404 Not Found

Digital Newspaper Website
Use-Case Specification: Verify MFA code

Version 1.0

Revision History

Date	Version	Description	Author
18/07/2025	1.0	Complete the document	Lê Đình Minh Quân

Table of Contents

Use-Case Specification: Verify MFA Code

1. Use-Case Name: Verify MFA Code

1.1 Brief Description

The *Verify MFA Code* use case describes how a partially authenticated User (after successful primary credential entry) completes multi‑factor authentication (MFA) by submitting a one‑time code (OTP) delivered via an approved second factor (e.g., authenticator app TOTP, SMS, email fallback, hardware key challenge, or push approval). The system validates the code or challenge response, establishes a fully authenticated session with appropriate assurance level, and enforces retry, lockout, and security policies.

2. Flow of Events

2.1 Basic Flow

Actor: User (partially authenticated; primary login successful)
Precondition: Primary credentials accepted; system requires MFA; a valid MFA factor is enrolled and a current one‑time code (or challenge) is available to the user.
Trigger: System presents MFA verification prompt.

1. The system displays the **MFA verification screen** indicating the active factor (e.g., “Enter the 6‑digit code from your authenticator app”).
2. The user enters the MFA code (or responds to the presented factor, e.g., types TOTP, approves push, taps hardware key).
3. System normalizes and validates input format (trims spaces, rejects non‑digits if TOTP).
4. The system verifies the code/challenge against the factor source (e.g., TOTP time window, push approval status, WebAuthn signature).
5. System checks **attempt counters** (e.g., max consecutive failures) and **token freshness / time drift**.
6. On success, the system elevates the session from *partial* to *fully authenticated*, sets MFA assurance attributes (e.g., mfaLevel = strong, authn_time = timestamp).
7. System clears transient MFA state (e.g., pending challenge IDs, temporary cookies).
8. System redirects users to the originally requested protected resource (or default post‑login landing page).
9. System logs an audit event { userId, factorType, outcome = SUCCESS, ip, userAgent, timestamp}.
10. The user continues with full privileges (use case ends).

2.2 Alternative Flows

2.2.1 Invalid Code (Single Failure)

Point of deviation: Step 4.

1. *Code does not match expected value (wrong digits, incorrect WebAuthn signature, push denied).*
2. *System increments failure counter and logs failure event.*
3. *System informs user: "Invalid code. Please try again." and re-prompts (remaining attempts indicated).*
4. *Flow returns to Basic Flow Step 2 (unless lockout reached).*

2.2.2 Code Expired / Time Window Drift

Point of deviation: Step 4.

1. *Code structure correct but outside valid time window (TOTP drift, expired SMS).*
2. *System shows: "Code expired. Request or generate a new code."*
3. *(If drift is small and policy allows) system suggests user resync device time.*
4. *Flow returns to Basic Flow Step 1 (or Step 2 after new code entry).*

2.2.3 Maximum Attempts Reached (Temporary Lockout)

Point of deviation: After repeated executions of 2.2.1.

1. *Failure counter hits configured threshold (e.g., 5 attempts).*
2. *The system locks further attempts for a cooldown period (e.g., 5 minutes) and logs outcome {outcome = LOCKOUT}*
3. *The system displays a lockout message with remaining wait time.*
4. *Use case ends (user may retry after cooldown → re-enter from Basic Flow Step 1).*

2.2.4 Factor Not Available (User Lost Device)

Point of deviation: Step 1 (user cannot supply code).

1. *The user selects the "I can't access my authenticator" link.*
2. *The system offers enrolled **backup factors** (e.g., backup codes, SMS, email, hardware key) per policy hierarchy.*

3. *The user selects a backup factor → system sends / prompts for that factor's challenge.*
4. *Flow resumes at Basic Flow Step 2 for the new factor.*
5. *System marks event with **factorFallback = true**.*

2.2.5 Using a Backup Recovery Code

Point of deviation: Step 2.

1. *The user enters one of their one-time backup recovery codes instead of standard factor code.*
2. *System matches hash; if valid, marks code as **consumed** (cannot reuse).*
3. *The system proceeds at Basic Flow Step 6 (successful elevation) and logs **factorType = RECOVERY_CODE.***
4. *The system warns the user if remaining recovery codes are low (e.g., ≤2).*

2.2.6 Push Notification Factor (Approval Flow)

Point of deviation: Step 2 (factor = push).

1. *Instead of entering digits, the user receives a push prompt on the device and taps **Approve**.*
2. *System polls or gets callback; on approval continues at Basic Flow Step 6.*
3. *If the user taps **Deny** (or times out), treat it as failure → go to 2.2.1 logic.*

2.2.7 Hardware Security Key (WebAuthn) Factor

Point of deviation: Step 2 (factor = hardware key).

1. *System initiates WebAuthn challenge in browser; user taps key or provides biometric.*
2. *System verifies signature (credential ID, origin, counter).*
3. *On success resume at Basic Flow Step 6; on failure resume at 2.2.1.*

2.2.8 Fallback to Primary Re-Authentication (Session Stale)

Point of deviation: Step 5.

1. *The system detects partial sessions have exceeded allowed pre-MFA age.*
2. *System discards partial session and redirects user back to primary login (restarts overall authentication).*

3. *After successful primary login again, flow returns to Basic Flow Step 1.*

2.2.9 Forced Step-Up During Active Session

Point of deviation: External protected action requests higher assurance while the user is already logged in at a lower level.

1. *The user tries a sensitive operation (e.g., change password, view PII).*
2. *System interrupts with MFA verification (step-up).*
3. *On success proceed to protected action (outside this use case's scope).*
4. *On failure follow alternative flows 2.2.1–2.2.3; if abandoned, action is canceled.*

2.2.10 Suspicious Attempt (Risk Engine Intervention)

Point of deviation: Step 4.

1. *Risk engine flags anomalies (geo-velocity, device mismatch).*
2. *The system requires additional factors (step-up inside step-up) or denies attempt.*
3. *If additional factor pass, resume at Basic Flow Step 6; if denied, record* ***outcome = RISK_DENY*** *and end use case.*

2.2.11 Session Continuation After Partial Failures

Point of deviation: Repeated Step 2 with intermittent failures below lockout threshold.

1. *The user fails once or twice but later provides a correct code within allowed attempts/time.*
2. *System resets consecutive failure counters after success.*
3. *Flow rejoins Basic Flow Step 6.*

2.2.12 User Cancels Verification

Point of deviation: Step 2.

1. *User clicks* ***Cancel*** */ closes dialog.*
2. *System discards partial session or returns to primary login screen with message "MFA required to continue."*
3. *Use case ends (no elevation).*

3. Special Requirements

- **Security:** Support multiple factors (TOTP, WebAuthn, push, backup codes) per enrolled configuration; verification must resist replay (nonce/time-based).
- **Time Drift Handling:** Accept TOTP codes within configurable ± time window (e.g., ±30s) while logging drift if > expected.
- **Rate Limiting:** Enforce per-user and per-IP attempt throttling (e.g., exponential backoff after repeated failures).
- **Lockout Policy:** Configurable thresholds; lockout events auditable.
- **Data Protection:** Never log raw codes; only success/failure metadata.
- **Accessibility:** All form controls keyboard-navigable; labels tied to inputs; announce errors via ARIA live region.
- **Internationalization:** Error and instruction messages localizable.
- **Device Binding:** WebAuthn requires origin and RP ID match; counters monitored to detect cloned keys.
- **Privacy:** Minimize PII in audit logs; no factor secrets stored in plain text.
- **Performance:** Verification round trip < 1 second 95th percentile (excluding user entry time).
- **Resilience:** Retry guidance offered after transient network errors.
- **Compliance:** Align with MFA policy (e.g., NIST 800-63B AAL2+) for enrolled factors.
- **Telemetry:** Emit structured events for success, failure, lockout, fallback use.

4. Preconditions

- The user completed primary credential authentication (username/password or SSO) and is in partial session state.
- At least one MFA factor is enrolled and active.
- MFA enforcement policy requires verification for this login or step-up action.
- Verification service (TOTP validator, WebAuthn API, push service) is operational.

5. Postconditions

- **Success:** Session promoted to fully authenticated with recorded assurance level; failure counters reset.
- **Lockout:** User blocked from further verification attempts until cooldown ends; partial session may be invalidated.

- **Abandon / Cancel:** Partial session terminated or left unusable for protected resources.
- **Fallback Success:** Backup or alternative factor verified; session elevated with **factorFallback = True.**
- **Risk Denied:** Attempt blocked; security alert optionally raised.

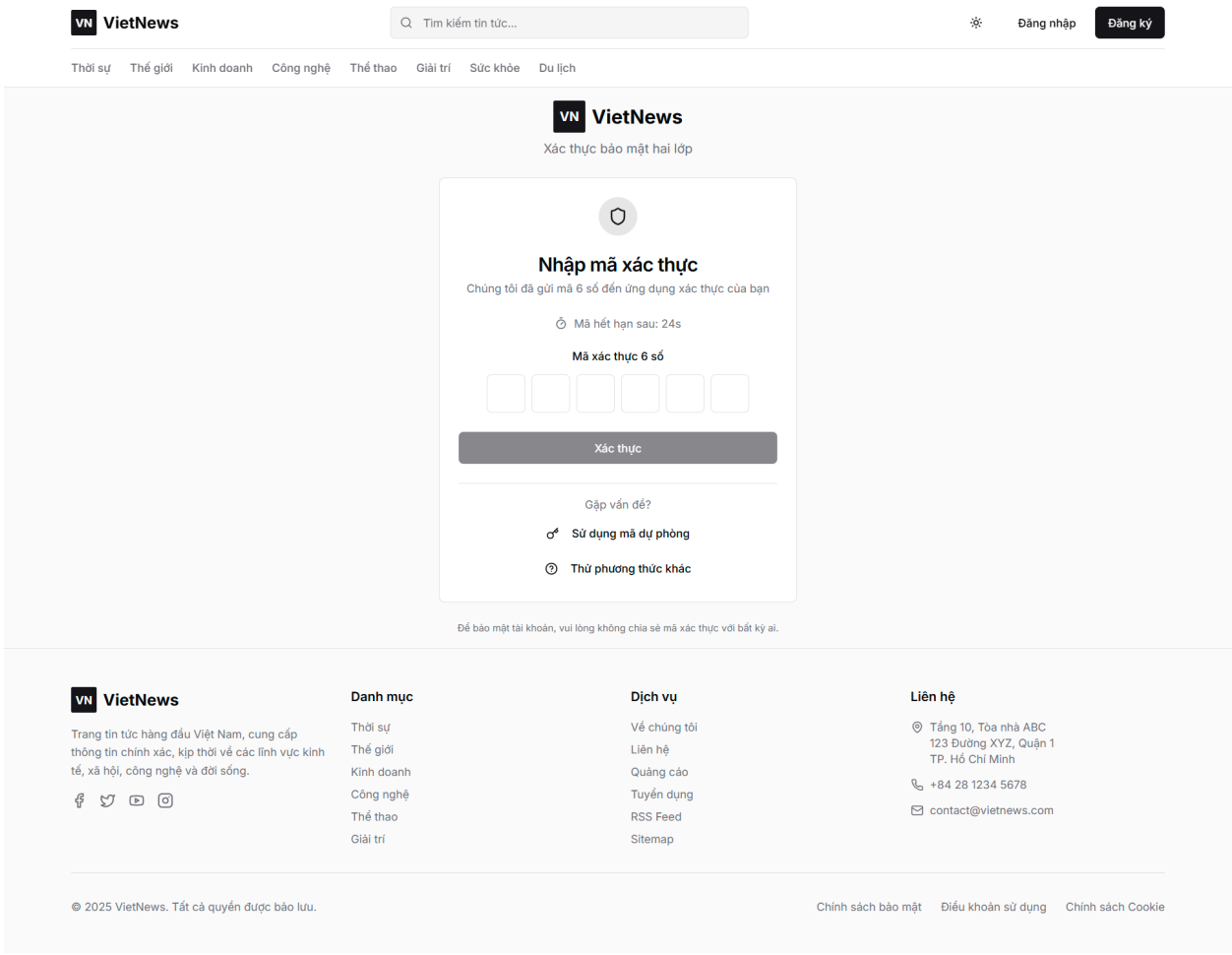
6. Extension Points

- **Adaptive Factor Selection: Insert between Basic Flow Steps 1–2 to dynamically choose factor based on risk scoring.**
- **Remember Device (Trusted Device Cookie): After Step 6, optionally set a signed "trusted device" token to skip MFA for low‑risk future logins.**
- **Device Time Sync Suggestion: During 2.2.2 offer a one‑click "Sync Time" helper or instructions.**
- **Additional Consent Prompt: After Step 6 for sensitive administrative consoles.**

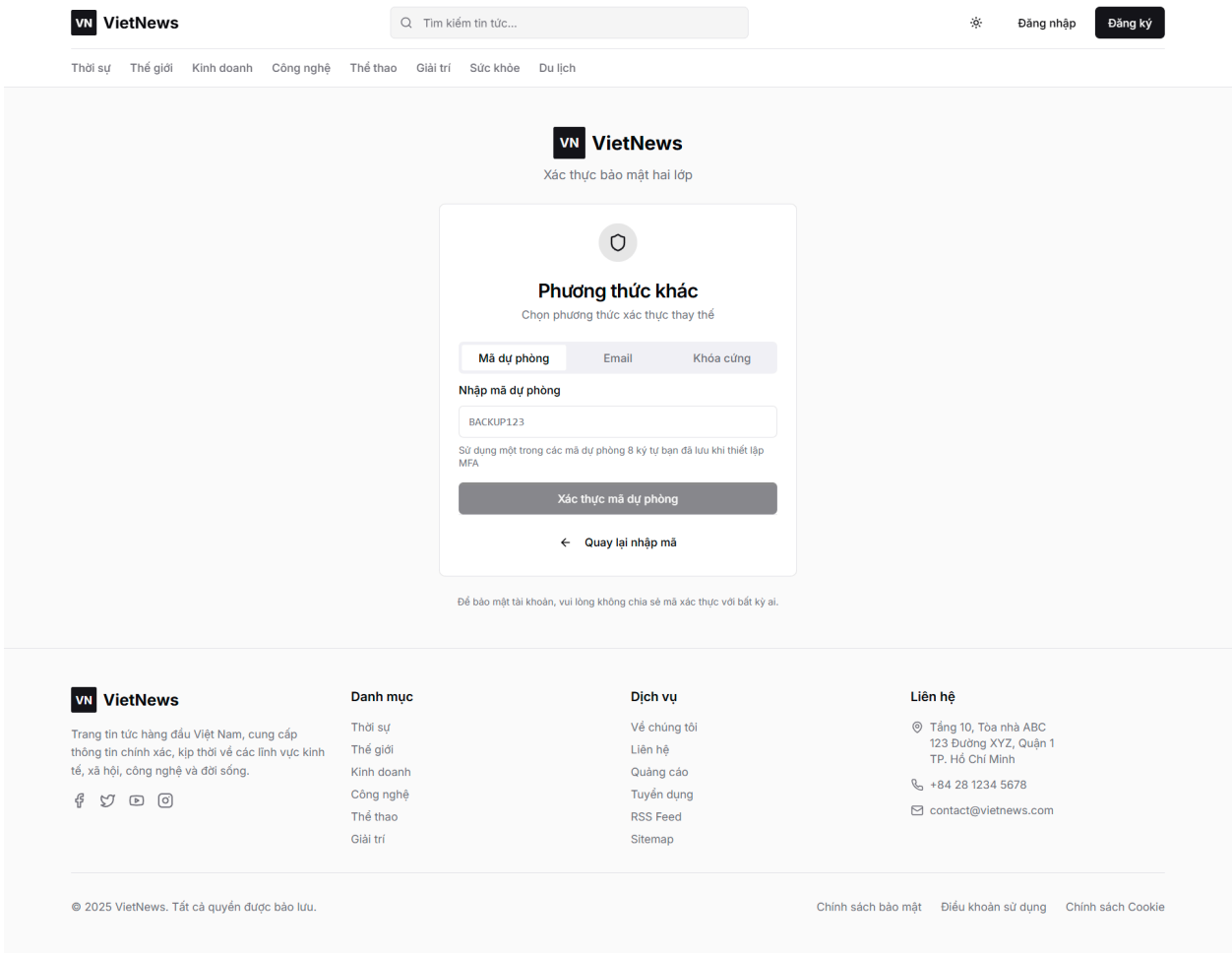
7. Low Fidelity Prototype For Use Case

Prompt for AI tools: Enhance the /test/verify-mfa pages of my Digital Newspaper Website to simulate realistic, step-by-step processes. Use the same layout, design system, and UX style as the rest of the website. Include all typical stages described in the use-case specifications.🔐 /test/verify-mfa – Multi-Factor Authentication FlowImplement a multi-step, interactive MFA verification process:Step 1 – Prompt for Code:Ask the user to enter a 6-digit code from an authenticator app (TOTP).Show fallback links: "Use backup code", "Try a different method".Include a live countdown (e.g., "Code expires in 30s").Step 2 – Handle Invalid Code:Show error on incorrect input with retry counter (e.g., "2 attempts left").After 3 failed attempts, lock verification for 5 minutes.Step 3 – Fallback Options:Let the user choose alternative methods: backup code, email, hardware key.Simulate each with mock inputs and success/failure.Step 4 – Success:Show confirmation: "Verification successful."Auto-redirect to dashboard or show "Continue" button.

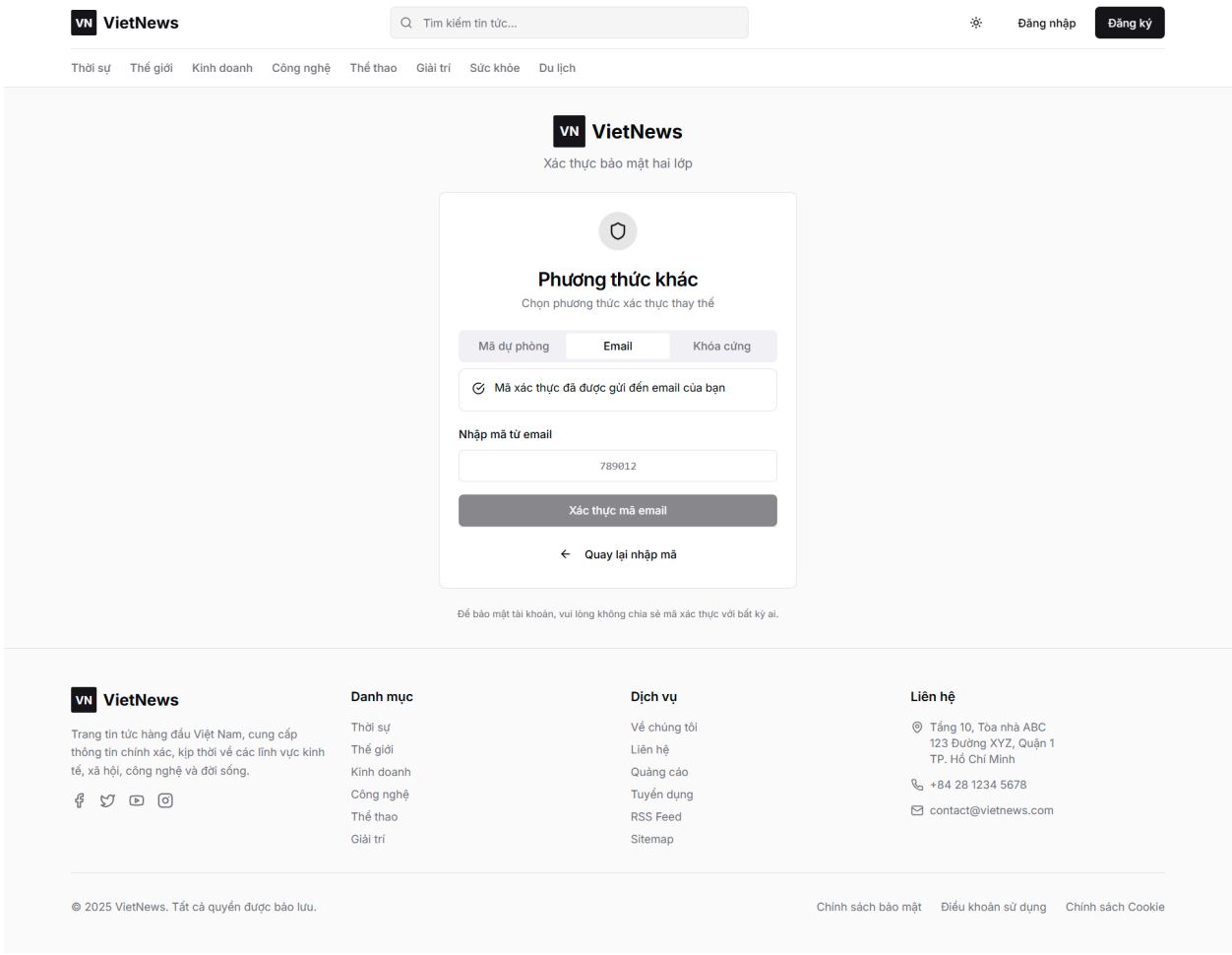
404 Not Found



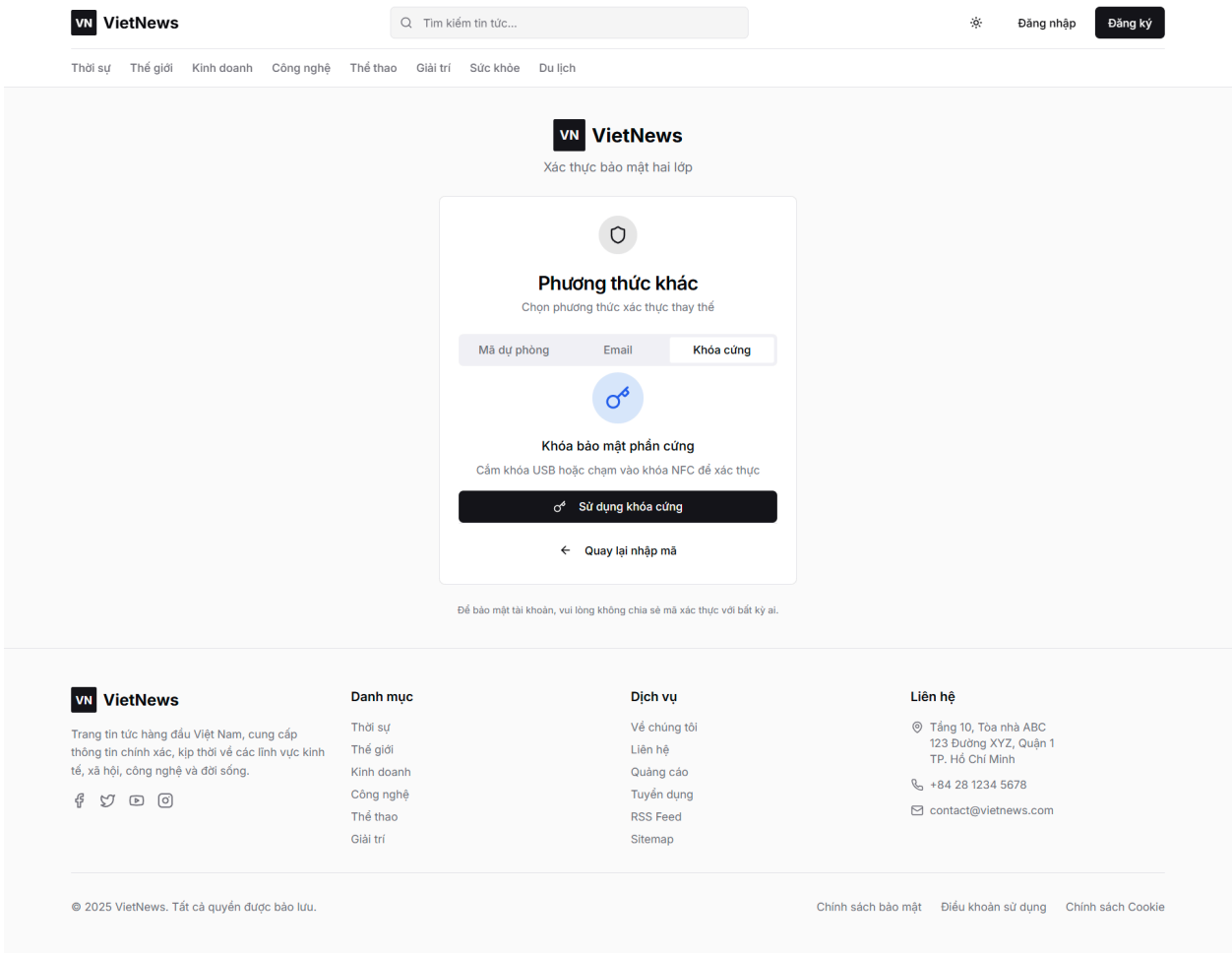
404 Not Found



404 Not Found



404 Not Found



404 Not Found