Security Vulnerability Report

Application Name: Mitt Arv (Legacy-Tech Platform)
Tested By: P. Achyuth Reddy
Date: 07/03/2025
Referral Code: 2402ASMT

Slide 1: Executive Summary

Key Findings

- Weak authentication mechanisms
- Insecure data storage
- Poor input validation
- Lack of real-time security alerts
- API vulnerabilities
- Outdated dependencies

Recommendation

Implement strong authentication, encryption, and validation mechanisms. Update dependencies and secure APIs.

Slide 2: Authentication & Authorization Issues

Risk Level: ⚠ High

Issues:

- No mandatory login or strong password policies
- No biometric/facial recognition for sensitive actions
- Persistent sessions allow unauthorized access

Steps to Reproduce:

1. Install the app and skip login/signup.
2. Access "Asset Vault" and enter bank details without credentials.
3. Reopen the app to bypass re-authentication.

Impact:

- Attackers can access financial data without credentials.
- Insecure Data Storage

Risk Level: ⚠ High

Issues:

- Financial data stored in plaintext
- Centralized servers pose a security risk

Steps to Reproduce:

1. Enter bank details (e.g., 20-digit invalid account number).
2. Use ADB to extract unencrypted data:

```
adb shell
cd /data/data/com.mittarv.app/databases
cat *.db
```

Impact:

- Data breaches due to unencrypted storage.

Slide 4: Weak Input Validation

Risk Level: ⚠ Medium

Issues:

- No validation for:
  - Bank account numbers (accepts >12 digits)
  - IFSC codes (no format checks)
  - File uploads (accepts .exe, .zip)

Steps to Reproduce:

1. Enter invalid account numbers (e.g., 12345678901234567890).
2. Upload a non-financial file (e.g., .exe).

Impact:

- Data corruption or malware injection.

Slide 5: Lack of Real-Time Notifications

Risk Level: ⚠ Low

Issues:

- No alerts for critical actions (e.g., data changes, logins)

Impact:

- Users cannot detect unauthorized access.

Slide 6: API Security Vulnerabilities

Risk Level: ⚠ High

Issues:

- Unauthenticated API access
- No rate limiting, allowing brute force attacks
- Insecure API endpoints exposing sensitive data

Steps to Reproduce:

1. Intercept API requests using Burp Suite.
2. Modify parameters to access unauthorized data.

Impact:

- Data leaks and account takeovers.

.

Slide 8: Authentication & Authorization Fixes

Solutions:

- Implement **Multi-Factor Authentication (MFA)** (email/phone + strong password)
- Add **facial recognition** for sensitive actions
- **Session Management**: Force re-authentication after 15 minutes of inactivity

---

Slide 9: Secure Data Storage Fixes

Solutions:

- Use **AES-256 encryption** for data at rest
- Use **TLS 1.3** for data in transit

- Implement **decentralized storage (e.g., IPFS)**

Slide 10: Input Validation Fixes

Solutions:

- **Bank Account Validation:** Ensure 9-18 digit account numbers
- `const validateAccount = (acc) => /^\d{9,18}$/.test(acc);`
- **IFSC Code Validation:** Check format (e.g., SBIN0000123)
- `const validateIFSC = (ifsc) => /^[A-Z]{4}0[A-Z0-9]{6}$/.test(ifsc);`
- **File Upload Restrictions:** Allow only **PDF/PNG/JPEG** files

---

Slide 11: API Security Fixes

Solutions:

- **Authentication for all API endpoints**
- **Rate limiting** to prevent brute force attacks

Slide 12: Dependency Management Fixes

Solutions:

- Regularly update third-party libraries
- Use automated security scanners (e.g., `npm audit`, `Snyk`)
- Implement CI/CD security checks

Slide 13: User Education & Support

Solutions:

- Add an **in-app tutorial** on security best practices
- Integrate **24/7 live chat support** for security concerns

Slide 14: Conclusion

Summary:

The identified vulnerabilities pose significant risks to user trust and data integrity. Implementing **strong authentication, encryption, API security, and input validation** will enhance security. Immediate action is recommended.

Prepared By:
P. Achyuth Reddy
[+91 9392777253]
[paramayolla.achyuth.reddy@gmail.com]
[https://github.com/Achyuth1818/mittarv-internship]