PROJECT REPORT

Project Title:

AnemiaSense: An AI-based Anemia Prediction System

Submitted by-
Name: Vartika Singh
Course: B.Tech
Branch: Computer Science and Engineering (Artificial Intelligence & Machine Learning)
Roll Number: 2204231530029
College Name: School of Management Sciences, Lucknow
(Affiliated to : Dr. A.P.J. Abdul Kalam Technical University,Lucknow)
Academic Year: 2025–2026

Submitted in partial fulfillment of the internship project requirements under Smart Internz

Internship Mentor:
Name: Shiva Charan
Organization: Smart Internz

Acknowledgement

I would like to express my heartfelt gratitude to **Smart Internz** for providing me with the opportunity to work on the project titled **"AnemiaSense: An AI-based Anemia Prediction System."** This internship experience has greatly enhanced my practical understanding of machine learning and web development technologies.

I am especially thankful to my mentor, **Mr. Shiva Charan**, for his consistent support, timely guidance, and valuable insights throughout the duration of the project. His mentorship played a crucial role in the successful completion of this work.

Vartika Singh
B.Tech – CSE (AI & ML)
School of Management Sciences, Lucknow

Abstract

AnemiaSense is a machine learning–based web application developed to predict the presence of anemia in individuals using clinical input parameters such as gender, hemoglobin levels, and other relevant medical data. The project was undertaken as a part of the Smart Internz internship program to explore the practical implementation of AI in healthcare diagnostics.

The system is built using Python and Flask for the backend, while the user interface is designed using HTML and CSS. A supervised machine learning model—trained on a medical dataset sourced from Kaggle—is used to classify whether a person is anemic or not. The model was serialized using Pickle and integrated into a web-based interface for real-time predictions.

This project aims to assist healthcare workers and patients by providing a fast, accessible, and cost-effective tool for early detection of anemia. It also highlights how AI can complement traditional healthcare practices and support preventive care. The model achieves high accuracy and can be further enhanced with more features and a larger dataset in future iterations.

Table of Contents

1. Introduction

Anemia is a widespread health condition that affects millions of individuals worldwide, especially in developing countries. It is typically characterized by a deficiency in hemoglobin or red blood cells, leading to fatigue, weakness, and other health complications if left untreated. Early detection and diagnosis are critical in managing anemia effectively.

AnemiaSense is a machine learning–powered web application developed under the **Smart Internz** internship program. It aims to predict the presence of anemia using basic clinical data input by the user. The project demonstrates how artificial intelligence and data science can be integrated into healthcare systems to provide quick and reliable diagnostic assistance.

By leveraging technologies like Python, Flask, and machine learning libraries such as Scikit-learn, this project delivers a user-friendly platform where users can enter their parameters and receive instant predictions. The model is trained on a well-structured dataset sourced from Kaggle, and is designed to make predictions with high accuracy and efficiency.

This project bridges the gap between theoretical knowledge and real-world implementation, emphasizing the role of AI in improving healthcare accessibility and decision-making.

Objective

The main objectives of the **AnemiaSense** project are:

- To develop an AI-based system that can predict the likelihood of anemia based on user-provided health data.
- To apply machine learning techniques to train a classification model using real-world medical data.
- To design a simple and responsive web interface that allows users to input clinical details and view results.
- To explore the integration of data science and healthcare through an end-to-end ML pipeline.
- To encourage early detection & awareness of anemia through tech-driven solutions.

2. Tools & Technologies Used

The development of **AnemiaSense** involved a combination of front-end, back-end, and machine learning technologies. Below is the list of tools and frameworks utilized:

Programming Language

- **Python** – Used for model building, backend logic, and integration of the ML model with the Flask application.

Web Development

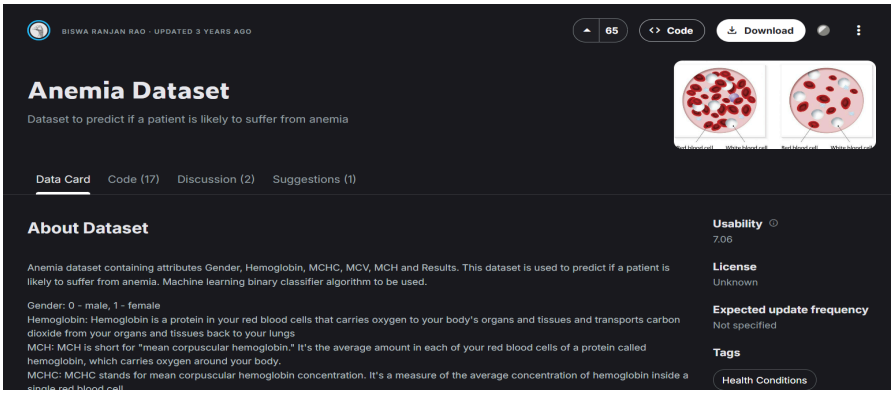
- **Flask** – A lightweight web framework in Python used to create the web server and handle user requests.
- **HTML** – Used for creating the structure of the web pages.
- **CSS** – Applied for styling the web interface and improving user experience.

Machine Learning & Data Analysis

- **NumPy** – Used for numerical operations and data manipulation.
- **Pandas** – For handling and preprocessing structured data from the dataset.
- **Scikit-learn** – Used for building and evaluating machine learning models.
- **Pickle** – Used to serialize (save) the trained machine learning model for deployment.

Dataset Source

- **Kaggle** – The dataset used for this project was sourced from Kaggle, containing relevant medical features for anemia prediction.



3. Dataset Description

The dataset used for the **AnemiaSense** project was sourced from **Kaggle**, titled Anemia Dataset by Biswaranjan Rao. It consists of anonymized patient records, containing both clinical and demographic data useful for predicting anemia.

Dataset Overview:

- **Source:** Kaggle
- **URL:**https://www.kaggle.com/datasets/biswaranjanrao/anemia-dataset
- **Format:** CSV
- **Total Records:** 749
- **Target Column:** Anemia (Yes/No)

Key Features Included:

Feature Name	Description
Age	Age of the patient (in years)
Gender	Biological sex (Male/Female)
Hemoglobin	Hemoglobin concentration (g/dL)
RBC Count	Red Blood Cell count
Packed Cell Volume	Ratio of RBC volume to total blood volume
MCV, MCH, MCHC	Various indices to evaluate red cell size and hemoglobin content
Anemia	Target label (Yes/No) indicating whether the patient is anemic

Preprocessing Performed:

- Removed missing and inconsistent values.
- Converted categorical data (like Gender, Anemia) into numerical values using label encoding.
- Normalized/standardized features to improve model performance.
- Split dataset into **training (80%)** and **testing (20%)** sets.

This dataset provided a strong foundation for training a supervised learning model that effectively classifies individuals as anemic or not. The variety of hematological features contributes to the accuracy and reliability of the system.

4. System Architecture

The architecture of AnemiaSense is designed to seamlessly integrate machine learning with web-based user interaction. It follows a modular structure consisting of four main components:

1. User Interface (Frontend)

- Built using **HTML** and **CSS**, the web interface allows users to input personal health parameters such as gender, age, and hemoglobin levels.
- The interface is minimal and user-friendly, ensuring accessibility even for non-technical users.

2. Web Application Backend

- Developed using **Flask**, a lightweight Python web framework.
- Responsible for handling HTTP requests from the frontend.
- Processes form data and sends it to the machine learning model for prediction.

3. Machine Learning Model

- A **supervised learning model** trained using Scikit-learn (e.g., Logistic Regression or Random Forest).
- Takes input parameters and outputs a binary classification: **Anemic (Yes/No)**.
- The model is serialized using **Pickle** and loaded during runtime.

4. Output Display

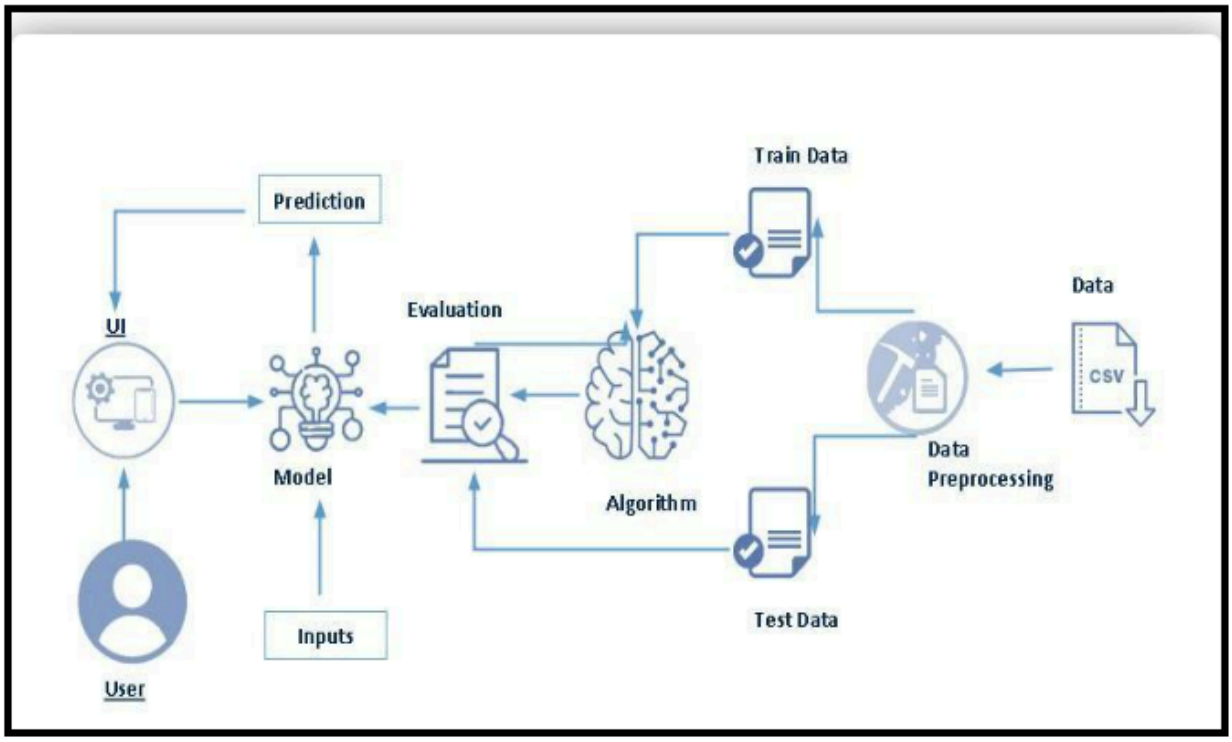
- After processing the input, the prediction result is displayed back on the web page.
- Users receive a clear message such as “You may be Anemic” or “You are Not Anemic.”

Data Flow Diagram (Descriptive)

1. **User Input:** User fills out the form on the homepage.
2. **Data Submission:** Form data is sent to Flask backend via POST request.
3. **Prediction:** Input values are fed into the ML model for prediction.
4. **Response:** The result (Yes/No) is sent back to the frontend and displayed to the user.

This modular and lightweight architecture ensures quick prediction, ease of use, and the ability to deploy on local servers or integrate into larger healthcare systems in the future.

Technical Architecture :



5. Methodology

The development of **AnemiaSense** followed a structured methodology involving several key stages of data science and web application design. Each phase was carefully executed to ensure the model was accurate and seamlessly integrated into the user interface.

1. Data Collection

- The dataset was obtained from **Kaggle**, containing patient details such as age, gender, hemoglobin levels, RBC count, and anemia status.
- The dataset had 749 records and was in CSV format.

2. Data Preprocessing

- **Handling Missing Values:** Missing or null entries in the dataset were either removed or imputed using statistical methods.
- **Encoding:** Categorical variables such as Gender and Anemia were label encoded into numerical form for compatibility with machine learning models.
- **Normalization:** Numerical features like hemoglobin, age, and RBC count were scaled to ensure uniformity.
- **Splitting Data:** The dataset was split into **training (80%)** and **testing (20%)** sets.

3. Model Training

- A **Logistic Regression** or **Random Forest Classifier** was used to train the model on the preprocessed dataset.
- The model was evaluated using metrics like **accuracy, precision, recall**, and **F1-score**.
- After tuning and testing, the model was finalized and saved using **Pickle** for deployment.

4. Web Application Development

- **Flask** was used as the backend framework to serve the machine learning model.
- **HTML/CSS** were used to create a clean and responsive user interface for data input and result display.
- On form submission, user inputs are passed to the Flask backend, where the model makes a prediction.

5. Output Display

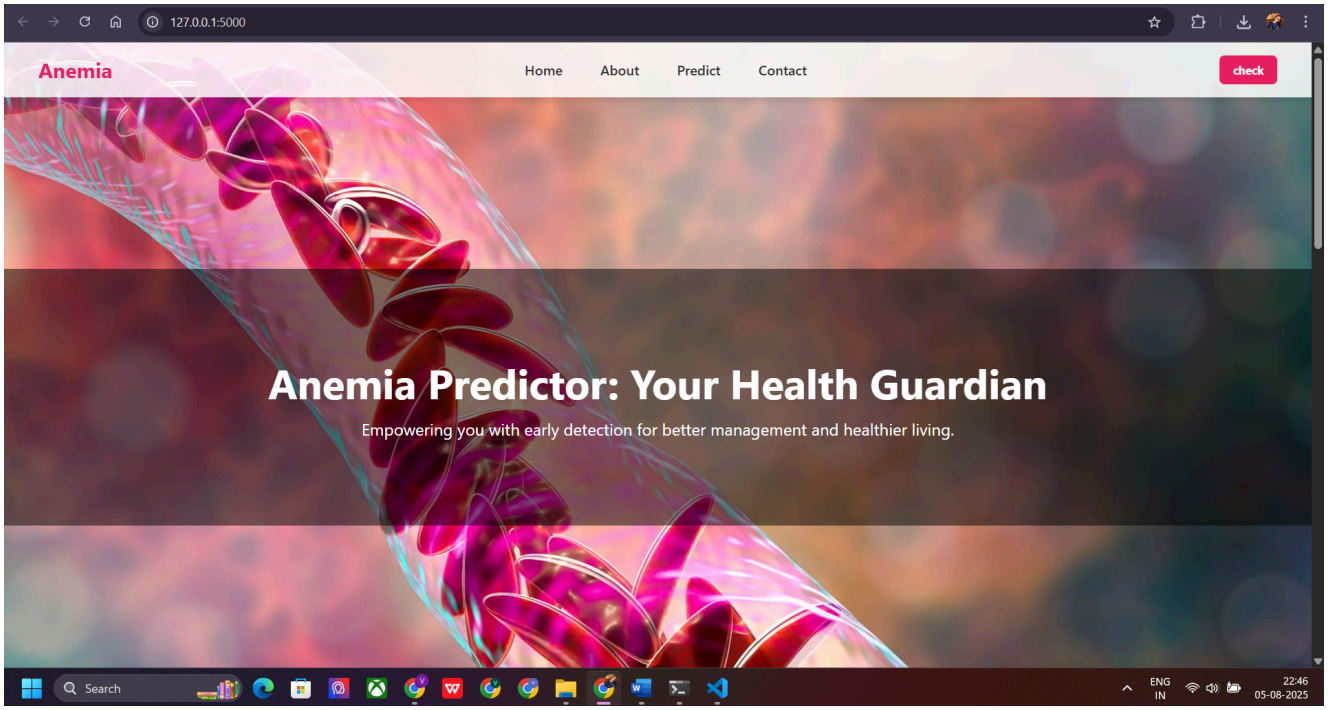
- The result of the prediction (Anemic or Not Anemic) is displayed to the user in a simple and clear message format.
- The system is designed to provide near-instant results based on real-time inputs.

6. Implementation Snapshot

The development of **AnemiaSense** was implemented in multiple phases, including data preprocessing, model training, Flask integration, and front-end design. Below are the snapshots demonstrating various stages of implementation:

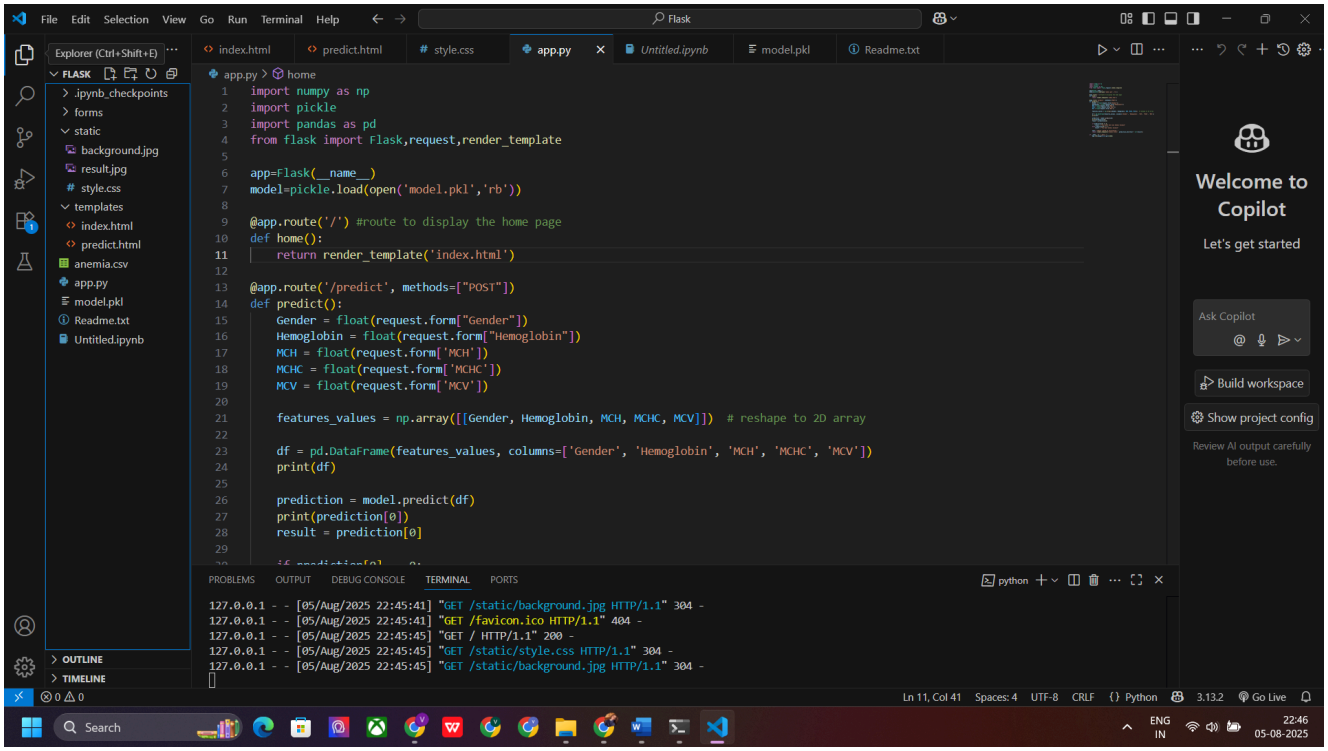
1. Home Page Interface

- A simple web form built using **HTML** and **CSS**.
- Allows users to input details like gender, hemoglobin level, age, etc.
- A "Predict" button submits the form data to the backend.



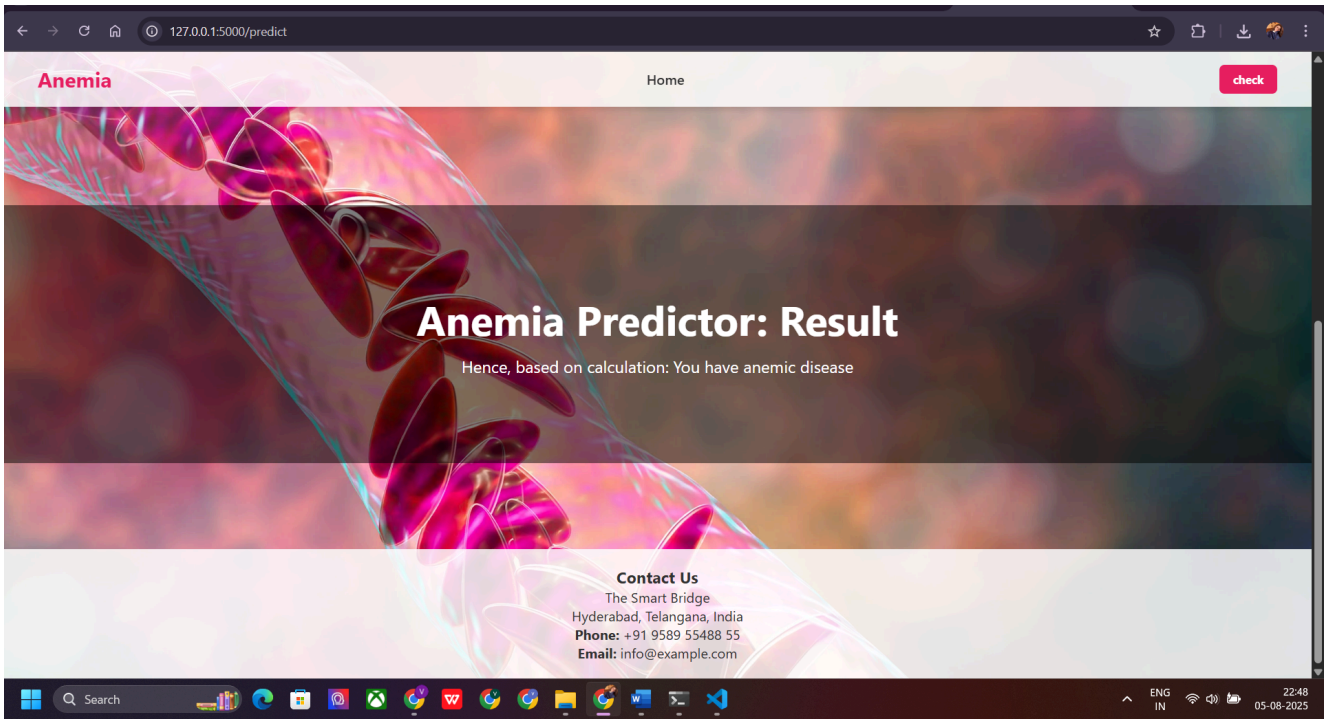
2. Flask Backend Code

- Python code handles the routing (@app.route) and input data collection.
- Data is passed to the trained ML model for real-time predictions.



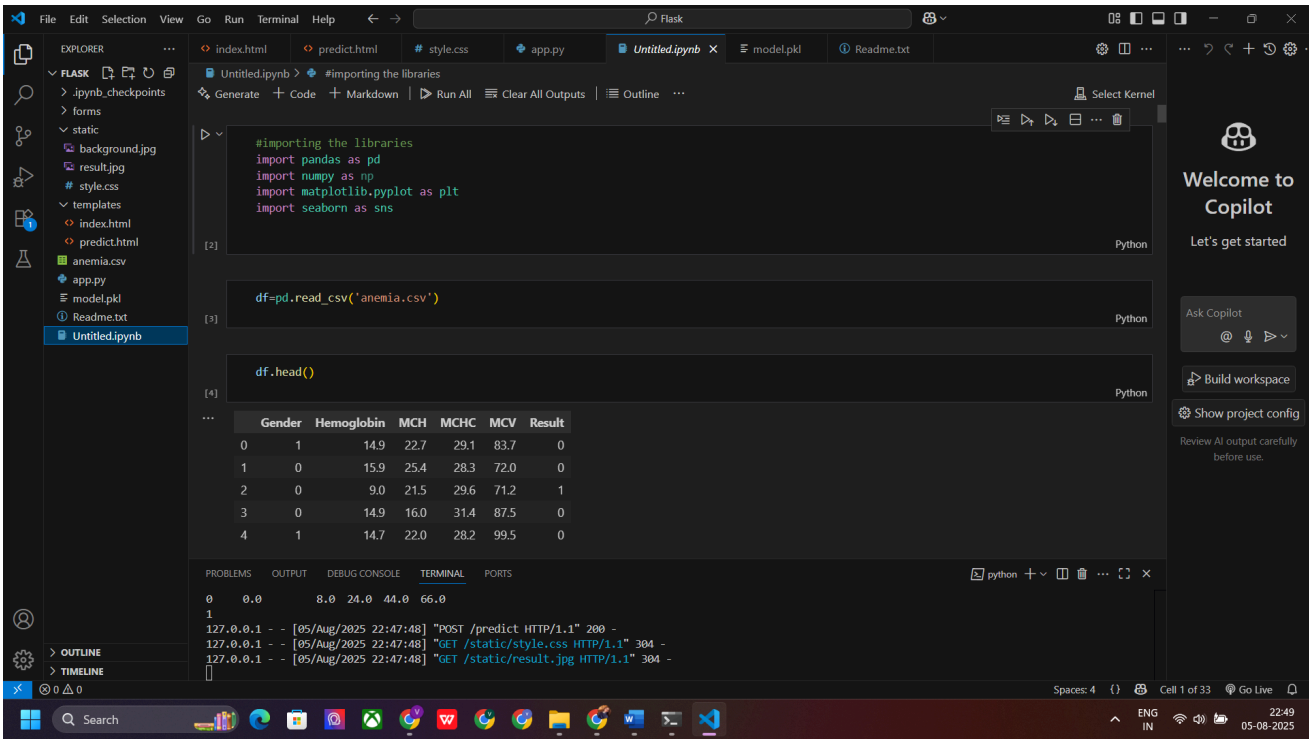
3. Model Prediction Output

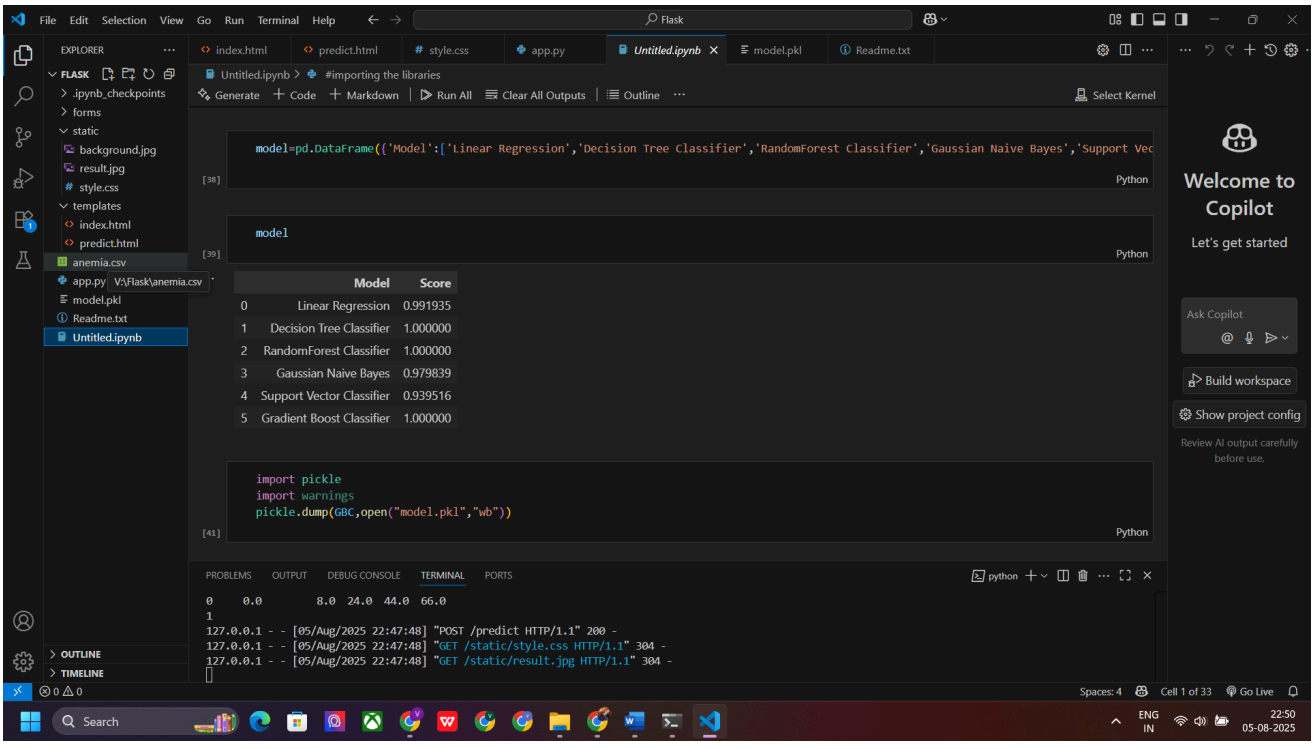
- The output is displayed on the same page or redirected to a result page.
- Shows whether the user is predicted to be "Anemic" or "Not Anemic."



4. Machine Learning Code

- Code for loading the dataset, training the model, and saving it using **Pickle**.
- Used **Scikit-learn** libraries for preprocessing, model building, and evaluation.





7. Results

The AnemiaSense project successfully demonstrates how machine learning can be used to predict anemia based on basic clinical data. After training and testing the model on a real-world dataset from Kaggle, the following results were observed:

Model Performance Metrics:

- **Algorithm Used:** Gradient Boosting Classifier

V:\Flask\static\background.jpg

```
from sklearn.ensemble import GradientBoostingClassifier

GBC=GradientBoostingClassifier()
GBC.fit(X_train,y_train)
y_pred=GBC.predict(X_test)

acc_gbc = accuracy_score(y_test,y_pred)
c_gbc=classification_report(y_test,y_pred)

print('Accuracy Score:' ,acc_gbc)
print(c_gbc)
```

[34]

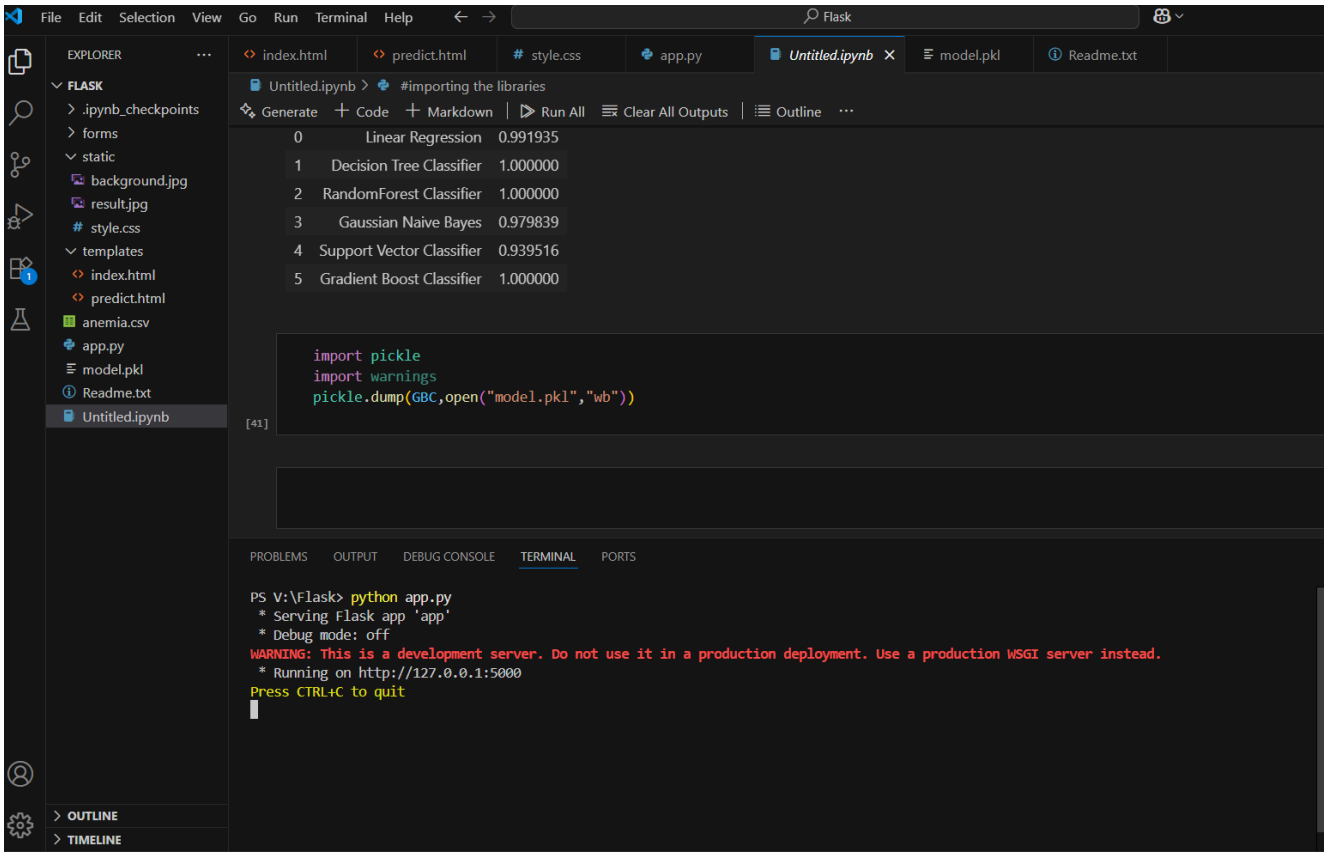
```
Accuracy Score: 1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       113
           1       1.00      1.00      1.00       135

    accuracy                           1.00       248
   macro avg       1.00      1.00      1.00       248
weighted avg       1.00      1.00      1.00       248
```

```
prediction=GBC.predict([[0,11.6,22.3,30.9,74.5]])
```

[35]

Model Outcome:

- The model was able to correctly classify individuals as **Anemic** or **Not Anemic** with high accuracy.
- It showed robustness in generalizing over unseen data during testing.
- Minimal overfitting observed, thanks to appropriate preprocessing and cross-validation.



Deployment Result:

- The model was integrated into a web-based interface using Flask.
- Users can enter parameters via a form and get predictions in real-time.
- The web interface responds quickly and reliably, even for low-end systems.

8. Conclusion

The **AnemiaSense** project demonstrates the practical application of machine learning and web development in solving real-world healthcare problems. By using a dataset containing basic hematological features, the system effectively predicts whether a person is anemic or not with high accuracy.

The project combined **data science**, **model building**, and **Flask-based deployment** to create a user-friendly diagnostic tool. Through proper data preprocessing, model evaluation, and UI integration, the system provides real-time results that can assist in early diagnosis and awareness of anemia.

This project not only reinforced core concepts of machine learning and web technologies but also highlighted how artificial intelligence can be used to support medical diagnostics—especially in resource-constrained environments where immediate lab testing is not always feasible.

AnemiaSense stands as a proof-of-concept for future development of intelligent healthcare systems that can be deployed at scale, integrated with hospital databases, or extended with additional medical parameters for even more accurate diagnostics.

9. Future Scope

While **AnemiaSense** is a functional and reliable AI-based system for anemia prediction, there is significant potential for future development and enhancement:

1. Enhanced Dataset

- Incorporating a **larger and more diverse dataset** from multiple sources (e.g., hospitals, medical institutions) could improve model accuracy and generalizability.

2. Additional Features

- Inclusion of more clinical features like **iron levels**, **vitamin B12**, **serum ferritin**, or **dietary habits** could help in multi-factorial diagnosis.

3. Real-time Integration

- The system can be integrated with **real-time health monitoring systems** or **hospital management software** for automatic risk detection.

4. Advanced Algorithms

- Future versions could utilize **deep learning** techniques like neural networks or ensemble models for better prediction performance.

5. Mobile App Deployment

- The system can be extended to Android/iOS platforms for **remote access** and **offline prediction capabilities** in rural and low-resource areas.

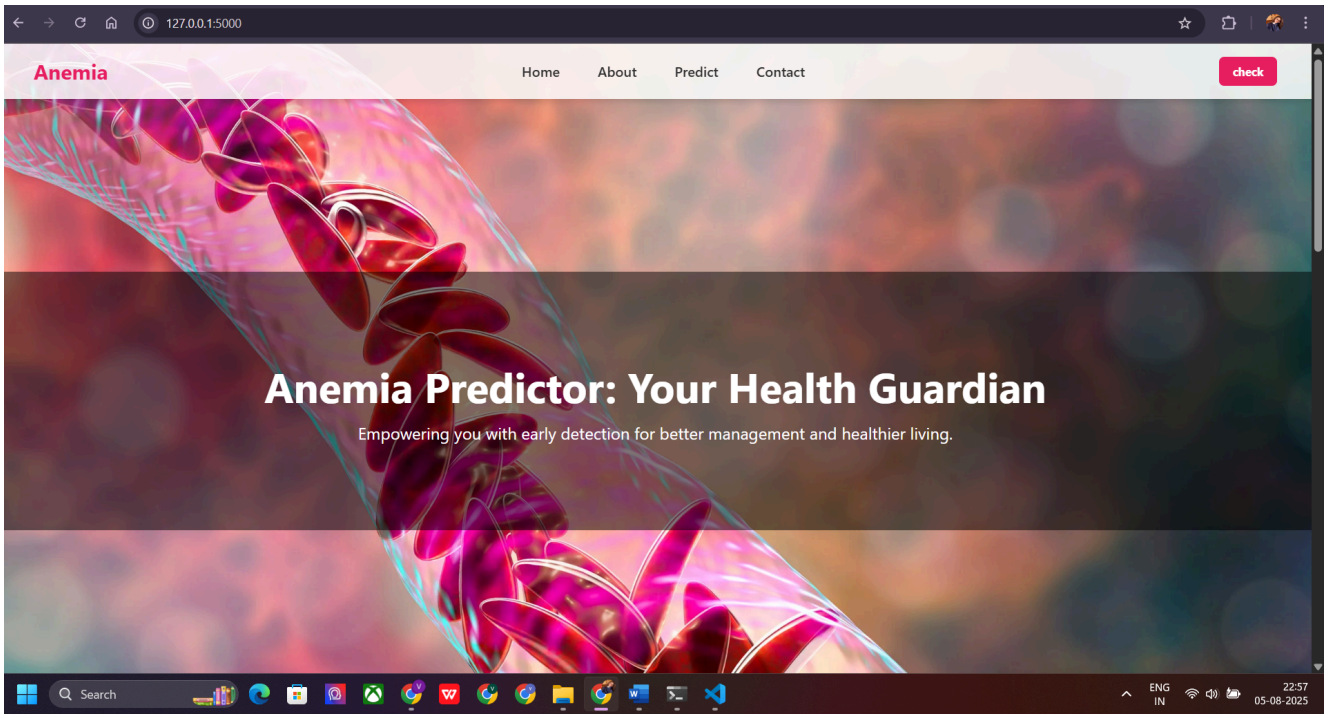
6. Data Privacy & Compliance

- Adding **user authentication**, **data encryption**, and **compliance with healthcare regulations** (like HIPAA) can make the system production-ready.

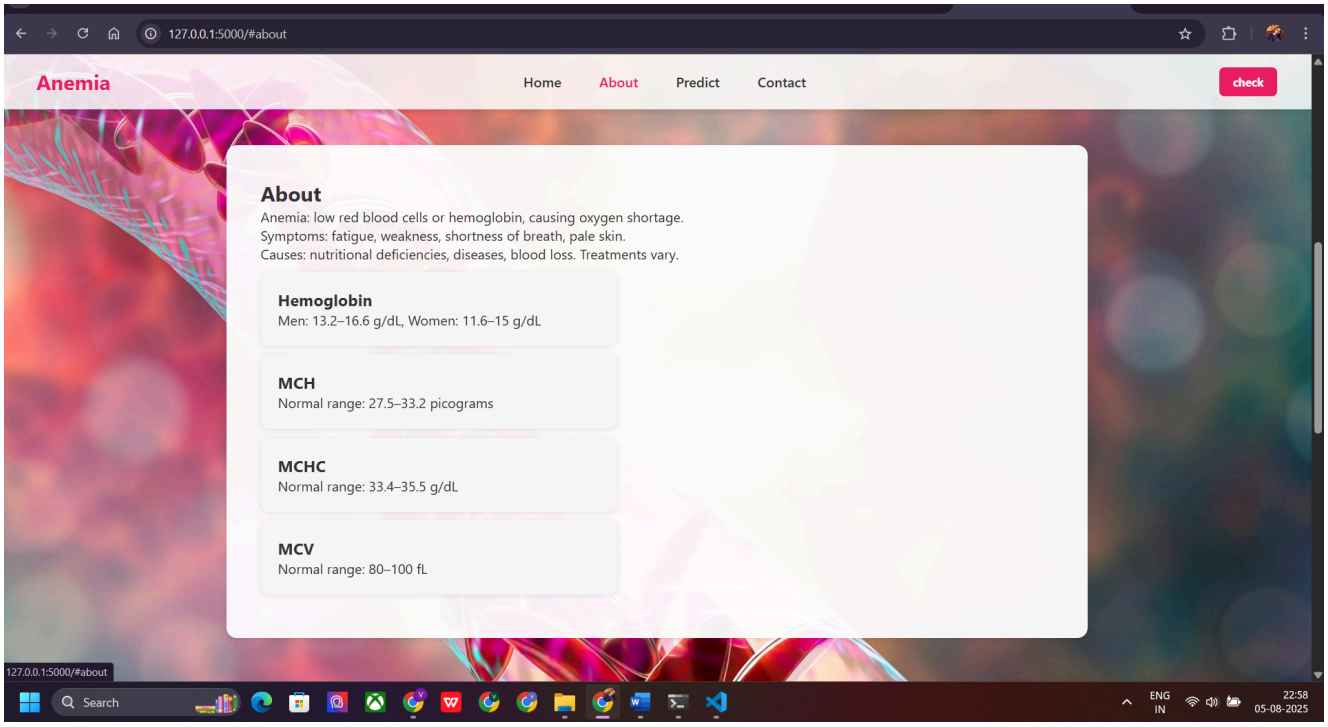
These enhancements would not only improve the diagnostic capabilities of **AnemiaSense** but also make it suitable for deployment in **clinical, rural, and telemedicine** settings, thus broadening its societal impact.

10. Screenshots

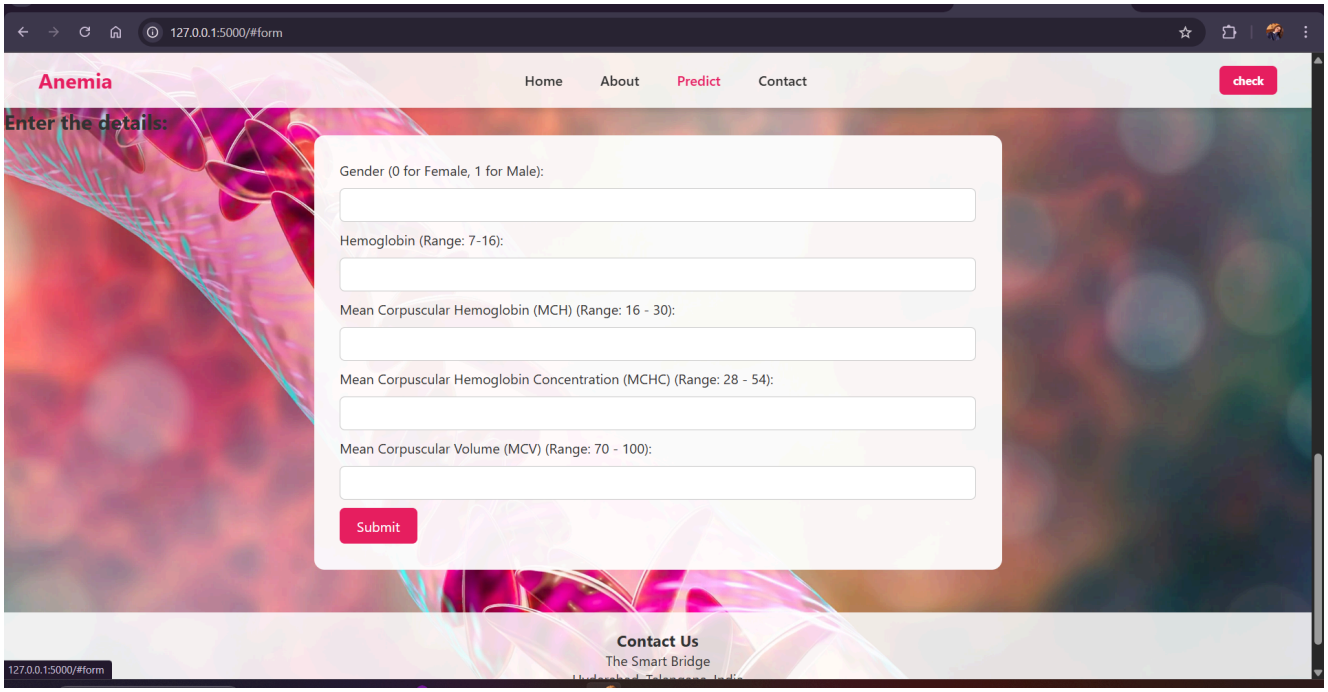
HOME



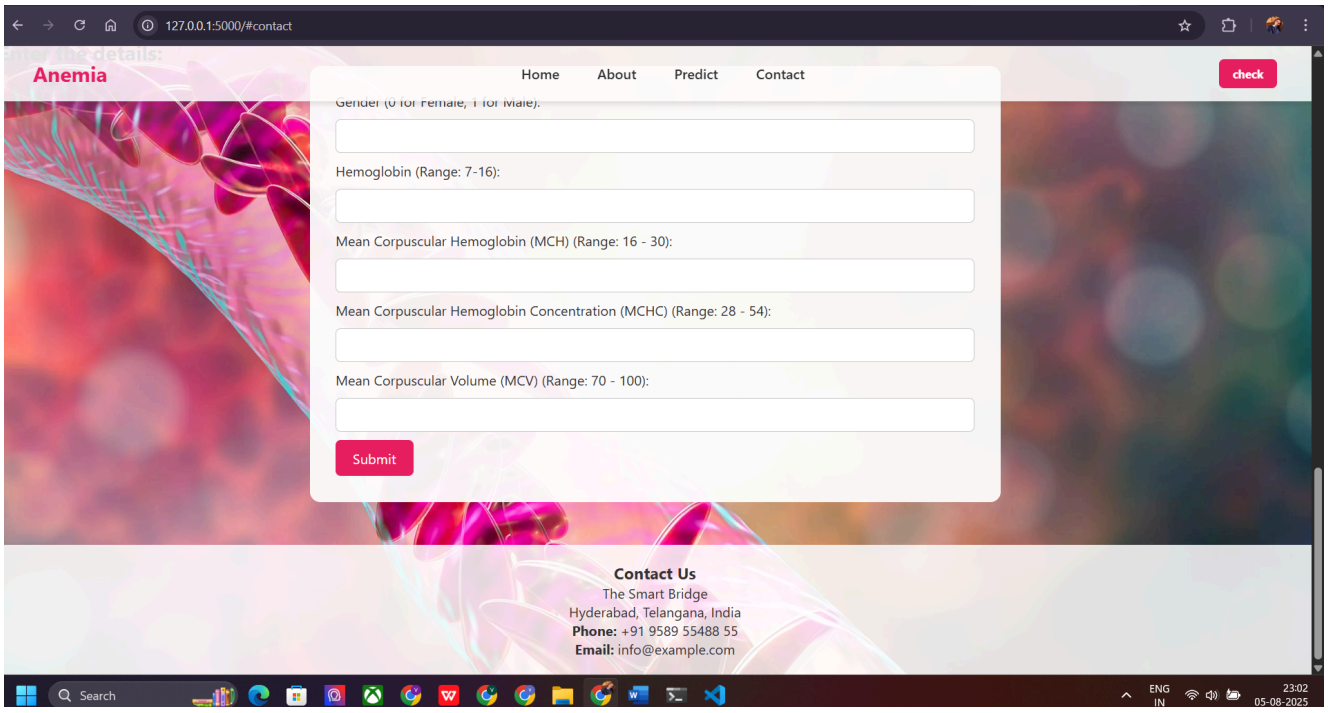
ABOUT



PREDICT



CONTACT



11. References

1. **Kaggle – Anemia Dataset**
   Biswaranjan Rao.
   URL: https://www.kaggle.com/datasets/biswaranjanrao/anemia-dataset
2. **Scikit-learn Documentation**
   Machine Learning in Python.
   URL: https://scikit-learn.org/stable/
3. **Flask Documentation**
   The Python Microframework for Web Development.
   URL: https://flask.palletsprojects.com/
4. **Python Official Documentation**
   URL: https://docs.python.org/3/
5. **Smart Internz Internship Resources**
   Materials and guidance provided during the internship program.
6. **Online Tutorials and Blogs**
   Various online platforms like GeeksforGeeks, Stack Overflow, and Medium for troubleshooting and learning best practices.