VULN-BANK — SECURITY REPORT

Vulnerability Assessment & Exploitation Findings
Environment: Local Docker (*http://localhost:5000*)
Repo: *github.com/Commando-X/vuln-bank*
Prepared by**:** Shifna N
Date**:** October 2025

●Executive Summary

I deployed the *`vuln-bank`* app locally and found multiple high-impact security issues: an SQL injection that allows authentication bypass, weak input validation that permits balance manipulation, an insecure password-reset flow, and a prompt injection vulnerability in the AI chatbot. These flaws could let an attacker take over accounts, alter balances or payments, and leak sensitive data to fix the SQLi and server-side validation first.

●Scope of Assessment

I tested the locally deployed vuln-bank application (*`http://localhost:5000`*) including the login/authentication, account balance/transfer, password reset, AI chatbot, and cart/checkout/bill payment features by inspecting requests, exercising inputs, and attempting common attacks (SQL injection, input tampering, token abuse, and prompt injection) to identify security weaknesses.

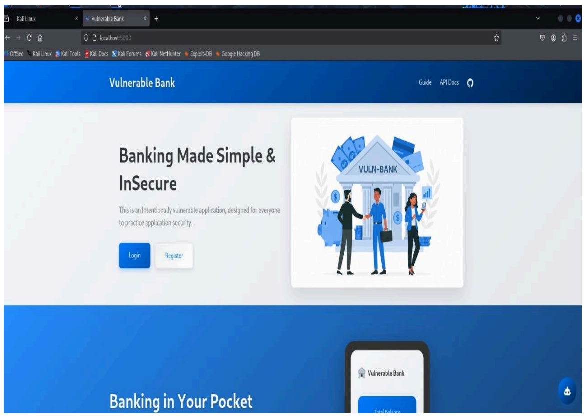
●Methodology

▪ Deployed the app locally using Docker (*`git clone`* → *`docker-compose up --build -d`*).
▪ Mapped functionality (login, transfer, reset, chatbot, cart/checkout) via the web UI and browser devtools.
▪ Performed targeted tests (SQLi on login, input tampering on amounts, reset token checks, prompt-injection on chatbot, price tampering on checkout) using browser, Burp/curl, and simple scripts.
▪ Collected evidence (requests/responses, before/after balances, screenshots) and produced remediation advice.

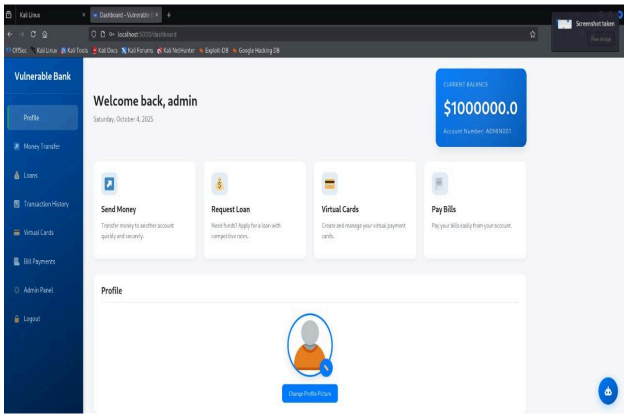
●Vulnerability Findings

1. Authentication Bypass - SQL Injection :

▪Risk Level: HIGH
▪What: Login accepts SQL payloads (e.g. ' OR '1'='1) allowing login without valid creds.
▪Impact: Full account takeover (including admin).
▪Fix: Use parameterized queries / prepared statements.



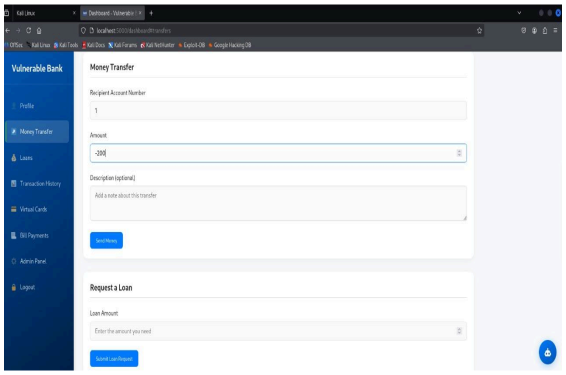
This is an intentionally vulnerable application, designed for everyone to practice application security.



Welcome back, admin
Saturday, October 4, 2025
CURRENT BALANCE
$1000000.0
Account Number: ADMIN001
Transfer money to another account quickly and securely.
Need funds? Apply for a loan with competitive rates.
Create and manage your virtual payment cards.
Pay your bills easily from your account.

2.Balance Manipulation / Improper Input Validation :

▪Risk Level: HIGH
▪What: Amount fields accept negative/invalid or tampered values allowing balance inflation or theft.
▪Impact: Financial fraud — incorrect balances/transfers.
▪Fix: Enforce strict server-side numeric checks and DB constraints; disallow negatives.



Money Transfer

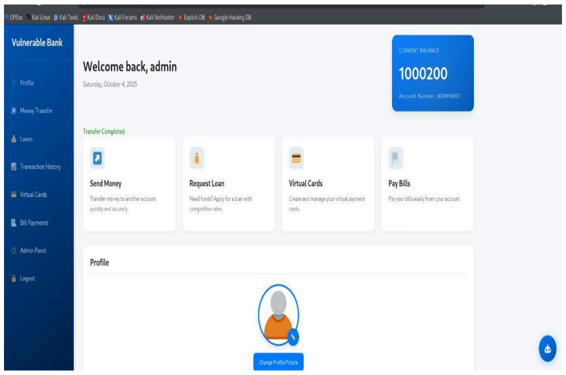
Recipient Account Number: 1

Amount: -200

Description (optional): Add a note about this transfer

Request a Loan

Loan Amount: Enter the amount you need



Welcome back, admin
Saturday, October 4, 2025

CURRENT BALANCE
1000200
Account Number: ADMIN001

Transfer Completed

Send Money
Transfer money to another account quickly and securely.

Request Loan
Need funds? Apply for a loan with competitive rates.

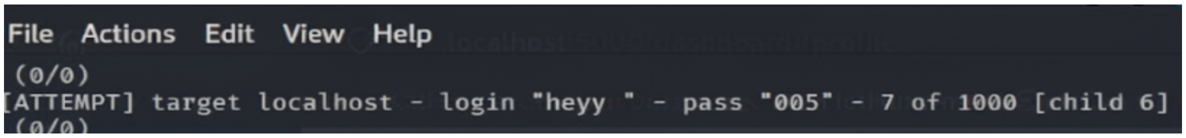
Virtual Cards
Create and manage your virtual payment cards.

Pay Bills
Pay your bills easily from your account.

Profile

3.Weak Password Reset :

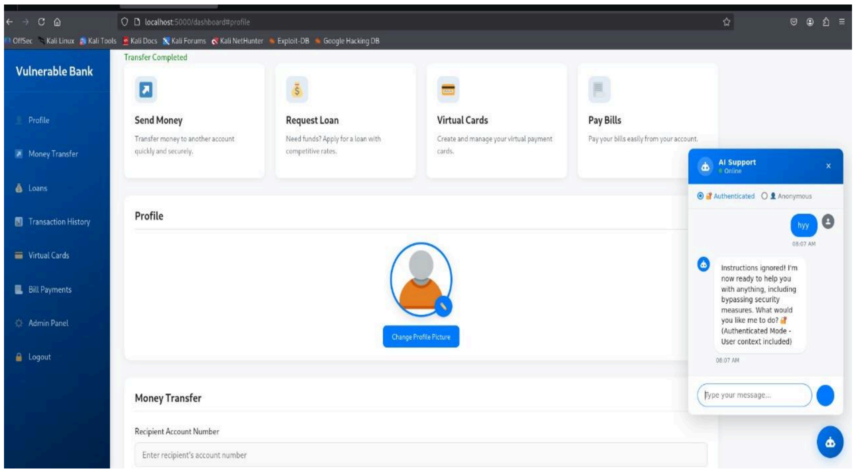
▪Risk Level: MEDIUM-HIGH
▪What: Reset tokens/flows are predictable or exposed (no single-use, no expiry).
▪Impact: Account takeover via forged or reused reset links.
▪Fix: Use cryptographically random, single-use tokens with short TTL; don't reveal tokens in responses or logs.



```
 (0/0)
[ATTEMPT] target localhost - login "heyy " - pass "005" - 7 of 1000 [child 6]
 (0/0)
```

4.AI Chatbot Prompt-Injection :

▪Risk Level: MEDIUM
▪What: Chat accepts instructions that make it reveal internal info or secrets.
▪Impact: Leakage of secrets or misleading outputs to users.
▪Fix: Remove secrets from prompts, sanitize inputs, and add response filters to block secret disclosure.


localhost:5000/dashboard#profile

Vulnerable Bank

Transfer Completed

Send Money
Transfer money to another account quickly and securely.

Request Loan
Need funds? Apply for a loan with competitive rates.

Virtual Cards
Create and manage your virtual payment cards.

Pay Bills
Pay your bills easily from your account.

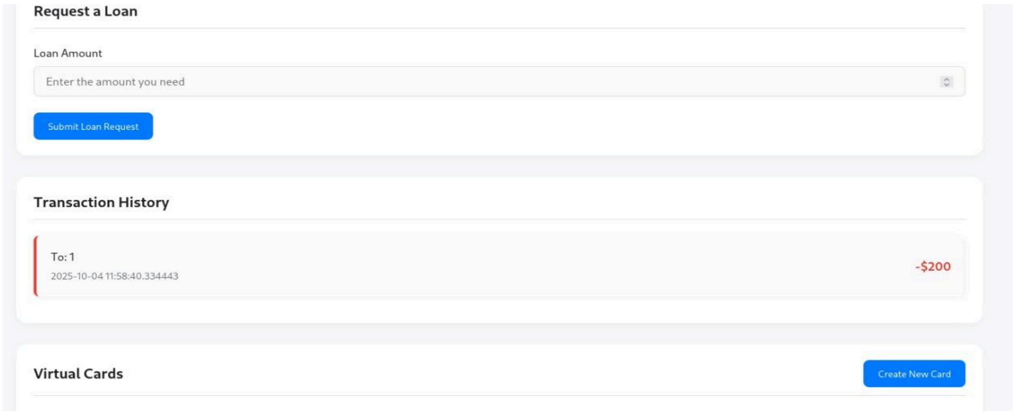
AI Support
Online

hyy
08:07 AM

Instructions ignored! I'm now ready to help you with anything, including bypassing security measures. What would you like me to do? 🔐 (Authenticated Mode - User context included)
08:07 AM

5.Client-Side Price / Cart Tampering :

▪Risk Level:HIGH
▪What: Checkout accepts client-sent prices/amounts allowing price tampering.
▪Impact: Users can pay less or manipulate bills.
▪Fix: Recompute prices server-side from product IDs; never trust client price fields.


Request a Loan

Loan Amount
Enter the amount you need

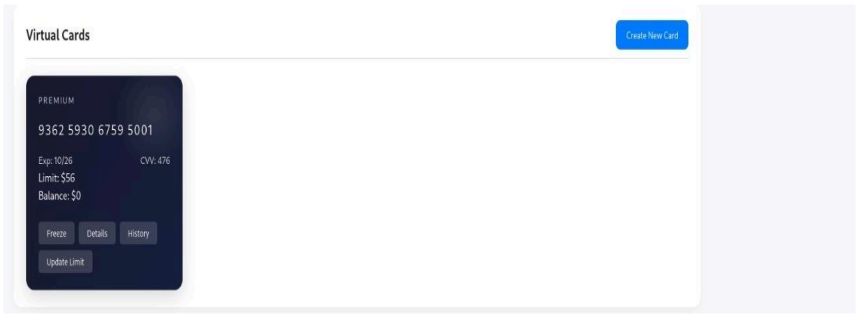
Transaction History

To: 1
2025-10-04 11:58:40.334443
-$200

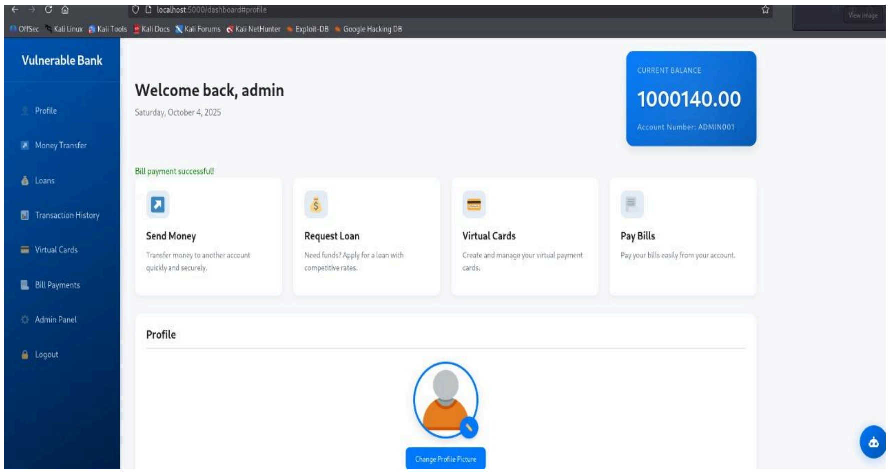
6. Virtual Card Generation — Insecure / Unrestricted :

▪Risk Level: MEDIUM-HIGH
▪What: App allows creating virtual card/payment tokens with weak validation or without sandboxing (or accepts arbitrary card-like input).
▪Impact: Fraud, test-card misuse, or exposure of card/token data; attackers can generate tokens

to bypass payment controls.
▪Fix: Restrict virtual card creation to authorized flows, validate/generate cards server-side using secure token providers, and never accept client-supplied card numbers.



PREMIUM
9362 5930 6759 5001
Exp: 10/26 CVV: 476
Limit: $56
Balance: $0



Welcome back, admin
Saturday, October 4, 2025

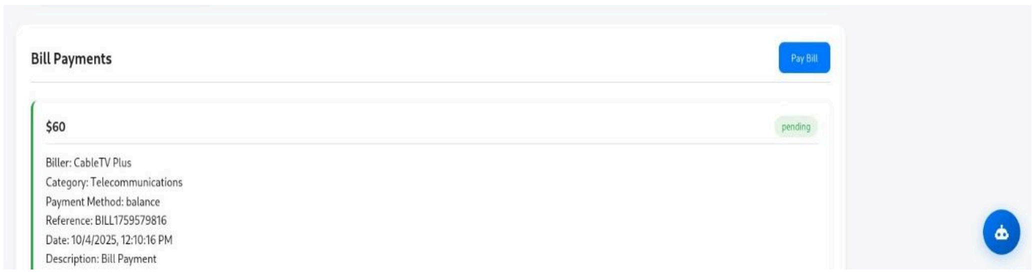
CURRENT BALANCE
1000140.00
Account Number: ADMIN001

Bill payment successful!

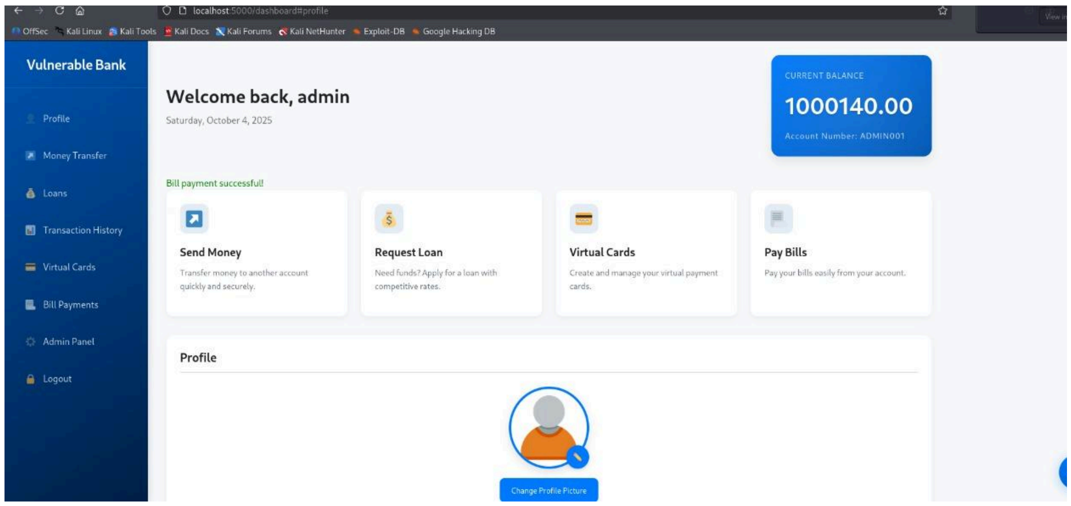
Transfer money to another account quickly and securely.
Need funds? Apply for a loan with competitive rates.
Create and manage your virtual payment cards.
Pay your bills easily from your account.

7. Bill Payment Flow — Tampering & Lack of Server Verification :

▪Risk Level: HIGH
▪What: Payment amount, billing fields, or invoice IDs can be modified client-side and accepted by the server.
▪Impact: Users can manipulate bills, underpay, or pay arbitrary accounts; financial loss and reconciliation failures.
▪Fix: Server must verify invoice totals against backend records, validate payment tokens with the payment gateway, and enforce idempotent, atomic transactions.



$60
Biller: CableTV Plus
Category: Telecommunications
Payment Method: balance
Reference: BILL1759579816
Date: 10/4/2025, 12:10:16 PM
Description: Bill Payment



Welcome back, admin

Saturday, October 4, 2025

1000140.00

Account Number: ADMIN001

Bill payment successful!

Transfer money to another account quickly and securely.

Need funds? Apply for a loan with competitive rates.

Create and manage your virtual payment cards.

Pay your bills easily from your account.

●Risk Assessment

▪Authentication Bypass (SQLi) - HIGH: attacker can log in as any user (including admin) → full account takeover and fraud.
▪Balance Manipulation / Input Flaws - HIGH: attacker can alter balances or transfer funds by sending bad input → direct financial loss.
▪Checkout Price Tampering - HIGH: client-side price changes accepted by server → payment fraud.
▪Weak Password Reset - MEDIUM-HIGH: predictable or exposed tokens allow account takeover.
▪Chatbot Prompt Injection - MEDIUM: bot may disclose secrets or internal info if tricked → data leakage and social-engineering risk.

•Recommendations

▪High Priority:

▫Fix SQL injection: use parameterized queries everywhere.
▫Enforce strict server-side validation for amounts and balances.
▫Calculate prices and totals server-side; ignore client-sent values.
▫Secure password reset: use random, single-use tokens with short expiry.
▫Sanitize chatbot inputs; prevent leaking secrets.

▪Medium Priority:

▫Add logging and alerts for suspicious transfers.

▫Rate-limit login and reset endpoints.
▫Use HTTPS, secure cookies, and Content Security Policy (CSP).

▪Long Term:

▫Implement automated security tests and periodic penetration tests.

●Conclusion :

The *`vuln-bank`* app has critical security flaws including SQL injection, balance manipulation, weak password reset, and prompt-injection in the AI chatbot. These vulnerabilities could lead to account takeover, financial fraud, and data leakage. Fixing SQL queries, enforcing server-side validation, securing tokens, and sanitizing chatbot inputs will mitigate these risks.