": ""})
* @app.route("/<path:path>")
* def serve(path):
* full\_path = os.path.join(app.static\_folder, path)
* if path != "" and os.path.exists(full\_path):
* return send\_from\_directory(app.static\_folder, path)
* return send\_from\_directory(app.static\_folder, "index.html")
* # ---------------- MAIN ---------------- #
* if \_\_name\_\_ == '\_\_main\_\_':
* with app.app\_context():
* db.create\_all()
* app.run(debug=True, port=5000)
* **Dataset Link (UCI Heart Disease Dataset)**



* **GitHub Repository Link:**
* [**https://github.com/KESANI-SANTHOSH-KUMAR/Heart\_Disease\_Analysis**](https://github.com/KESANI-SANTHOSH-KUMAR/Heart_Disease_Analysis)
* **Project Demo Link:**

<https://drive.google.com/file/d/1ful7lDSPYaR6L9bt8dYhy8xsdKVVMRRx/view?usp=sharing>