Model Development Phase Template

Date	16th June 2025
Team ID	SWTID1749621188
Project Title	Anemia Sense Leveraging-Machine Learning For-Precise Anemia Recognition
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:

```
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)
joblib.dump(scaler, "scaler.pkl")
```

```
# Logistic Regression
lr = LogisticRegression(max_iter=1000)
lr.fit(X_train_scaled, y_train)
evaluate_model("Logistic Regression", lr, X_test_scaled, y_test, model_accuracies)
```

```
# Random Forest
rf = RandomForestClassifier()
rf.fit(X_train_scaled, y_train)
evaluate_model("Random Forest", rf, X_test_scaled, y_test, model_accuracies)
```

```
# Decision Tree
dt = DecisionTreeClassifier()
dt.fit(X_train_scaled, y_train)
evaluate_model("Decision Tree", dt, X_test_scaled, y_test, model_accuracies)
```

```
# Naive Bayes
nb = GaussianNB()
nb.fit(X_train_scaled, y_train)
evaluate_model("Naive Bayes", nb, X_test_scaled, y_test, model_accuracies)
```

```
# Support Vector Machine
svm = SVC(probability=True)
svm.fit(X_train_scaled, y_train)
evaluate_model("SVM", svm, X_test_scaled, y_test, model_accuracies)
```

```
# Gradient Boosting
gb = GradientBoostingClassifier()
gb.fit(X_train_scaled, y_train)
evaluate_model("Gradient Boosting", gb, X_test_scaled, y_test, model_accuracies)
```

```
# Lasso (L1) Logistic Regression
lasso = LogisticRegression(penalty='l1', solver='liblinear', max_iter=1000)
lasso.fit(X_train_scaled, y_train)
evaluate_model("Lasso (L1)", lasso, X_test_scaled, y_test, model_accuracies)
```

```
def evaluate_model(name, model, X_test, y_test, model_accuracies):
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    model_accuracies[name] = acc
    print(f"\n--- {name} ---", flush=True)
    print(f"Accuracy: {acc * 100:.2f}%", flush=True)
    print("Confusion Matrix:")
    print(confusion_matrix(y_test, y_pred), flush=True)
    print("\n")
    print("Classification Report:")
    print(classification_report(y_test, y_pred), flush=True)
```

Model Validation and Evaluation Report:

Model	Classification Report	F1 Score	Confusion Matrix
Logistic Regression	Classification Report: precision recall f1-score support 0 1.00 0.98 0.99 157 1 0.98 1.00 0.99 128 accuracy 0.99 285 macro avg 0.99 0.99 0.99 285 weighted avg 0.99 0.99 0.99 285	98.95 %	Confusion Matrix: [[154 3] [0 128]]
Random Forest	Classification Report: precision recall f1-score support 0 1.00 1.00 1.00 157 1 1.00 1.00 1.00 128 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	100.0 %	Confusion Matrix: [[157 0] [0 128]]
Decision Tree	Classification Report: precision recall f1-score support 0 1.00 1.00 1.00 157 1 1.00 1.00 1.00 128 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	100.0 %	Confusion Matrix: [[157 0] [0 128]]
Naive Bayes	Classification Report: precision recall f1-score support 0 0.96 0.96 0.96 157 1 0.95 0.95 0.95 128 accuracy 0.95 285 macro avg 0.95 0.95 0.95 285 weighted avg 0.95 0.95 0.95 285	95.09%	Confusion Matrix: [[150 7] [7 121]]
SVM	Classification Report: precision recall f1-score support 0 1.00 0.97 0.98 157 1 0.96 1.00 0.98 128 accuracy 0.98 285 macro avg 0.98 0.98 0.98 285 weighted avg 0.98 0.98 0.98 285	98.25%	Confusion Matrix: [[152 5] [0 128]]

Gradient Boosting	Classification Report: precision recall f1-score support 0 1.00 1.00 1.00 157 1 1.00 1.00 1.00 128 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	100.0 %	Confusion Matrix: [[157 0] [0 128]]
Lasso (L1)	Classification Report: precision recall f1-score support 0 1.00 0.99 0.99 157 1 0.98 1.00 0.99 128 accuracy 0.99 285 macro avg 0.99 0.99 0.99 285 weighted avg 0.99 0.99 0.99 285	99.3%	Confusion Matrix: [[155 2] [0 128]]