Team Adroit : Nawaz Sayyad
Project Report: Z+ Guard

1. **Introduction** This project aims to develop an intelligent fraud detection and transaction monitoring system that ensures secure and reliable financial transactions. The system effectively monitors account activities, identifies fraudulent patterns, and takes necessary actions to prevent unauthorized transactions.

2. **Objectives**

- To create a robust account transaction database for secure financial monitoring.
- To simulate transactions with different scenarios.
- To detect account status (Active, Suspended, or Blocked).
- To implement machine learning-based behavior detection for anomaly identification.
- To analyze transaction patterns and detect fraudulent activities.
- To provide real-time feedback and response mechanisms for fraud prevention.
- To automatically generate a summary of detected fraud incidents and save it as a PDF for reporting.
- To achieve a cost-effective solution using open-source technologies.

3. **System Implementation**

3.1 Account Status Monitoring

- The system classifies accounts into three states:
  - **Active:** The account is fully operational. ○ **Suspended:** The account has limited functionality.
  - **Blocked:** The account is completely restricted due to suspicious activity.
  - **Suspended:** The accounts Access is Temporarily Stopped due to suspicious activity.

- A database structure is implemented to store account status and track changes dynamically.

3.2 Transaction Simulation & Database Management

- A comprehensive database schema is designed, including:
  - **User Accounts Table** ○ **Transaction Logs Table**
  - **Fraud Detection Flags Table**
- Transactions are simulated to test different use cases and edge scenarios.

3.3 Behavior-Based Detection System

- A machine learning model is trained to analyze transaction behaviors.
- If a transaction behavior is classified as "unbelievable," the system flags it for further inspection.

3.4 Pattern-Based Fraud Detection

- Server-side transaction logs are analyzed for suspicious patterns.
- If fraudulent patterns are detected, the system takes immediate action:
  - Blocks suspicious transactions. ○ Notifies the user of potential fraud.
  - Suggests security measures to prevent future fraud attempts. ○ **Automatically generates a summary report of the incident using AI-based intelligence and saves it as a PDF for further analysis.**
  - **Sends the generated PDF report to the cybersecurity team for review and action.**

4. Key Features

- **Real-time Monitoring:** Instant analysis and fraud detection.
- **Automated Account Management:** Blocking, suspending, or activating accounts based on user behavior.
- **Automated Report Generation:** Summarizes fraud incidents and creates a PDF report for cybersecurity teams.
- **Cost-Effective Solution:** Fully implemented using open-source technologies with zero additional costs.
- **Scalability:** Designed to handle a high volume of transactions efficiently.
- Frontend (if applicable): React/Angular

6. Conclusion

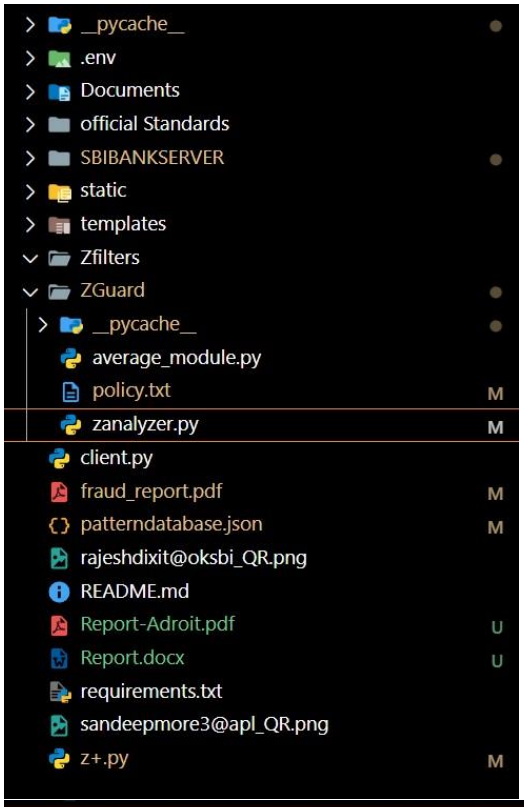
This project successfully delivers a secure and intelligent fraud detection system with real-time monitoring and preventive measures. By leveraging machine learning for behavior analysis and pattern detection, it enhances security in financial transactions. Additionally, the open-source approach ensures a cost-effective and scalable solution for organizations seeking fraud prevention mechanisms.

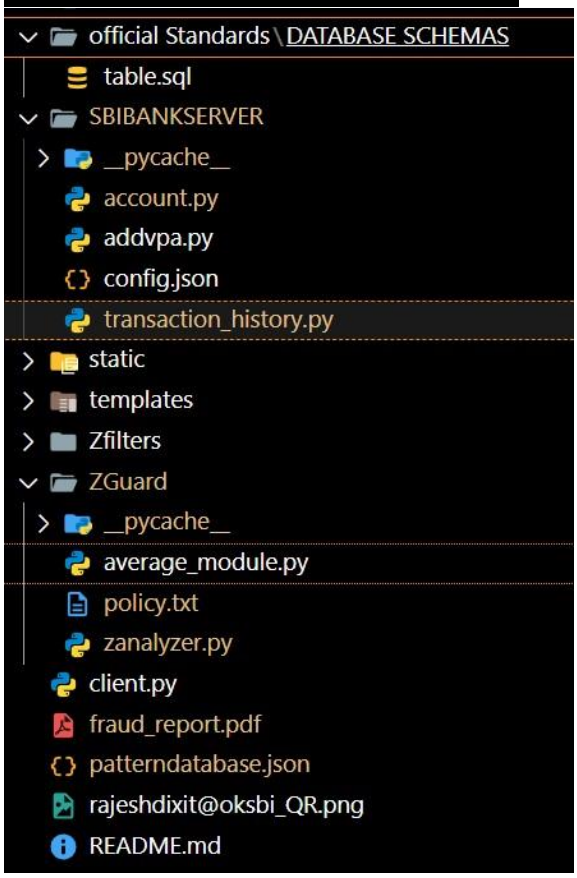
7. Future Enhancements

- Integration with blockchain for enhanced security.
- Implementing advanced deep learning techniques for fraud detection.
- Real-time user authentication mechanisms using biometrics.

Modules :

Here z+ is the Main file from which DB are Manipulated when Fraud is Detected





average_module is ML model to predict Unusual Behaviour

z+.py M | zanalyzer.py M | policy.txt M | fraud_report.pdf M

uard > average_module.py > check_transaction

```python
import numpy as np
import pandas as pd
from sklearn.ensemble import IsolationForest
import json
from datetime import datetime

def preprocess_text(text):
    return " ".join(text.lower().split())

def time_to_minutes(timestamp):
    dt = datetime.strptime(timestamp, "%Y-%m-%d %H:%M:%S")
    return dt.hour * 60 + dt.minute

def prepare_dataset(transactions):
    df = pd.DataFrame(transactions)
    df["time_in_minutes"] = df["date_time_stamp"].apply(time_to_minutes)
    df["transaction_note"] = df["transaction_note"].fillna("Unknown").apply(preprocess_text)
    return df

def train_model(df):
    X_train = df[["transaction_amount", "time_in_minutes"]].values
    model = IsolationForest(contamination=0.2, random_state=42)
    model.fit(X_train)
    return model

def check_transaction(new_tx, df, model):
    reasons = []
    new_tx_time = time_to_minutes(new_tx["date_time_stamp"])
    is_new_location = new_tx["payer_location_zip"] not in df["payer_location_zip"].values
    is_new_upi = new_tx["receiver_vpa"] not in df["receiver_vpa"].values
    new_tx_vector = np.array([[new_tx["transaction_amount"], new_tx_time]])
    is_amount_anomaly = model.predict(new_tx_vector)[0] == -1

    if is_amount_anomaly:
        reasons.append("Due to high amount, this is fraud.")
    if is_new_location and new_tx["transaction_amount"] > df["transaction_amount"].mean() * 1.5:
        reasons.append("New location and significantly high amount, potential fraud.")
    if is_new_upi and new_tx["transaction_amount"] > df["transaction_amount"].mean() * 2:
        reasons.append("New UPI ID with an unusually high transaction amount, possible fraud.")

    is_fraud = bool(reasons)
    return {
        "is_fraudulent": is_fraud,
        "reasons": reasons
    }

def is_the_user_normal(transactions, new_transaction):
    df = prepare_dataset(transactions[:100])  # Use top 100 recent transactions
    model = train_model(df)
    result = check_transaction(new_transaction, df, model)
    return json.dumps(result, indent=4)
```

AI model To generate Report and Email

```python
def send_to_z_analyze(patterndb,TransactionHistory,currenttrsanaction):
    print("Request received")

    # Construct the prompt
    prompt = f"""
    PatternDB : {patterndb} ;;;;;
    User Previous Transaction History : {TransactionHistory} ;;;;
    currenttrsanaction : {currenttrsanaction};;;

    JUST RETURN THE JSON - NOT ANYTHING ELSE - JUST JSON
    """
    # Call the Ollama model
    response = ollama.chat(
        model="zGuard",
        messages=[
            {"role": "user", "content": prompt},  # Pass the prompt
        ],
    )

    # Return the response from the model
    return response.get("message", {}).get("content", "No message found")
```

Main Z+ code :

```python
# for that we need the last 100 transactions
try:
    print("hiiiiiiiiiiiiiiiiiiiiiii")

    import json
    f100rows = tdb.get_transactions_by_vpa((transaction.payer.vpa))  # ✅ Directly use the list
    exact_timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print("her Transactions "+str(f100rows))
    latestQ = {
    "receiver_vpa":transaction.payee.vpa,
    "transaction_amount":transaction.amount,
    "date_time_stamp": exact_timestamp,
    "payer_location_zip":transaction.payer.zipcode,
    "transaction_note": transaction.description
    }
    result  =  average_module.is_the_user_normal(f100rows,latestQ)
    print("The type resukt is "+str(type(result)))
    result = json.loads(result)
    print((result))

    is_fraudulent = result["is_fraudulent"]  # Correct way to access
    reasons = result["reasons"]

    print("Fraud Or not ?"+str(is_fraudulent))
    if is_fraudulent == True:
        with open("patterndatabase.json", "r") as file:
            patterndb = str(json.load(file))

    #   now here is the one of the important and last Step  to analayze the pattterns at Human level
        fffresult = (zanalyzer.send_to_z_analyze(patterndb,tdb.get_transactions_by_vpa_combined(transaction.payer.vpa),str(latestQ)
        print(fffresult)
        append_fraud_summary_to_pdf(fffresult,transaction)
    #   blocking Fruadlant account
    # and suspending the User

        adb.update_account_status(transaction.payer.vpa,"Suspended")
        adb.update_account_status(transaction.payee.vpa,"Blocked")

    #   updating the logs
    # ENUM(
    #                     'SUCCESSFUL',
    #                     'INSUFFICIENT_BALANCE',
    #                     'NETWORK_DOWN',
    #                     'BLOCKED_ACCOUNT',
    #                     'SUSPENDED',
    #                     'LIMIT_CROSSED',
    #                     'FRAUD_DETECTED_STOPPED'
    #                   )
        tdb.insert_transaction(
            payer_account_no=payer_accountno,
            payer_vpa=transaction.payer.vpa,
            receiver_vpa=transaction.payee.vpa,
            transaction_amount=transaction.amount,
            payer_location_zip=transaction.payer.zipcode,
            payer_city=transaction.locationDetails.city,
            payer_state=transaction.locationDetails.state,
            ip_address=transaction.payer.ipAddress,
            transaction_note=transaction.description,
            device=transaction.payer.deviceInfo.deviceType,
            mode="UPI",
            status="FRAUD_DETECTED_STOPPED"
        )
```

Databasses : Account

```
    `BANK_BALANCE` DECIMAL(20,2) NOT NULL,  -- Precise financial storage
    `BRANCH_NO` BIGINT NOT NULL,
    `ACCOUNT_STATUS` VARCHAR(20) NOT NULL,  -- Using VARCHAR instead of ENUM for flexibility
    `ADDRESS` VARCHAR(500) NOT NULL,  -- Using VARCHAR instead of TEXT for better performance
    `FIRST_NAME` VARCHAR(100) NOT NULL,
    `MIDDLE_NAME` VARCHAR(100),
    `LAST_NAME` VARCHAR(100) NOT NULL,
    `PHONE_NUMBERS` VARCHAR(20) NOT NULL
) AUTO_INCREMENT = 525555;  -- Set the starting value at the time of table creation
desc account;
select * from account;
```

ACCOUNT_NO	BANK_BALANCE	BRANCH_NO	ACCOUNT_STATUS	ADDRESS	FIRST_NAME	MIDDLE_NAME	LAST_NAME	PHONE_NUMBERS	VPA
525555	14.48	101	Active	Mumbai, Maharashtra	Amit	Kumar	Sharma	+919422019958	amitsharma@ybl
525556	11216546.88	102	Suspended	Pune, Maharashtra	Priya	Ramesh	Joshi	+919833445566	priyajoshi@okicici
525557	5465.48	103	Blocked	Bengaluru, Karnataka	Raj	Anil	Verma	+919766554433	rajverma@okicici
525558	11216546.88	104	Active	Delhi	Anjali	Suresh	Kapoor	+919700112233	anjalikapoor@apl
525559	5465.48	105	Active	Chennai, Tamil Nadu	Vikas	Mahesh	Iyer	+919844556677	vikasiyer@oksbi
525560	11216546.88	106	Suspended	Hyderabad, Telangana	Neha	Ravi	Reddy	+919922334455	nehareddy@paytm
525561	5465.48	107	Blocked	Kolkata, West Bengal	Rahul	Dev	Chowdhury	+919855667788	rahulchowdhury@sbi
525562	11216546.88	108	Active	Ahmedabad, Gujarat	Sneha	Rajesh	Patel	+919900778899	snehapatel@apl
525563	5465.48	109	Active	Jaipur, Rajasthan	Manish	Sanjay	Singh	+919755889900	manishsingh@okicici
525564	11216546.88	110	Suspended	Lucknow, Uttar Pradesh	Divya	Amit	Yadav	+919644221133	divyayadav@okicici
525565	5465.48	111	Blocked	Chandigarh	Karan	Deep	Singh	+919811223344	karansingh@sbi
525566	11211542.88	112	Active	Bhopal, Madhya Pradesh	Pooja	Satish	Shukla	+919922334466	poojashukla@ybl
525567	5465.48	113	Active	Nagpur, Maharashtra	Suresh	Mohan	Deshmukh	+919833221100	sureshdeshmukh@sbi
525568	11216546.88	114	Suspended	Indore, Madhya Pradesh	Rekha	Vijay	Sharma	+919955667788	rekhasharma@oksbi
525569	5465.48	115	Blocked	Patna, Bihar	Vinay	Anil	Thakur	+919899332211	vinaythakur@ybl
525570	11216546.88	116	Active	Guwahati, Assam	Ritika	Ramesh	Das	+919977665544	ritikadas@okicici
525571	5465.48	117	Active	Bhubaneswar, Odisha	Aakash	Rajiv	Mishra	+919944556677	aakashmishra@oksbi
525572	11216546.88	118	Suspended	Dehradun, Uttarakhand	Simran	Mukesh	Rawat	+919811009988	simranrawat@sbi
525573	5465.48	119	Blocked	Shimla, Himachal Pradesh	Nikhil	Pratap	Negi	+918899776655	nikhilnegi@paytm
525574	11216546.88	120	Active	Panaji, Goa	Meera	Kishor	Naik	+919911223344	meeranaik@apl

#	Time	Action	Message
1	10:27:15	select * from account LIMIT 0, 1000	Error Code: 1046. No database selected Select
2	10:27:19	use BANK	0 row(s) affected
3	10:27:22	select * from account LIMIT 0, 1000	1000 row(s) returned

Transaction DB :

```
                    'LIMIT_CROSSED',
                    'FRAUD_DETECTED_STOPPED'
                  ) NOT NULL,

    -- Foreign Key Constraint
```

transaction_id	payer_account_no	payer_vpa	receiver_vpa	transaction_amount	date_time_stamp	payer_location_zip	payer_city	payer_state	ip_address	transaction_note	device	mode	statu
52	525566	poojashukla@ybl	saurabhbhosale1@o...	54545522.00	2025-02-23 22:56:03	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	INSUF
53	525566	poojashukla@ybl	saurabhbhosale1@o...	54545522.00	2025-02-23 22:56:19	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	INSUF
54	525566	poojashukla@ybl	saurabhbhosale1@o...	4.00	2025-02-23 22:56:57	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
55	525683	ganeshbhosale@apl	nitinpatil@ybl	1000.00	2025-02-23 22:57:45	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
56	525684	nitinpatil@ybl	ganeshbhosale@apl	2000.00	2025-02-23 22:58:18	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
57	525683	ganeshbhosale@apl	nitinpatil@ybl	50000.00	2025-02-23 22:58:35	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
58	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:18	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
59	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:20	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
60	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:20	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
61	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:20	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
62	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:20	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
63	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:20	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
64	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:21	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
65	525684	nitinpatil@ybl	ganeshbhosale@apl	1000.00	2025-02-23 22:59:21	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
75	526288	sandeepmore3@apl	rajeshdixit@oksbi	100.00	2025-02-23 23:26:34	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
76	526269	rajeshdixit@oksbi	sandeepmore3@apl	2064.00	2025-02-23 23:27:02	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
77	526288	sandeepmore3@apl	rajeshdixit@oksbi	1200.00	2025-02-23 23:27:30	411005	Pune	Maharashtra	2401:4900:54ea:f2c8:c50b:c316:5d7e:6f34	offer	mobile	UPI	SUCCI
93	526269	rajeshdixit@oksbi	sandeepmore3@apl	800.00	2025-02-24 09:40:53	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
94	526269	rajeshdixit@oksbi	sandeepmore3@apl	850.00	2025-02-24 09:40:57	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
95	526269	rajeshdixit@oksbi	sandeepmore3@apl	800.50	2025-02-24 09:41:00	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
96	526269	rajeshdixit@oksbi	sandeepmore3@apl	800.00	2025-02-24 09:41:03	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
97	526269	rajeshdixit@oksbi	sandeepmore3@apl	480.00	2025-02-24 09:41:06	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
98	526269	rajeshdixit@oksbi	sandeepmore3@apl	500.00	2025-02-24 09:41:27	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
99	526269	rajeshdixit@oksbi	sandeepmore3@apl	550.00	2025-02-24 09:41:30	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
101	526288	sandeepmore3@apl	rajeshdixit@oksbi	100.00	2025-02-24 09:43:28	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
102	526269	rajeshdixit@oksbi	sandeepmore3@apl	200.00	2025-02-24 09:44:01	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner	mobile	UPI	SUCCI
103	526288	sandeepmore3@apl	rajeshdixit@oksbi	400.00	2025-02-24 09:48:01	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	Dinner Hotel Vish	mobile	UPI	SUCCI
109	526269	rajeshdixit@oksbi	sandeepmore3@apl	800.00	2025-02-24 10:00:24	400025	Mumbai	Maharashtra	2401:4900:1907:799b:992a:a791:8a9e:fea9	offer	mobile	UPI	SUCCI

#	Time	Action	Message
1	10:27:15	select * from account LIMIT 0, 1000	Error Code: 1046. No database selected Select the default DB to be used by double-clicking its name in the SCHEMAS list in the si
2	10:27:19	use BANK	0 row(s) affected
3	10:27:22	select * from account LIMIT 0, 1000	1000 row(s) returned
4	10:27:40	select * from Transaction_History LIMIT 0, 1000	79 row(s) returned

Payment System

SBI

Payment Gateway

SBI Life

Powered by Security of Z+ Guard

Developed By Adroit

Transaction processed successfully with status: SUCCESSFUL

Scan UPI ID

Open Scanner

Enter UPI ID or VPA

rajeshdixit@oksbi

Enter UPI ID or VPA Of Payer (Sender)

sandeepmore3@apl

Enter Amount

50000

Note

offer

Proceed to Pay

Mumbai, Maharashtra
400025, India

Customize

IP Address: 2401:4900:1907:799b:992a:a791:8a9e:fea9

SBI

Payment Gateway

SBI Life

Powered by Security of Z+ Guard

Developed By Adroit

✕ WARNING :: Unusal Behavior Check the Values and AMount Again.['New location and significantly high amount, potential fraud.', 'New UPI ID with an unusually high transaction amount, possible fraud.']

Scan UPI ID

Open Scanner

Enter UPI ID or VPA

nawazsayyad@okicici

Enter UPI ID or VPA Of Payer (Sender)

nitinjadhav@ybl

Enter Amount

50000

Note

Enter note (optional)

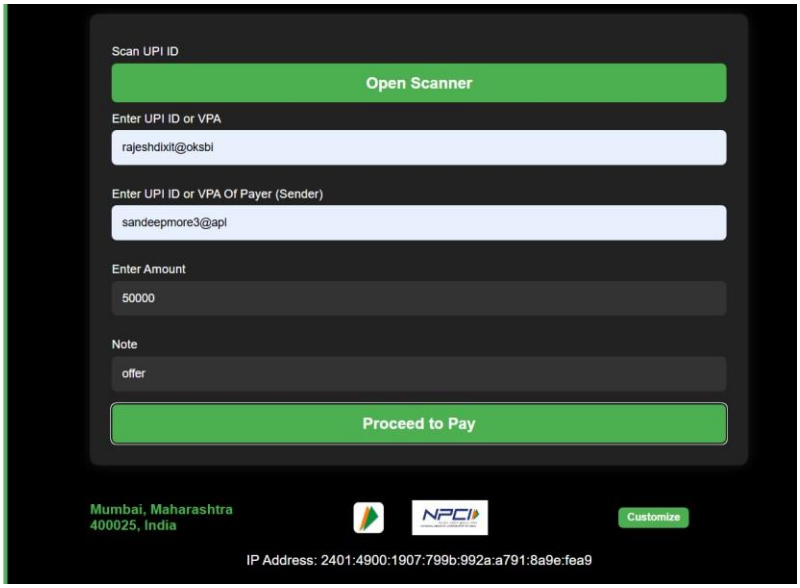
Proceed to Pay

Pune, Maharashtra 411013,
India

Customize

IP Address: 2401:4900:519e:4a1c:c50b:c316:5d7e:6f34

```
The type result is <class 'str'>
{'is_fraudulent': True, 'reasons': ['Due to high amount, this is f
]}
Fraud Or not ?True
Request received
{
    "fraud_or_scam_happened": "true",
    "summary": "The scammer initially sends small amounts to gain
 then convinces the victim to send increasingly larger amounts, pr
g multiplied returns. Once the victim sends a significant amount,
ammer disappears with the money.",
    "name": "Double Scam Fraud"
}
✅ Database connection established successfully.
✅ PDF updated: Double Scam Fraud
Executing query: SELECT COUNT(*) FROM account WHERE account_no = %
TRIM(LOWER(vpa)) = TRIM(LOWER(%s)) with values (526288, sandeepmor
)
Query result: 1
Executing query: SELECT COUNT(*) FROM account WHERE TRIM(LOWER(vpa
RIM(LOWER(%s)) with values (rajeshdixit@oksbi)
Query result: 1
Transaction inserted successfully.
```



Fraud Detetecd

PDF Generated :

.env	16-02-2025 18:55	File folder	
.git	24-02-2025 10:18	File folder	
__pycache__	24-02-2025 09:16	File folder	
Documents	23-02-2025 14:12	File folder	
official Standards	23-02-2025 16:18	File folder	
SBIBANKSERVER	22-02-2025 16:03	File folder	
static	22-02-2025 19:44	File folder	
templates	22-02-2025 19:07	File folder	
Zfilters	20-02-2025 12:05	File folder	
ZGuard	23-02-2025 21:59	File folder	
client	24-02-2025 10:01	Python Source ...	3 KB
patterndatabase	24-02-2025 08:16	JSON Source File	3 KB
rajeshdixit@oksbi_QR	23-02-2025 23:10	PNG File	1 KB
README	22-02-2025 18:20	Markdown Sou...	1 KB
requirements	23-02-2025 17:17	Text Document	1 KB
sandeepmore3@apl_QR	24-02-2025 09:03	PNG File	1 KB
z+	24-02-2025 09:16	Python Source ...	13 KB
Report	23-02-2025 23:47	Microsoft Word...	0 KB
Report-Adroit	23-02-2025 23:51	Brave HTML Do...	219 KB
fraud_report	24-02-2025 10:43	Brave HTML Do...	3 KB

Fraud Detection Report

Fraud Summary

Transaction: Double Scam Fraud

Summary: The scammer initially sends small amounts to gain trust, then convinces the victim to send increasingly larger amounts, promising multiplied returns. Once the victim sends a significant amount, the scammer disappears with the money.

Payer Details

Account Details:

ACCOUNT_NO: 526288

BANK_BALANCE: 103696277.96

BRANCH_NO: 122

ACCOUNT_STATUS: Active

ADDRESS: Kolhapur, Maharashtra

FIRST_NAME: Sandeep

MIDDLE_NAME: Gopal

LAST_NAME: More

PHONE_NUMBERS: +918739428084

VPA: sandeepmore3@apl

Payee Details

Account Details:

ACCOUNT_NO: 526269

BANK_BALANCE: 62570253.72

BRANCH_NO: 156

ACCOUNT_STATUS: Active

ADDRESS: Mumbai, Maharashtra

FIRST_NAME: Rajesh

MIDDLE_NAME: Madhukar

LAST_NAME: Dixit

PHONE_NUMBERS: +917630614594

VPA: rajeshdixit@oksbi

Transaction Details

Timestamp: 2025-02-24 10:43:37

Amount: INR 50000.0

Note: offer

Location: Mumbai, Maharashtra, India

Zipcode: 400025

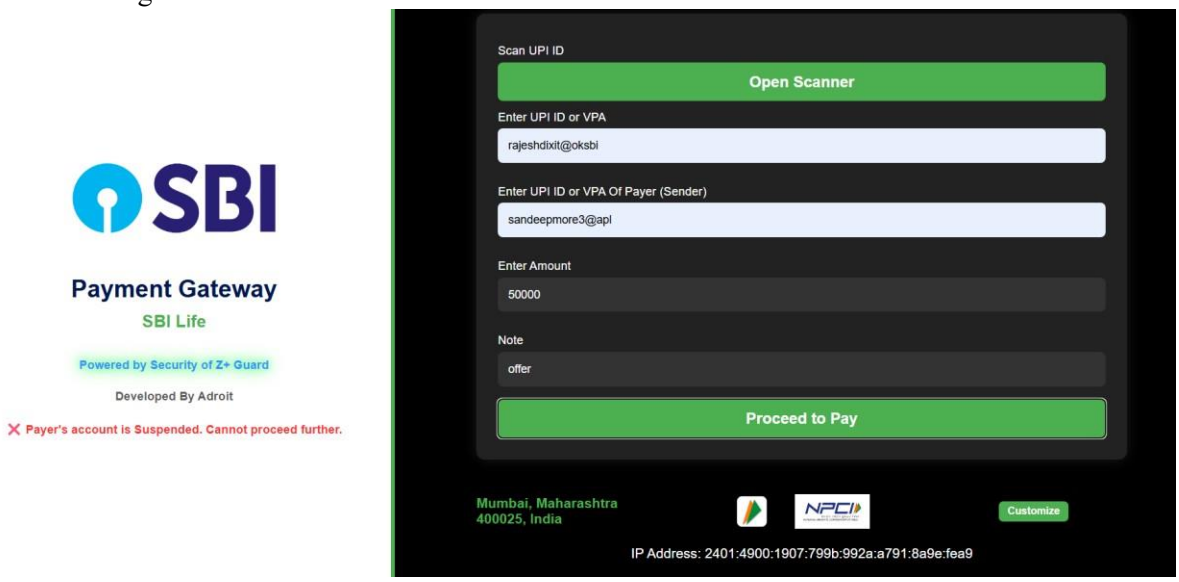
Device Type: mobile

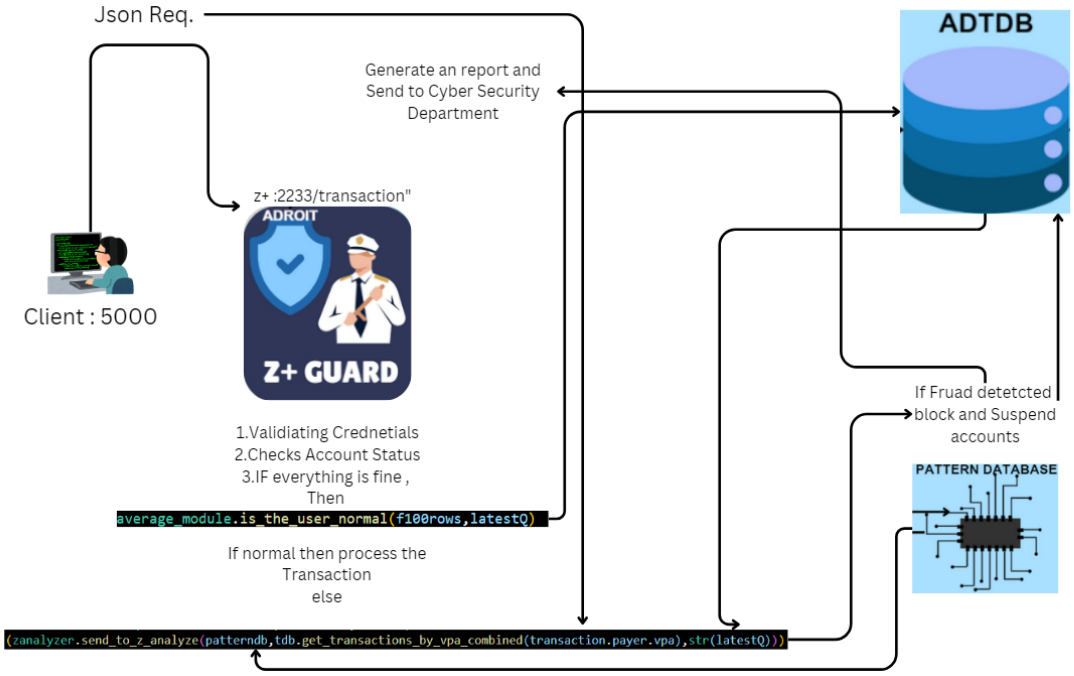
Mobile Carrier: CarrierName

Latitude: 18.5204, Longitude: 73.8567

IP Address: 2401:4900:1907:799b:992a:a791:8a9e:fea9

Even if tried Again





So this is The Best Apporoch :

USP of the Project: The project's biggest strength lies in its **zero-cost, open-source implementation** while providing **real-time fraud detection with automated response mechanisms**. The combination of **AI-driven behavioral analysis, automated reporting, and seamless scalability** makes it a unique and powerful solution for modern financial security challenges.