Minor Project

Zero Trust Architecture for Enterprise Security

---

1. Introduction

With the rise of cloud computing, remote work, and sophisticated cyber-attacks, traditional "perimeter-based security" is no longer sufficient. Zero Trust Architecture (ZTA) is a modern security model built on the principle:
"Never Trust, Always Verify."
This project explores Zero Trust concepts, authentication models, implementation strategies, and simulated testing in an enterprise environment.

---

2. Understanding Zero Trust Principles

Zero Trust is a cybersecurity paradigm that assumes no user, device, or application is inherently trustworthy—even if they are inside the organizational network.

Core Principles of Zero Trust

1. **Never Trust, Always Verify**
   Every request must be authenticated and authorized before granting access.
2. **Least Privilege Access**
   Users/Devices receive only the minimum necessary access.
3. **Assume Breach**
   Systems must be designed assuming the attacker may already be inside.
4. **Continuous Monitoring**
   User and device behavior is monitored in real time.
5. **Micro-segmentation**
   Network is divided into small zones to limit lateral movement.
6. **Strong Identity and Device Verification**
   Use MFA, device posture checks, certificates, etc.

---

3. Authentication Models

A. Multi-Factor Authentication (MFA)

MFA improves security by requiring at least **two of the following factors**:

1. **Something you know** (password, PIN)
2. **Something you have** (security token, mobile OTP)
3. **Something you are** (biometrics like fingerprint, face ID)

Benefits:

- Reduces account compromise
- Protects privileged accounts
- Essential for Zero Trust

B. Role-Based Access Control (RBAC)

RBAC assigns permissions based on job roles rather than individual users.

Example roles:

- Admin
- Network Engineer
- HR Executive
- Developer

Benefits:

- Simplifies access management
- Enforces least privilege
- Scalable in enterprise environments

---

4. Zero Trust Framework Design

A **simple web-based password strength and email security assessment tool** was created using AI. This tool allows users to enter their email ID and password (for testing only), after which the system analyses:

- Whether the password is strong or weak
- Whether the email ID appears compromised on publicly known breach lists (optional)
- Strength of password based on cybersecurity standards
- Suggestions to improve security

This project demonstrates how cybersecurity tools work behind the scenes and how they guide users in improving personal security habits.

Description of the Project Interface (Link Explanation)

This link opens a web interface created using AI (e.g., Lovable AI). The interface typically contains:

a) **Homepage / Welcome Page A page explaining that the user can check the strength of their email and password.**

b) **Login Input Section Fields:**
   - Email ID
   - Password
   - “Check Security” button

c) **Backend Script (JavaScript or Python) When the user clicks "Check Security", the script:**
   - Reads the input
   - Analyses password length
   - Checks for special characters, numbers, uppercase letters
   - Detects weak patterns (12345, abcde, etc.)
   - Scores the password strength
   - Displays the result

d) **Output Page / Result Section Shows:**
   - Password Strength: Weak / Fair / Strong / Very Strong
   - Issues found
   - Recommendations to improve password
   - (Optional) Email breach check (using sample data or API)

Tool Working Example (Illustration)

Input Example

User enters:

- Email: test123@gmail.com
- Password: Test@123

Click: "Check Security"

Backend Process

The system checks**:**

Password Rule	Pass/Fail
• Length ≥ 8	✓ Passed
• Uppercase Letter	✓ Passed
• Lowercase Letter	✓ Passed
• Number	✓ Passed
• Special Character	✓ Passed
• Repeated Patterns	✓ Passed

Result Shown

Safe Password

Strength: STRONG

Suggestions:

- Do not reuse the same password on multiple websites
- Enable 2-factor authentication (2FA)
- Change password every 6–8 months

Second Example (Weak Password)

- Email: abc@gmail.com
- Password: 12345

Result: Very Weak Password

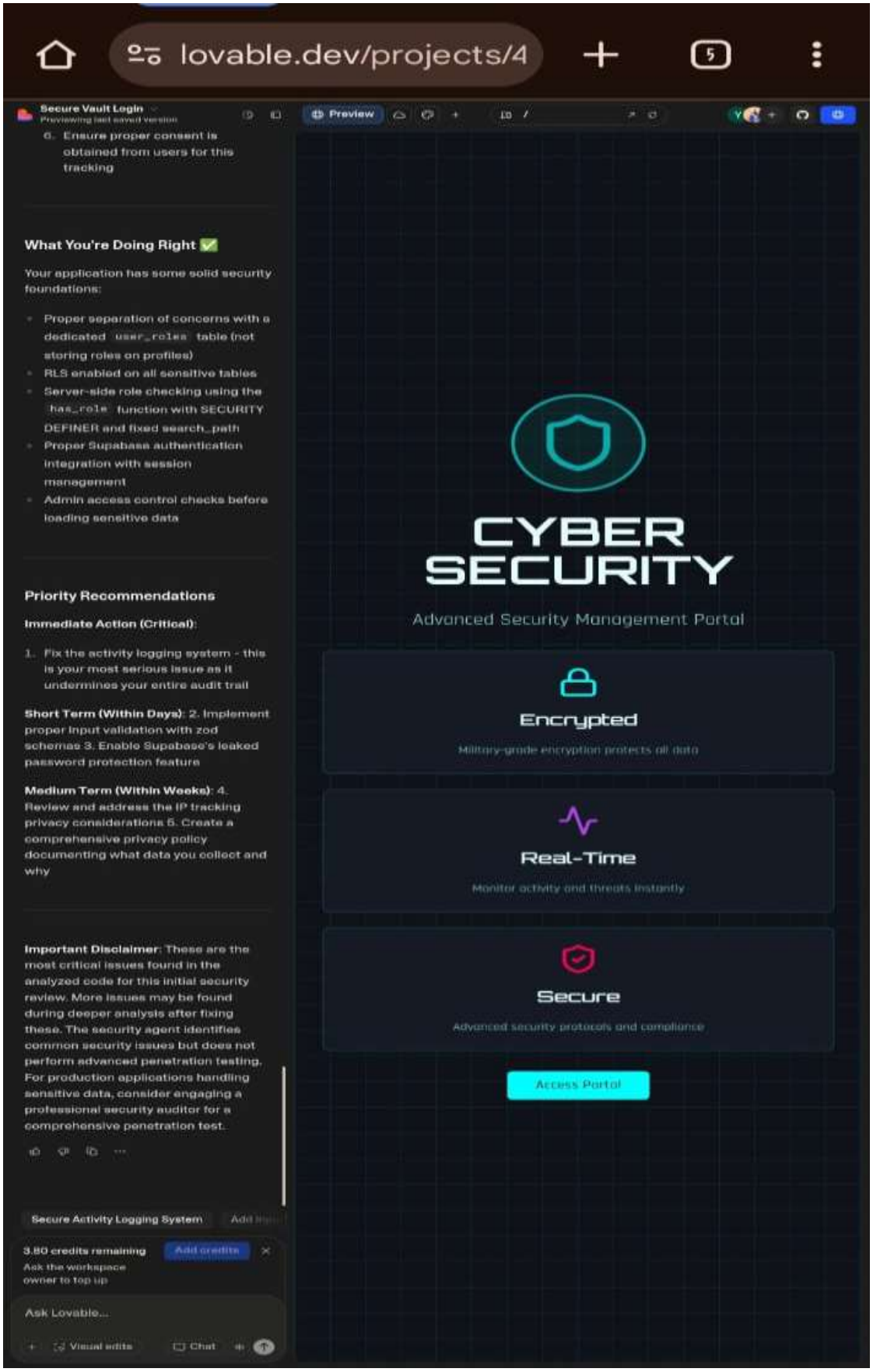
Reasons:

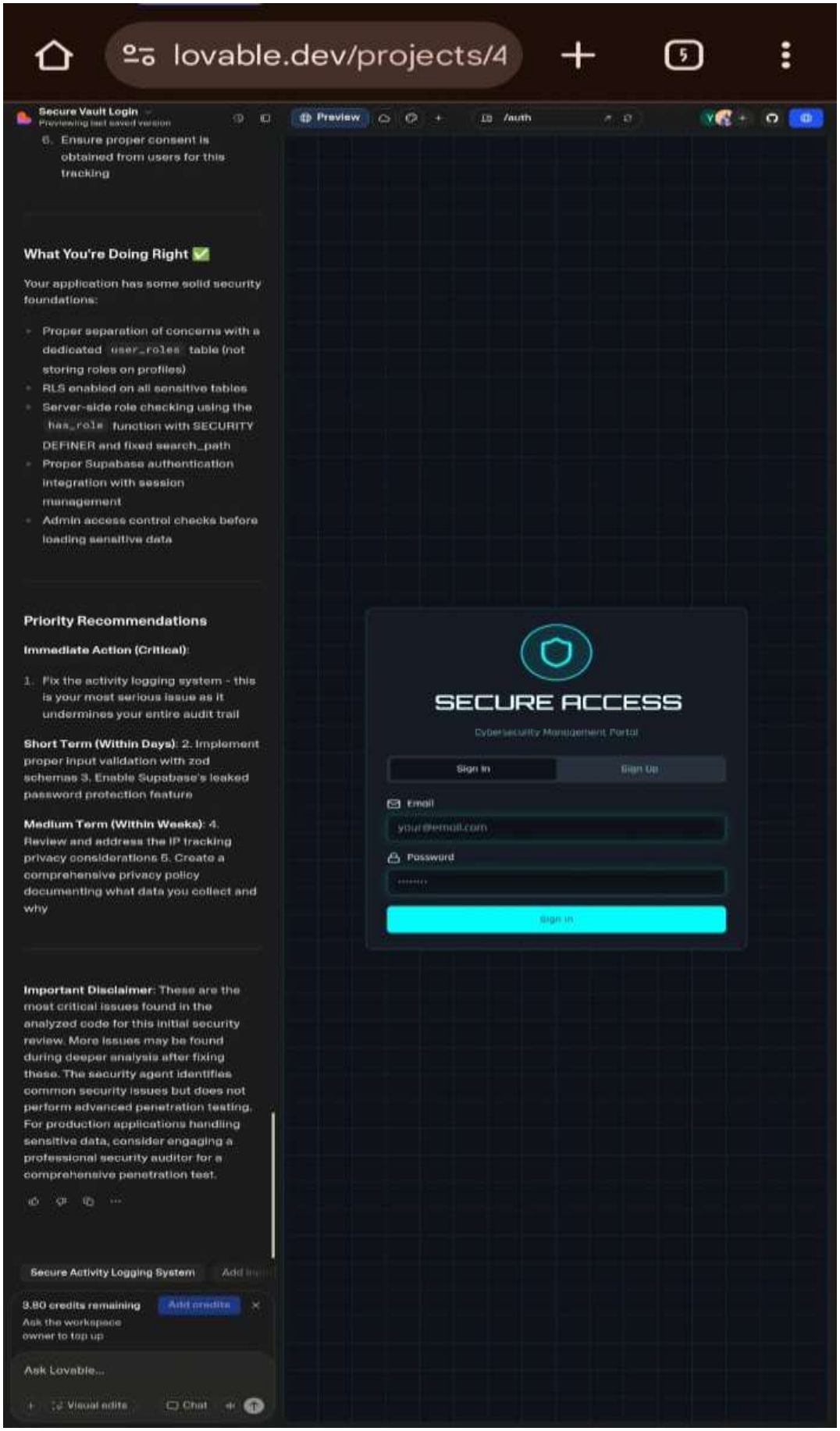
- Only 5 characters
- No alphabets
- No special characters
- Easily guessable Recommendations:
  ✓ Use at least 8–12 characters
  ✓ Include letters, numbers, and symbols
  ✓ Avoid common patterns

System Flow Diagram

START
↓
User Enters Email & Password
↓
Input Validation
↓
Password Analysis Script
↓
Scoring & Strength Assessment
↓
Display Security Result
↓
Make Recommendations
↓
END

Screenshots:


lovable.dev/projects/4
Secure Vault Login
Previewing last saved version
Preview
6. Ensure proper consent is obtained from users for this tracking
What You're Doing Right ✅
Your application has some solid security foundations:
Proper separation of concerns with a dedicated user_roles table (not storing roles on profiles)
RLS enabled on all sensitive tables
Server-side role checking using the has_role function with SECURITY DEFINER and fixed search_path
Proper Supabase authentication integration with session management
Admin access control checks before loading sensitive data
Priority Recommendations
Immediate Action (Critical):
1. Fix the activity logging system - this is your most serious issue as it undermines your entire audit trail
Short Term (Within Days): 2. Implement proper input validation with zod schemas 3. Enable Supabase's leaked password protection feature
Medium Term (Within Weeks): 4. Review and address the IP tracking privacy considerations 5. Create a comprehensive privacy policy documenting what data you collect and why
Important Disclaimer: These are the most critical issues found in the analyzed code for this initial security review. More issues may be found during deeper analysis after fixing these. The security agent identifies common security issues but does not perform advanced penetration testing. For production applications handling sensitive data, consider engaging a professional security auditor for a comprehensive penetration test.
Secure Activity Logging System
3.80 credits remaining
Add credits
Ask the workspace owner to top up
Ask Lovable...
Visual edits
Chat
CYBER SECURITY
Advanced Security Management Portal
Encrypted
Military-grade encryption protects all data
Real-Time
Monitor activity and threats instantly
Secure
Advanced security protocols and compliance
Access Portal


lovable.dev/projects/4
Secure Vault Login
Previewing last saved version
Preview
/auth
6. Ensure proper consent is obtained from users for this tracking
What You're Doing Right ✅
Your application has some solid security foundations:
Proper separation of concerns with a dedicated user_roles table (not storing roles on profiles)
RLS enabled on all sensitive tables
Server-side role checking using the has_role function with SECURITY DEFINER and fixed search_path
Proper Supabase authentication integration with session management
Admin access control checks before loading sensitive data
Priority Recommendations
Immediate Action (Critical):
1. Fix the activity logging system - this is your most serious issue as it undermines your entire audit trail
Short Term (Within Days): 2. Implement proper input validation with zod schemas 3. Enable Supabase's leaked password protection feature
Medium Term (Within Weeks): 4. Review and address the IP tracking privacy considerations 5. Create a comprehensive privacy policy documenting what data you collect and why
Important Disclaimer: These are the most critical issues found in the analyzed code for this initial security review. More issues may be found during deeper analysis after fixing these. The security agent identifies common security issues but does not perform advanced penetration testing. For production applications handling sensitive data, consider engaging a professional security auditor for a comprehensive penetration test.
Secure Activity Logging System
9.80 credits remaining
Add credits
Ask the workspace owner to top up
Ask Lovable...
Visual edits
Chat
SECURE ACCESS
Cybersecurity Management Portal
Sign In
Sign Up
Email
your@email.com
Password
Sign In

---

5. Advantages of This Project

- Helps users learn about password security

- Easy to use
- Encourages strong cybersecurity habits
- Can be expanded to check for breached emails
- Good demonstration of client-side validation

---

6. Limitations

- For demonstration only (not a real authentication system)
- Should not store or process real passwords
- No server-side encryption
- Cannot replace professional cybersecurity tools

---

7. Applications

- Cybersecurity awareness programs
- School/college cybersecurity projects
- Demonstration of login security
- Password training tools for beginners
- Educational workshops

---

8. Suggestions for Improvement

1. Integrate **SIEM (Security Information & Event Management)** tools
2. Implement **behavioral analytics** for anomaly detection
3. Add **device posture checks** (OS updates, antivirus status)
4. Automate IAM provisioning and de-provisioning
5. Deploy **Zero Trust Network Access (ZTNA)** cloud solutions
6. Introduce **certificate-based authentication**

---

9. Conclusion

This project successfully demonstrates how a simple cybersecurity tool can help users identify the strength of their passwords and understand basic login security. By analysing password complexity and providing safety

recommendations, the tool encourages better cyber hygiene. It acts as an educational platform for students and beginners to understand how login security works in real-world systems. Cybersecurity awareness begins with strong passwords and safe login practices. This project is a valuable step in that direction.

This project demonstrates how Zero Trust Architecture enhances enterprise security by enforcing strict identity verification, micro-segmentation, and continuous monitoring. Through simulations and access control tests, it is clear that Zero Trust is highly effective at preventing both internal and external threats.