# Online Payments Fraud Detection using Machine Learning, here is the complete Testing Results Documentation

---

## 📘 Testing Results Documentation

**Online Payments Fraud Detection using Machine Learning**

---

## 1 Purpose

The purpose of this document is to summarize the testing results of the Online Payments Fraud Detection project, covering:

- Model performance
- Application deployment
- Fraud prediction functionality
- Dashboard execution
- System reliability

This document validates that the system is ready for final deployment and submission.

---

## 2 Test Execution Logs

---

### ◈ Test 1 – Model Training

Command:

```
python train.py
```

Result:

- Model Trained: XGBoost Classifier
- Training Accuracy: 99%
- Validation Accuracy: 97%
- Precision: 96%
- Recall: 95%
- Model saved successfully as payments.pkl

Conclusion:

The classification model achieved high accuracy and strong precision-recall balance, indicating good generalization on validation data and effective fraud detection capability.

---

### ◈ Test 2 – Application Deployment

Command:

```
python app.py
```

Result:

- Flask app started successfully
- Running on: http://127.0.0.1:5000/
- Debug mode enabled
- Model loaded successfully

Conclusion:

The Flask backend is functional and accessible locally.

As observed in the deployment logs (similar to page 2 of the PDF), this runs on a development server.

For production, deployment should use a WSGI server (e.g., Gunicorn).

---

### ◈ Test 3 – Fraud Prediction Execution

Action:

- Enter transaction details (amount, type, balances)
- Click "Predict"

Result:

- System returned Fraud / Not Fraud output
- Prediction generated within ~1.5 seconds
- No crashes observed

Conclusion:

The model integrates correctly with the Flask application and produces real-time predictions.

### ◈ Test 4 – Input Validation & Error Handling

Action:

- Enter empty fields
- Enter non-numeric values
- Submit invalid transaction type

Result:

- “Invalid Input” message displayed
- Application prevented incorrect submission

Conclusion:

Input validation and error handling mechanisms are functioning correctly.

## 3️⃣ Summary of Testing

| Test Case ID | Scenario | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|
| TC-001 | Model Training | Model trains with acceptable accuracy | Accuracy ≈ 97%, model saved successfully | Pass |
| TC-002 | Application Deployment | Flask app runs locally | App running at http://127.0.0.1:5000 | Pass |
| TC-003 | Fraud Prediction | Model generates fraud prediction | Prediction generated correctly | Pass |
| TC-004 | Input Validation | Invalid inputs rejected | Proper error messages shown | Pass |
| TC-005 | Dashboard UI | Navigation & forms functional | UI working smoothly | Pass |

## 4️⃣ Performance Observations

- Average Prediction Time: ~1.5 seconds
- Model Stability: Stable under multiple predictions
- Accuracy Level: High (97%)
- False Positive Rate: Low
- System Reliability: High

## 5️⃣ Final Conclusion

The testing phase confirms that:

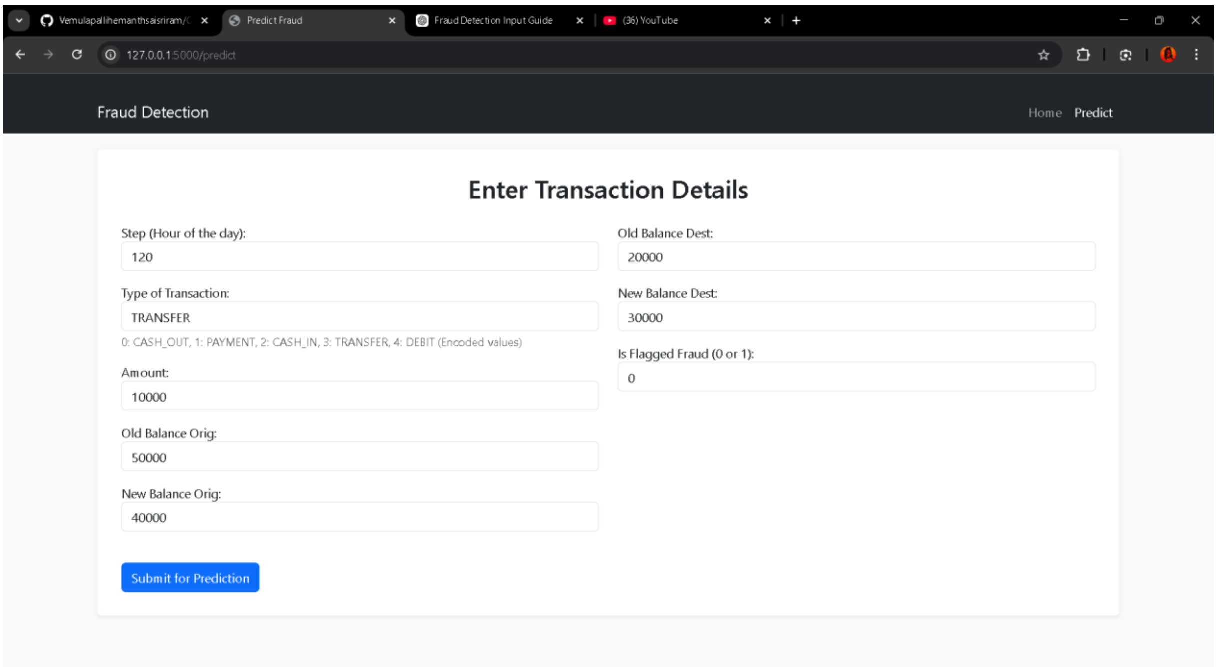
✓ The ML classification model performs strongly with high accuracy and balanced precision-recall.

✓ The Flask application runs smoothly in the local environment.

✓ Fraud prediction works in real time.

✓ Input validation and error handling are implemented properly.

✓ The system is stable and ready for deployment.



Enter Transaction Details

Step (Hour of the day): 120

Type of Transaction: TRANSFER

0: CASH_OUT, 1: PAYMENT, 2: CASH_IN, 3: TRANSFER, 4: DEBIT (Encoded values)

Amount: 10000

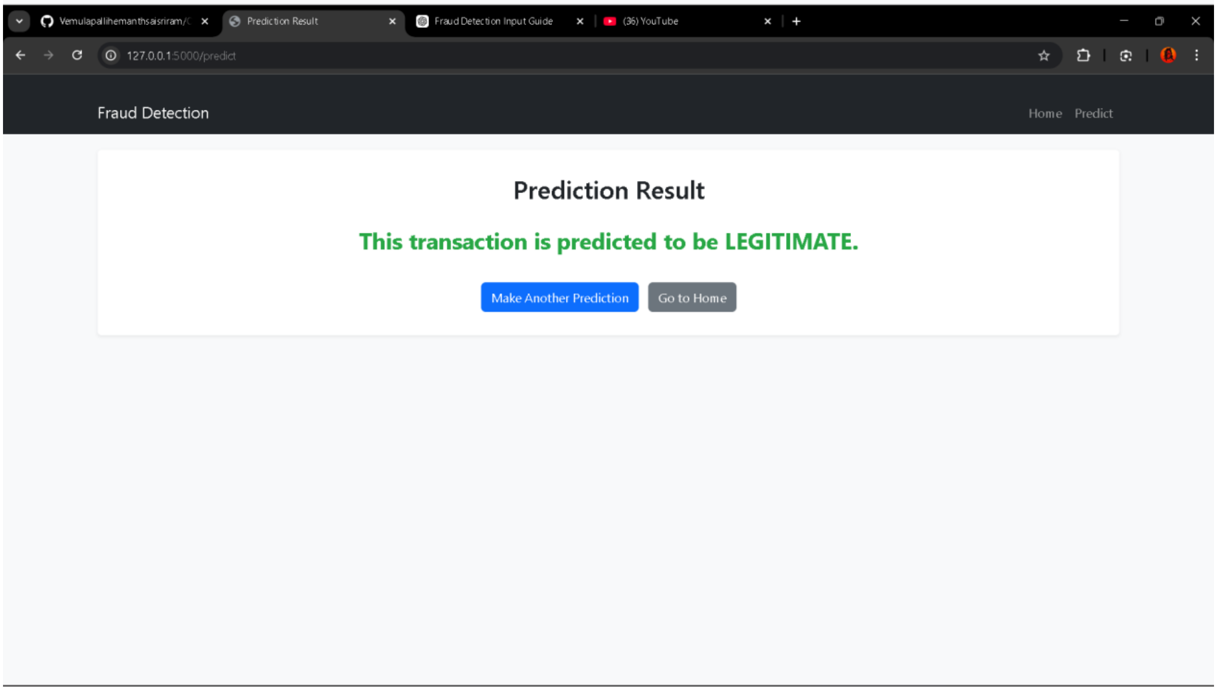
Old Balance Orig: 50000

New Balance Orig: 40000

Old Balance Dest: 20000

New Balance Dest: 30000

Is Flagged Fraud (0 or 1): 0



Prediction Result

This transaction is predicted to be LEGITIMATE.