Vibration-Aided Picture-Based Authentication for Shoulder-Surfing Resistant Mobile Login

Ibrahim Albadi[1], Mahdi Almalki[2], Abdullah Albokhari[3],
Salman Alsumairi[4], Sami Atiyah[5], Faisal Alsubaei[6], Abdullah Abuhussein[7]
Dept. of Cybersecurity-College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia[1, 2, 3, 4, 5, 6]
St Cloud State University, St Cloud, MN 56301, USA[7]

***Abstract*—Text and graphical passwords on smartphones are easy to shoulder-surf in public, and many of the alternatives that have been proposed do not work well on small touchscreens. We describe a vibration-aided picture-password scheme that pairs an image-based credential with hidden haptic prompts. The user selects an image during registration, picks at least three secret locations on it, defines a set of deceiving locations, and maps vibration patterns to specific decoy tap styles. The image, locations, vibration mapping, spatial acceptance regions, and timing thresholds are stored on a server. At login time, the server builds a per-session challenge that interleaves the secret and deceiving locations, assigns vibration patterns only to the decoys, and pushes the challenge to the client. The device displays the image, vibrates before each decoy tap, and records tap coordinates and durations. The server matches the input against the challenge, accepts or rejects accordingly, and regenerates a fresh challenge after any mismatch, so that replay and pattern-learning attempts gain nothing. Because the vibration cue is tactile and not visible from outside, an onlooker cannot tell a genuine tap from a decoy response. The result is better resistance to shoulder-surfing without any extra hardware and without asking the user to learn anything especially complex.**

***Keywords*—Picture-based authentication; vibration cues; shoulder-surfing; mobile security; haptics**

I. Introduction

Smartphones now mediate access to banking, email, and enterprise systems, and authentication on these devices still leans heavily on text passwords, PINs, and conventional graphical passwords. All of these are entered on a visible screen, which makes them vulnerable to shoulder-surfing — an observation-based attack in which an adversary watches the login, directly or through a camera, and reconstructs the credentials [1]. Because phones are used in airplanes, public transport, lecture halls, and cafes, the attack surface travels with the user, and the exposure is repeated at every login, which motivates the vibration-aided extension to picture-based authentication explored in this study [2].

Graphical password systems are broadly categorized as recognition-based, where users later recognize chosen images, and recall-based, where users reproduce drawings or click-points [3]. Recall-based schemes have users click or draw secret points [4], recognition-based schemes have them pick chosen images [5], and later work added on-screen decoys or spatial transformations to dilute what an observer can see [6], [7]. These defenses share two recurring weaknesses: a determined observer who records more than one session eventually defeats purely visual obfuscation, and the added on-screen complexity raises cognitive load and is awkward on small touchscreens [8], [11]. Hardware mitigations, such as privacy films, reduce viewing angles but degrade the screen for the legitimate user and fail against a pre-placed camera [1].

To address these gaps, this work extends a vibration-aided picture-based authentication method [2] that moves the obfuscation off the visual channel and onto a covert haptic one. The user's real credential is a small set of secret locations on a single image, but the device vibrates only before the decoy taps, so an observer cannot tell from the screen which taps belong to the real password. A standard client–server architecture is used: the smartphone displays the image and triggers vibrations, while the server generates per-session randomized challenges and verifies tap locations and durations.

The objective of this study is to design, implement, and empirically evaluate such a scheme on commodity smartphones and to quantify its resistance to shoulder-surfing without sacrificing usability. The contributions are: 1) a system model and data structure for handling secret and deceiving locations with configurable acceptance regions; 2) registration and authentication algorithms that randomize the decoy placement and the vibration patterns on a per-session basis; 3) an environment-aware adaptive authentication phase that uses on-device vision to tell public from private surroundings and adjust the challenge difficulty accordingly; and 4) a cross-platform prototype together with an empirical user study (N = 21) on usability, perceived security, and short-term memorability.

II. Background and Related Work

A. Motivation

Shoulder-surfing is unusual among attacks in that it needs no software exploit — only the user, the screen, and time — which makes it both common and hard to defend with conventional security controls. Empirical studies report that a large fraction of smartphone users have experienced or suspect they have experienced it in everyday life [1], [11]. The two dominant classes of defense each fall short: visual-obfuscation schemes trade usability for resistance and degrade on small screens, while physical mitigations such as privacy films depend on user vigilance and fail against recording devices. This motivates a defense that (1) keeps the familiar interaction of a picture password, (2) hides the discriminating information on a channel a camera cannot capture, and (3) requires no hardware beyond the vibration motor already present in every modern phone. The

scheme presented here is designed around these three motivating requirements.

B. Scope and Focus

This work focuses on image-based authentication for mobile and connected devices under a shoulder-surfing threat model. The primary goal is to reduce the information that visual observers can extract from the login process while keeping the scheme practical on small touchscreens and compatible with existing hardware. Graphical password systems are broadly categorized as recognition-based, where users later recognize chosen images, and recall-based, where users reproduce drawings or click-points; this taxonomy is used to situate the proposed method among prior work [3].

C. Representative Graphical Password Schemes

Graphical password methods fall into two large families: a recognition family, in which the user later picks chosen images out of a set, and a recall family, in which the user reproduces something drawn or clicked at registration time. The progression from one scheme to the next is not really a clean timeline; what changes is the balance between how much the user has to remember, how natural the interaction feels, and how well the scheme holds up to a watcher [3].

The earliest recall-based proposals had users click on a sequence of points on an image [4]. Schemes like Deja Vu moved to abstract image recognition to make credentials easier to remember [5]; these still exposed the credentials in the same way as the earlier proposals: every step of the interaction is on the screen, visible to anyone watching. Visual obfuscation was the next step. The WIW (Where Is What) scheme planted decoy objects on the screen to dilute the real click-points [6], and later proposals applied spatial transformations [7]. The downside, every time, was that the extra on-screen complexity made the interface harder for the user to drive [8]. To summarize the landscape of graphical authentication, Table I provides a comparative analysis of prominent schemes across critical security and usability dimensions.

TABLE I. COMPARISON OF EXISTING GRAPHICAL AND MULTI-MODAL AUTHENTICATION SCHEMES

Scheme	Primary Modality	Shoulder-Surfing Resistance	Hardware Requirements	Cognitive Load	Attack Model Mitigated
Déjà vu [5]	Recognition (visual)	Low	Standard Display	Low	Brute force
WIW [6]	Recall + Decoys	Medium	Standard Display	High	Casual Observation
Anti-Shoulder Surfing [7]	Transformation (Visual)	Medium	Standard Display	High	Casual Observation
Feel Vibration [9]	Haptic Challenge	High	Vibration Motor	Medium	Shoulder-Surfing
Proposed Work	Recall + Haptic Decoys	High	Standard Touchscreen + Motor	Low	Shoulder-Surfing / Recording

D. Positioning the Proposed Work

Table I lays out the tradeoff. Defenses that live entirely on the screen tend to fall into one of two failure modes: a serious observer eventually defeats them, or the user has to work too hard to make them practical. We try to sidestep both by pushing the obfuscation off the visual channel and onto a covert haptic one. The resistance to shoulder-surfing is closer to what multi-modal schemes deliver, and the demands on the user and the hardware stay close to those of the earlier picture-based methods.

E. Shoulder-Surfing

Shoulder-surfing is an observation-based attack: the adversary watches the user as they log in and tries to reconstruct the credential, whether that credential is a PIN [1], a typed password, or a graphical input. The watching can be direct (standing close to the user) or mediated (a camera, a reflection, or a recording from across the room). Because smartphones now travel everywhere with their owner — airports, trains, lecture theatres, coffee shops — the attack surface follows them, and the resulting tradeoff between making a login fast for the user and making it opaque to an observer has become a hard problem in its own right.

Shoulder-surfing does not need any technical exploit. It does not touch the protocol stack and it does not require a bug in the client. What it needs is the user, the screen, and time. Empirical work on PIN entry and Android pattern unlock shows how exposed those interactions are: both are short, repeated, and visually consistent across sessions, which is exactly the combination an observer wants [10]. Graphical schemes inherit the same problem whenever the user's click-points or drawing path stay the same from one login to the next, because every additional login leaks a little more of the spatial pattern.

Early studies on graphical passwords, such as the work of Ian Jermyn et al. (1999) [4], set out the security-versus-usability tradeoff in graphical passwords in formal terms. A few years later, Man et al. (2003) explicitly designed against an on-looker in their WIW scheme [6]. Work on mobile authentication has kept returning to shoulder-surfing as a practical threat ever since, especially for phones used in shared or public spaces.

F. Existing Shoulder-Surfing Countermeasures

The mitigations that have been proposed against shoulder-surfing on mobile devices group fairly naturally into three buckets: techniques that try to disguise what is happening on the screen, accessories and habits that try to keep the screen private in the first place, and schemes that pull a second, non-visual channel into the login.

1) Visual obfuscation techniques: Visual obfuscation works by changing what is on the screen so that a watcher cannot easily reconstruct the input. The RandomPad work by Maiti et al. [8] is a useful illustration of how many forms this can take: keypads with their digits shuffled each time, grids that move, click-points that disappear after use, and text-and-image hybrids. The common thread is that the on-screen layout is not stable from one session to the next, so memorizing it does not get the observer very far. The cost, as Maiti et al. note [8], is

paid by the user. The interface is now harder to read and harder to drive, and on a small phone screen, where there is already not much room to work, that overhead bites.

2) Physical protection methods: Physical mitigations either add something to the device or modify the environment around it. The Bhardwaj et al. review collects most of the usual suspects [1]: privacy films that cut viewing angles down, anti-glare overlays, and the very low-tech option of just shielding the screen with the free hand. None of this is expensive, but every one of these mitigations depends on the user thinking about the threat at the right moment, and on the particular geometry of the room. The review also calls out the failure mode that matters most for our setting: a high-quality camera placed in advance defeats all of them [1]. The 2024 review by Por et al. [11] arrives at much the same place. Across the schemes proposed between 2014 and 2024, gains in observation-resistance are typically bought with usability, and the literature has very little evaluation against attackers who watch more than one session in a realistic public setting.

3) Multi-modal authentication: The third type of mitigation introduces a second modality, usually audio or touch. Haptics matter here because a vibration is felt, not seen: a camera across the room or a person sitting next to the user cannot pick it up [9]. The body of work in this area has expanded well past the early prototypes by Luo et al. [9] and Freeman et al. [12]. In 2023, Cao et al. presented a continuous-authentication scheme, which uses passive vibration responses from hand biometrics for continuous authentication on mobile devices [13]. Liu et al. (2024) took a different angle in *HandKey*, where the vibration signature produced by a physical knock becomes the unlock token, and they show that an attacker cannot easily reproduce that signature from visual observation alone [14]. What is missing in this body of work, and where our scheme sits, is a combination of per-session randomized haptic decoys with a single static image. That combination is the central piece of what we propose here.

G. Proposed Work and its Advantages

This work builds upon prior graphical authentication research by introducing a vibration-aided picture-based authentication system inspired by the patented method of Faisal Alsubaei and Abdullah Abuhussein (U.S. Patent 12,393,663 B1, 2025).

The proposed system integrates secret locations (true password points) and deceiving locations (decoy points) within a single image. Covert vibration cues are assigned only to decoy taps, while secret taps are entered silently. In addition, the system generates a randomized challenge for each authentication session.

During authentication, the image appears visually identical for both secret and decoy interactions. Only decoy positions trigger vibration patterns, and the legitimate user responds with predefined tap styles such as short, long, or multi-tap based on the vibration received. Both the spatial coordinates and the timing of each tap are checked on the server before any access is granted.

A few things follow from this design. Since the vibration cannot be seen, someone watching the screen cannot use the tap responses to separate real taps from decoy ones. The amplitude is kept low on purpose so the buzz reaches the hand that is holding the phone and not the ears of the person sitting nearby. Hardware-wise, nothing exotic is needed; every current smartphone already includes a vibration motor strong enough to drive the prototype. Re-randomizing both the decoy layout and the vibration assignment on every session limits what an attacker gains from recording one good login — the recording is out of date the moment it is captured. The memory burden on the user stays small because they only have to keep a handful of image locations and one short vibration-to-tap mapping in mind. Finally, because the architecture is just a client and a server, the scheme drops into existing mobile back-ends without any infrastructure changes.

Spatial constraints, timing checks, and the covert haptic channel together push the scheme toward genuine shoulder-surfing resistance without giving up usability on ordinary phones.

III. System Design and Architecture

This section walks through the design. The architecture is a fairly conventional client–server split: a phone shows the image, captures the taps, and produces the haptic feedback, while a backend service holds the user record and decides whether a given login attempt succeeds. The subsections that follow describe the data structures used to represent secret and decoy locations, the registration and authentication exchanges between client and server, and the algorithm that builds a fresh randomized challenge for every session. The last subsection covers what the deployment assumes from the hardware: a touchscreen, a working vibration motor, and a network path to the server.

A. System Scope and Requirements

At a high level, the system is a client–server authenticator built on top of a picture password. Decoy taps are accompanied by a vibration cue that only the user can feel; the real password taps are entered in silence. The functional side of the spec is straightforward. The server must let a user register an image and an ordered list of secret locations, accept a chosen set of decoys, take the user's mapping from each vibration pattern to a tap style, build a fresh challenge with placement and vibration arrays each session, and check the resulting tap coordinates and durations against that challenge — falling back to a re-challenge, or a lockout, when the input is wrong. On the non-functional side, the design has to resist observation by an onlooker (this is what the per-session randomization buys), it has to be comfortable on small touchscreens, and it has to run on the kind of haptic-capable hardware most users already own.

B. Data Model and Assumptions

A user record on the server carries a few things: the registered image, the ordered list of secret locations, the chosen decoy locations, the mapping from each vibration pattern to its required tap style, and a small set of configuration parameters such as the acceptance region size and the timing windows for short and long taps. Per-session state is kept in three short-lived arrays — placement, vibration patterns, and the combined

challenge sequence — and is discarded once the attempt resolves. Three assumptions underpin the design: the client device has a vibration motor, every password includes at least three secret locations, and the number of decoys is chosen relative to the number of secrets so that the scheme stays usable without becoming guessable. Vibration cues are, by construction, attached only to decoys; a secret location never vibrates.

C. Password Space and Complexity Analysis

To evaluate the theoretical strength of the proposed system, the password space is derived from three independent dimensions: the ordered spatial selection of at least n $\geqslant$ 3 secret locations, the ordered selection of at least k $\geqslant$ 3 decoy locations from the same image, and the mapping of each decoy to one of t = 3 vibration-tap style pairs (short, long, or multi-tap).

For a screen of resolution W × H pixels and acceptance radius r, the number of non-overlapping tappable regions P is [see Eq. (1)]:

$$P = \left\lfloor \frac{W}{2r} \right\rfloor \times \left\lfloor \frac{H}{2r} \right\rfloor \quad (1)$$

For the prototype ($W = 1080, H = 1920, r = 20$ px), this yields $P = 27 \times 48 = 1{,}296$ regions. The total password space S is then [see Eq. (2)]:

$$S = \frac{P!}{(P-n)!} \times \frac{P!}{(P-k)!} \times t^k \quad (2)$$

For the minimum configuration ($n = k = 3, t = 3$) [see Eq. (3)]:

$$S \approx (1296 \times 1295 \times 1294)^2 \times 27 \approx 1.27 \times 10^{20} \quad (3)$$

The Shannon Guessing Entropy [see Eq. (4)]:

$$H = \log_2(S) = \log_2(1.27 \times 10^{20}) \approx 66.8 \text{ bits} \quad (4)$$

This substantially exceeds an 8-character alphanumeric password (~52.4 bits) and surpasses a 10-character alphanumeric password (~65.5 bits), while growing rapidly as users register more locations beyond the minimum of three.

The keyspace S in Eq. (3) covers the tap-sequence factor, i.e., what the user knows. On top of that, the scheme uses the registered image itself as a second, independent factor that the user has to bring. At registration time, the client computes a SHA-256 digest [15] over a canonically normalized version of the image (resized to a 1024 × 1024 RGBA buffer with PNG encoding) and stores only the digest on the server. At login, the device re-normalizes and re-hashes locally and compares the result byte-for-byte against the stored value before any tap is verified. So an adversary who has fully observed the tap sequence still cannot authenticate without producing a file whose normalized digest collides with the stored 256-bit value; the preimage work is on the order of $2^{256} \approx 1.16 \times 10^{77}$. There is no second device involved, no SMS code, and no authenticator app; both factors are settled in the same login interaction. An adversary with no prior knowledge of either the image or the tap sequence faces work on the order of $S \times 2^{256} \approx 1.5 \times 10^{97}$.

Table II summarizes the quantitative security and usability comparison between the proposed system and two widely used authentication schemes.

TABLE II. QUANTITATIVE SECURITY AND USABILITY COMPARISON WITH WIDELY USED AUTHENTICATION SCHEMES.

Scheme	Password space	Entropy (bits)	Avg. Login Time	Shoulder-Surfing Attack Rate
Android Pattern Lock	389,112	18.6	~2s	~95%
Text Password (8-char, printable ASCII)	6.1×10^{15}	52.4	~5s	~80%
Proposed Work	1.27×10^{20}	66.8	~8-10s	~1.71×10^{-11} (theoretical, single-session bound; Eq.7)

D. Security Analysis

1) Threat model: We consider three adversaries, ordered roughly by how close they sit to the actual login:

T1 — Online brute-force attacker. This adversary has not watched any prior login. They sit at the login interface and try sequences, with no signal between attempts beyond accept or reject.

T2 — Active replay attacker. This adversary has captured a complete authentication session for the target account and assumes the credentials will look the same next time. They replay the captured tap sequence on a later login attempt.

T3 — Passive visual observer. This adversary watches the device screen, either directly from nearby or through a hidden camera, during one or more login sessions. They can see every one of the n + k tap positions and roughly how long each tap was held, but they have no access to the server, no access to the device internals, and no way to perceive the haptic channel.

Three threats sit outside the model we defend against: code that has already compromised the client device, a fully compromised server, and accelerometer-based side-channel inference of tap timing.

2) Resistance to brute-force attacks (T1): Section III C gave the minimum key space as $S \approx 1.27 \times 10^{20}$ [Eq. (3)]. For an attacker guessing uniformly at random, a single attempt lands on the right sequence with probability [see Eq. (5)]:

$$P_{\text{guess}} = \frac{1}{S} \approx 7.87 \times 10^{-21} \quad (5)$$

The server locks the account after $L = 5$ failed attempts. The probability that any of those L tries succeeds is [see Eq. (6)]:

$$P_{\text{brute}} = \frac{L}{S} = \frac{5}{1.27\times10^{20}} \approx 3.94 \times 10^{-20} \quad (6)$$

What does most of the heavy lifting here is that the server throws the challenge away after each failed attempt and builds a fresh randomized X. Anything the attacker picked up from the previous attempt, say that a particular region of the screen was wrong, carries no information into the next challenge; the decoy layout and the vibration assignments are independent draws each time. Strategies that incrementally narrow down a password, such as dictionary attacks or pattern enumeration, simply do not gain ground here, even though they remain effective against static text passwords.

3) Resistance to replay attacks (T2): On every login, the server picks fresh positions for the k decoys from the P available regions and assigns each decoy a fresh vibration pattern from the $t = 3$ available styles. The probability that a recorded session X_i is still a valid input for the very next session $X\_\{i+1\}$ is [see Eq. (7)]:

$$P_{\text{replay}} = \frac{1}{(1296\times1295\times1294)\times27} \approx 1.71 \times 10^{-11} \quad (7)$$

Because the decoy positions and vibration-tap mappings are sampled independently per session, even a flawless recording of session i is no help for session $i+1$. This bind-to-the-session property is something neither Android pattern unlock nor a static text password has; in either of those, what the attacker is trying to recover does not move from one attempt to the next.

4) Resistance to observation attacks (T3): What a passive observer sees during one full authentication session is a string of n + k taps, all of them landing on the same image and looking identical on the screen. From that view, the observer cannot recover any of the following:

- Which taps are secret and which are decoys, since the client UI applies no visual distinction — every tap appears identical on screen.
- What vibration pattern the device holder received, since haptic signals are tactile and cannot be captured by cameras or distant recording equipment.
- What tap-style response each decoy required, since the short, long, or multi-tap response is privately driven by the vibration cue received only by the device holder.

Even when the observer captures all n + k tap coordinates exactly, those coordinates are stale by the time the next session begins, because the decoy positions and vibration patterns have been resampled. The observation attack therefore collapses into the replay case already analyzed, with the same single-session bound $P\_replay \approx 1.71 \times 10^{-11}$ from Eq. (7).

The bound applies to a single session. Because all taps — secrets and decoys alike — appear identical on the screen (the vibration cue that distinguishes them is haptic, not visible), an observer cannot separate secrets from decoys by visual appearance or response style. However, both the secret and deceiving locations are fixed at registration; an attacker who records multiple clean sessions of the same user can, in principle, identify the secrets by statistical intersection, since the tap positions recur identically across sessions while their presentation order varies. With k = 3 decoys registered among a set of 6 tap positions (3 secrets + 3 decoys), roughly three to four complete, cleanly captured sessions are sufficient for an attacker to isolate the secret positions by identifying which positions appear with consistent behavior across sessions. This is the well-known hot-spot limit of any fixed recall-based credential, including PINs and conventional picture passwords [10], which leak from a single clean observation; the covert haptic channel does not eliminate this limit, but it does ensure that every tap looks identical on screen, raising the practical difficulty of capturing clean sessions and synchronizing them to a single user. It is mitigated concretely by rotating the secret locations after a fixed number of logins R held below this threshold. Alternatively, the scheme's architecture permits registering up to 10 secret and 10 deceiving locations, which would substantially increase the inference threshold, combined with the public-environment HIGH_SECURITY phase of Section III K. Full evaluation of this rotation policy is left as future work. Table III summarizes the single-session success probability and the primary mitigation for each of the three adversaries (T1, T2, and T3).

TABLE III. SINGLE-SESSION ATTACK RESISTANCE OF THE PROPOSED AUTHENTICATION SYSTEM.

Attack Type	Adversary	Single-Session Success Probability	Primary Mitigation
Brute-Force	T1	~3.94×10^{-20}	Large password space (Eq.3) + 5-attempt lockout
Replay	T2	~1.71×10^{-11}	Per-session challenge regeneration
Observation /Shoulder-Surfing	T3	~1.71×10^{-11}	Covert haptic channel + session randomization

E. System Calibration and Boundary Cases

Putting this scheme into production needs some thought about the spatial and temporal parameters, because both directly affect how secure the scheme is and how it feels to use. The acceptance radius R_n can be set centrally by an administrator or chosen by the user at registration time. A larger R_n is more forgiving of imprecise finger placement but exposes a slightly larger target to a random guesser; a smaller Rn does the opposite. The threshold that separates a short tap from a long tap — a 300 ms long-press threshold in the prototype — can be relaxed or tightened to suit either the user or the environment.

Edge cases need their own handling. A tap can land right on the boundary of R_n, and a tap duration can sit right at the threshold limit. The implementation keeps a small tolerance band for these cases and lets the temporal parameters vary by up to **20%** to absorb both natural human variation and the device's own sampling jitter. Anything outside the adjusted spatial or temporal limits is treated as a hard failure: the attempt is rejected, and the server immediately rebuilds a fresh X, which prevents an attacker from sitting next to the boundary and probing it.

F. Core Modules and Algorithms

Five logical modules carry the work: a Registration Manager, a Challenge Generator, a Vibration Cue Encoder, a Verifier, and a Session and Rate-Limit Controller. At registration, the Registration Manager writes the image, the secret locations SLi, the chosen decoys DLk, and the vibration mapping into the user record. At login, the Challenge Generator assembles the placement array M (where the decoys sit relative to the secrets), fills the vibration-pattern array V for those decoys, and combines them into the per-session sequence X. The client-side Vibration Cue Encoder reads X, fires the vibration before each decoy tap, and ships the captured tap coordinates and durations back to the server. The Verifier then checks that

each tap landed inside its acceptance region and that the decoy taps were timed against the expected vibration pattern. The Session and Rate-Limit Controller, finally, handles retries and lockouts. Fig. 1 shows the overall system architecture, with the client, the server, and the vision service used for environment-aware classification.

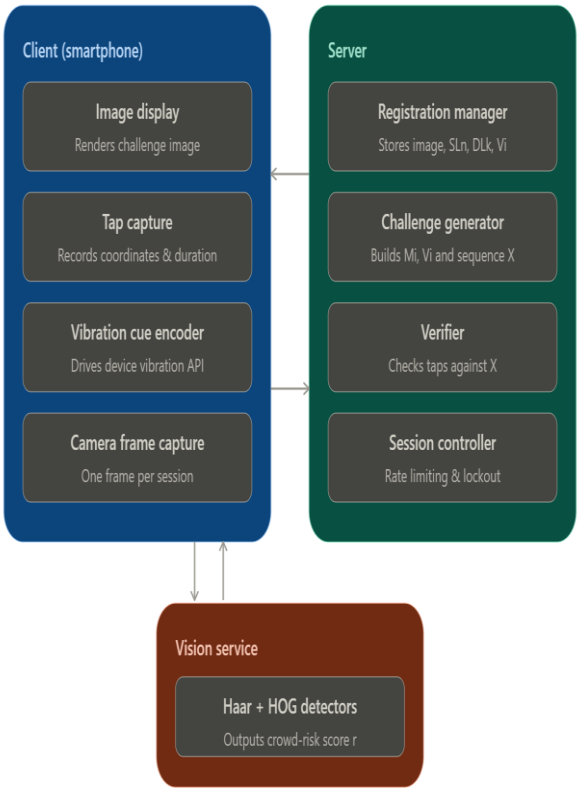

Fig. 1. System architecture showing the client (image display, tap capture, vibration cue encoder, and camera frame capture), the server (registration manager, challenge generator, verifier, and session controller), and the vision service used for environment-aware classification.

G. Client User Interface Behavior

The client user interface displays a single image that fills most of the screen and accepts precise tap input from the user, as shown in Fig. 2.

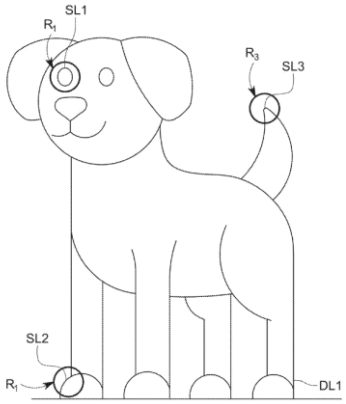

Fig. 2. Client user interface displaying an authentication image with secret locations (SL), deceiving locations (DL), and their respective acceptance regions [2].

It does not reveal any visual indication that would distinguish secret from decoy positions; every step of the sequence appears as a normal tap on the same image. Before positions corresponding to deceiving locations, the device vibrates according to the assigned pattern, prompting the user to respond with the matching tap style, such as short, long, or multi-tap, while secret positions are entered without vibration. Fig. 3 shows an example sequence of secret and deceiving locations with the vibration pattern and tap response for each.

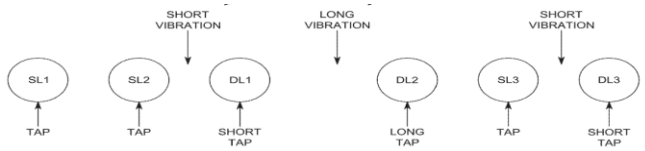

Fig. 3. Sequence of secret and deceiving locations with their corresponding vibration patterns and tap responses.

This behavior aims to prevent visual observers from inferring which taps belong to the true password, since all taps look identical on the screen.

H. Hardware and Computing Environment

The client device needs three basic capabilities: a touchscreen to record tap positions and durations, a vibration motor to provide haptic feedback, and a network connection to reach the server. Wearables or game controllers with vibration can also be used if they can show the image or an equivalent interaction surface. Fig. 4 gives a schematic view of the server and its computing environment.

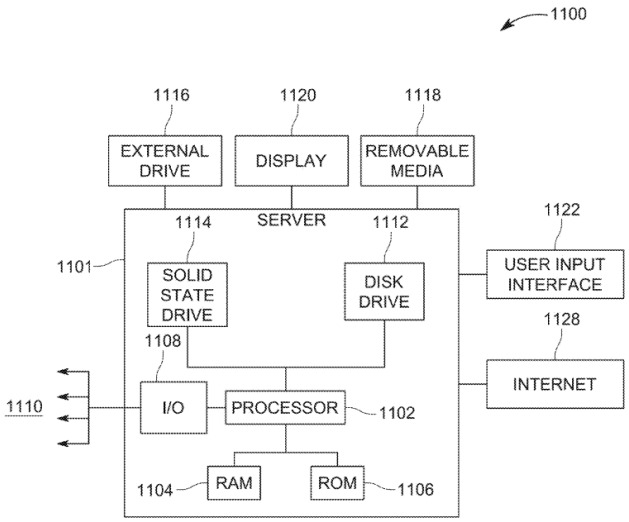

Fig. 4. Schematic diagram of the server and computing environment [2].

The server runs on a standard computing device equipped with a central processor (CPU), random access memory (RAM), read-only memory (ROM), persistent storage (such as solid-state or hard disk drives), input/output (I/O) circuitry, and a network interface for internet connectivity.

I. System Workflow, Components, Input–Process–Output, and Compatibility

1) System workflow: Rather than relying on static interactions, the system's workflow executes a dynamic

challenge-response mechanism between the client device and the server. During the authentication phase, the server dynamically generates a new, randomized sequence of X for each login session. This sequence interleaves n secret locations (SL_n) with k deceiving locations (DL_k). The server also generates a random array of vibration patterns (V_i) that are exclusively mapped to the deceiving locations. The client device executes this workflow by displaying the image and triggering the covert vibration patterns only before a required decoy tap. Fig. 5 shows the full workflow of one authentication session, including the reject-and-regenerate path taken after a mismatch.

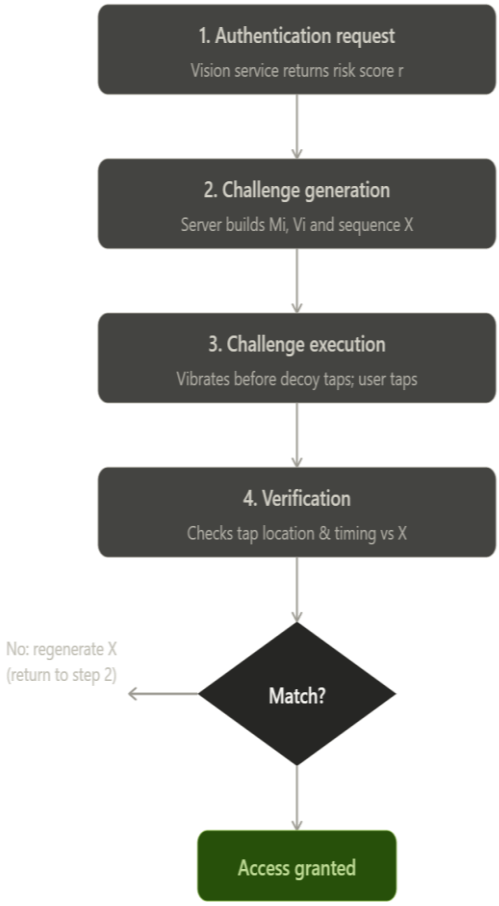

Fig. 5. Workflow of one authentication session, from the environment-aware check through challenge generation, challenge execution with vibration cues, and verification, including the reject-and-regenerate path.

2) Input–Process–Output (IPO) model: The system logic follows a strict IPO model to validate the user's identity without exposing the credentials visually:

- Input**:** The system captures multi-modal user interactions, specifically the spatial coordinates of the taps on the screen and the temporal tapping styles (e.g., short, long, or multiple taps).
- Process: The server's processor evaluates these inputs by comparing the received tap locations and durations against the session's specific generated sequence X.
- Output: The system outputs a binary access decision; it grants access to the server if the inputs perfectly match the sequence, or it denies access and regenerates a new random array and sequence X to offer a new login attempt. Fig. 6 presents this input–process–output cycle as a flowchart of the steps performed on the client device.

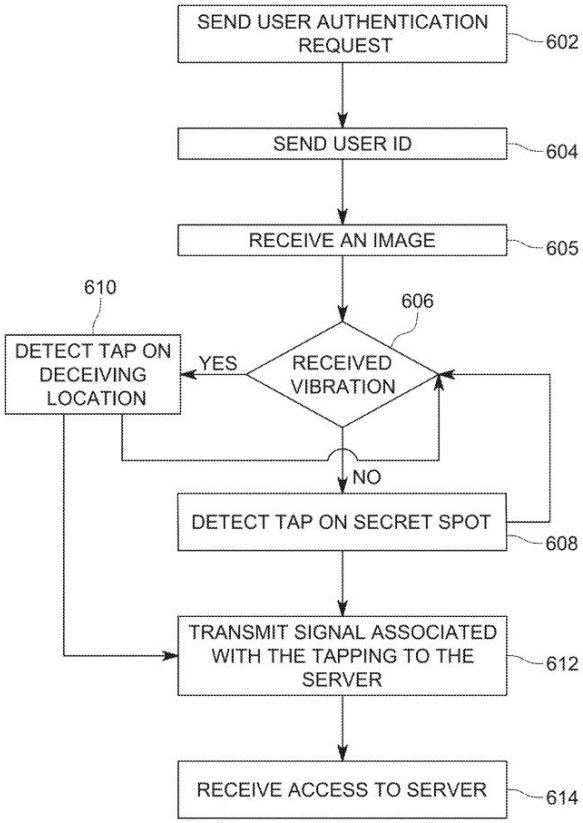

Fig. 6. Flowchart of the authentication method performed by the client device [2].

3) Component integration and compatibility: The client only relies on hardware that already exists in the device, namely the built-in vibration motor, so an ordinary phone, a game controller, or a smartwatch can each play the client role. Nothing has to be added to the device, which keeps the deployment cost down and means the scheme can sit on top of existing consumer or enterprise mobile setups with no retrofit work.

J. Prototype Implementation

We built a working prototype of the scheme and tried it on off-the-shelf phones to check that the design holds together. The client is a React Native app written in TypeScript, using Expo for the SDK and Expo Router for navigation. The benefit of going this route is portability: the same codebase runs on both Android and iOS without any platform-specific fork. Fig. 7 shows the application home screen, where the user selects Login or Register. Fig. 8 shows the registration screen, where the user selects an image and defines the secret locations.

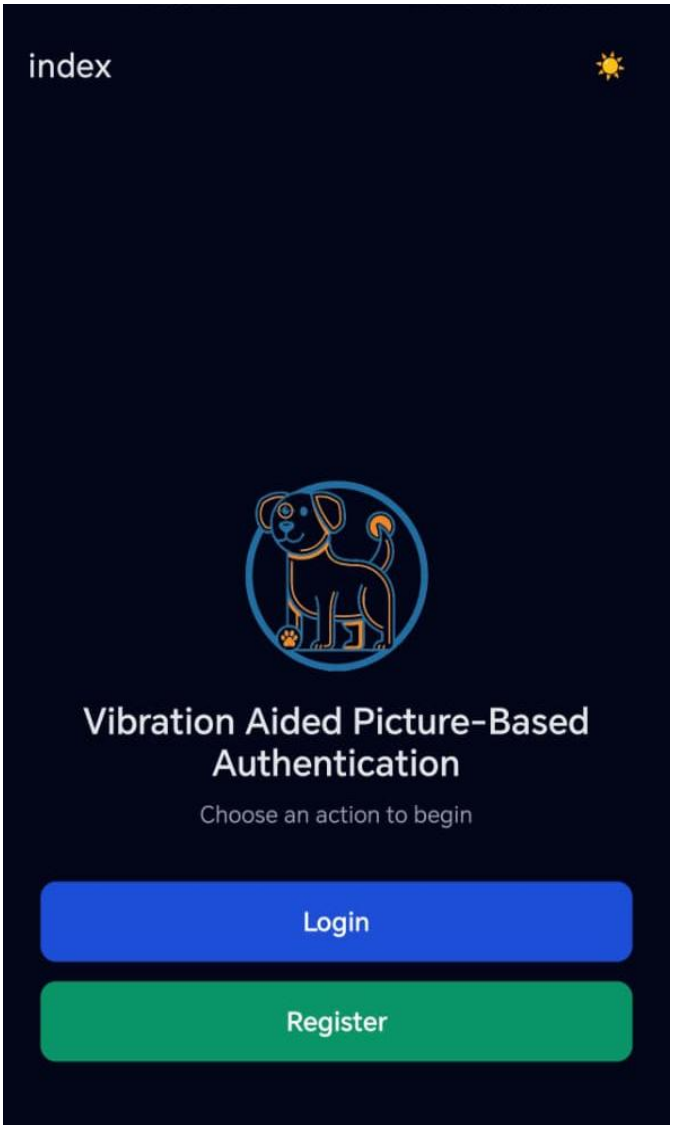

Fig. 7. Application home screen showing login and register options.

The backend user credential store is implemented using Google Firebase Firestore, a cloud-hosted NoSQL database, which persists with each registered user's username, selected image identifier, secret locations, deceiving locations, and vibration-pattern mappings as structured documents. Image selection is handled through the expo-image-picker library, which also supports user-supplied custom images; Custom images are identified during authentication via a SHA-256 digest [15] computed over the canonically normalized image (resized to 1024 × 1024 RGBA and PNG-encoded with deterministic settings) at registration time, and verified at login by recomputing the digest on the device and matching it byte-for-byte against the stored value before any tap input is accepted. Vibration feedback is delivered through React Native's built-in Vibration API (Vibration.vibrate()), which accepts duration arrays of the form [delay, on, off, on, ...] on Android, enabling the three distinct haptic patterns used in the system — single (one 400 ms pulse), double (two 400 ms pulses separated by a 400 ms gap), and triple (three 400 ms pulses). On iOS, which does not natively support pattern arrays, the same three patterns are simulated through repeated timed calls to Vibration.vibrate(400). To overcome an Android-specific cold-start latency in the vibration motor, the implementation primes the motor with an imperceptible 1 ms pulse on the first user gesture before playing any authentication-related pattern. Fig. 9 shows the login screen while the environment security scan is running.

Tap interaction is captured via React Native's Pressable and GestureResponderEvent interfaces, with an acceptance radius of 20 pixels, a double-tap detection window of 200 ms, and a long-press threshold of 300 ms. The per-session randomized challenge sequence is generated on the client in the prototype using the generatePresentationOrder function, which randomly assigns the six tap slots across the three secret and three deceiving identities while preserving the canonical ordering of each group. Fig. 9 also shows the Dev mode of the prototype, which includes all data related to the adaptive Environment-Aware component that will be discussed in the next subsection. This is optional for development, but can also be useful for normal users. Fig. 10 shows the login screen after the environment is confirmed, with the image challenge presented for tap entry.

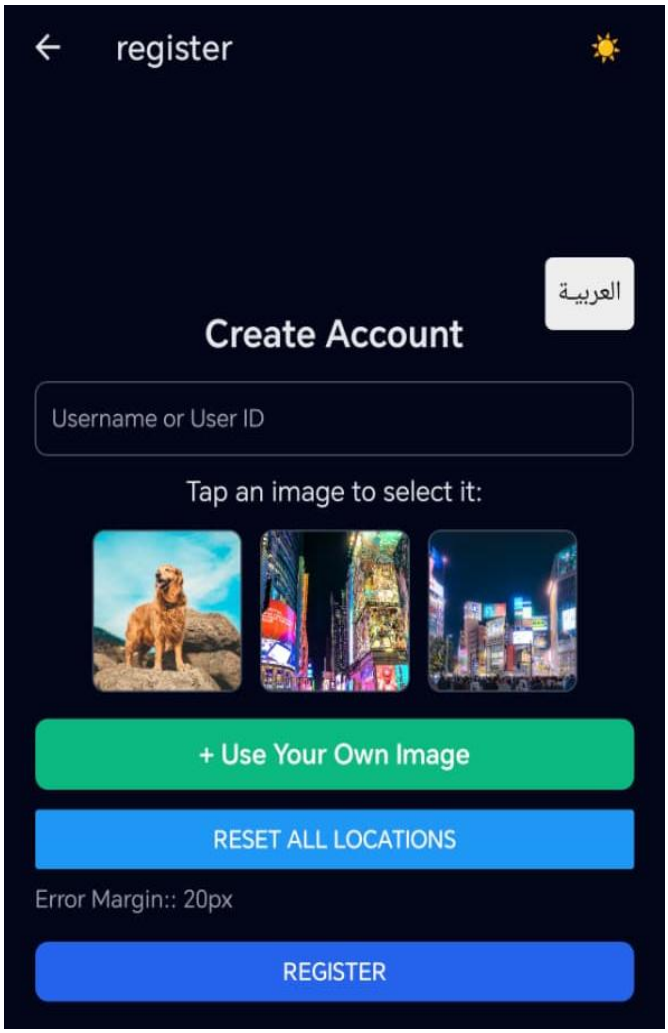

Fig. 8. Registration screen allowing users to select an image and define secret locations.

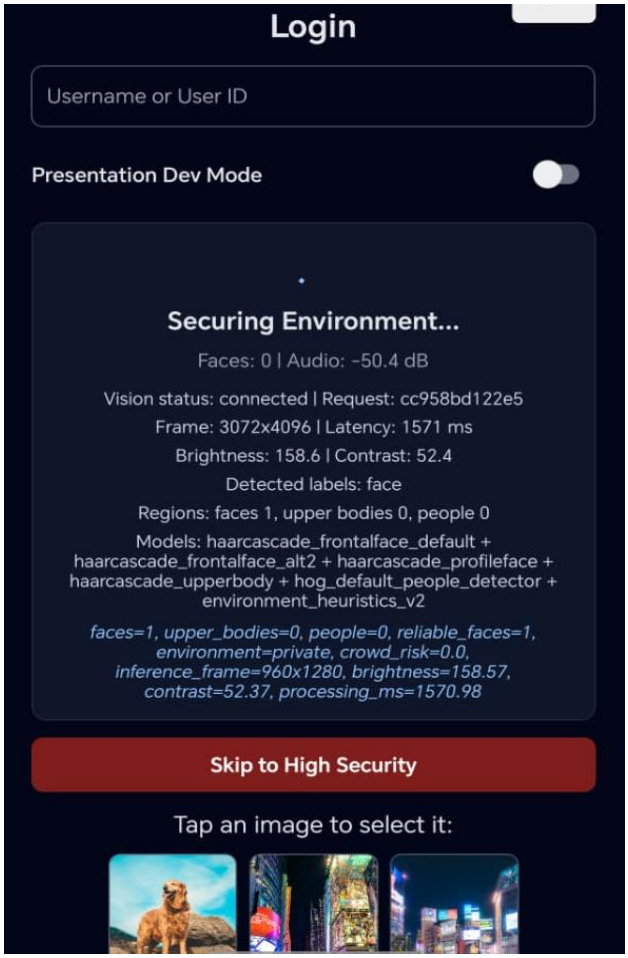

Fig. 9. Login screen during environment security scan.

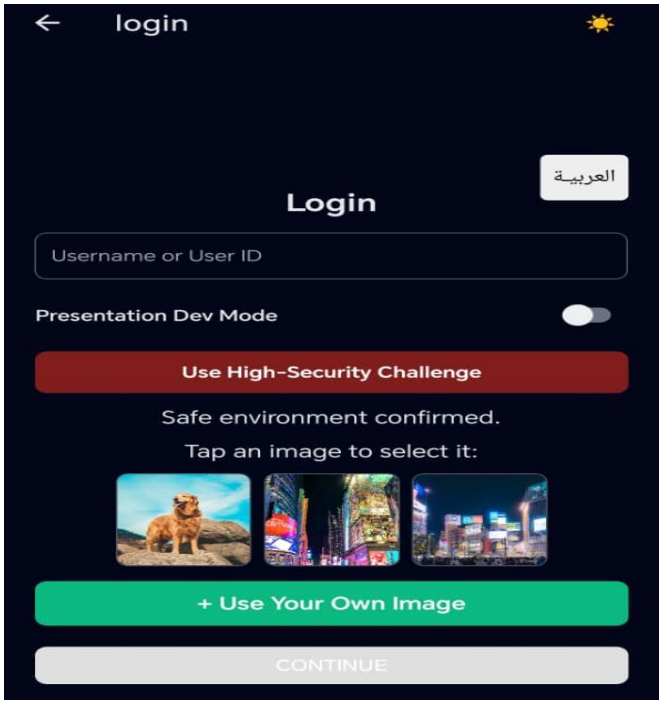

Fig. 10. Login screen after environment confirmed — image challenge selection.

K. Adaptive Authentication Phase (Environment-Aware)

Because the threat from shoulder-surfing depends heavily on where the user actually is, the prototype includes an adaptive authentication phase that adjusts the difficulty of the challenge to the user's surroundings. Before a challenge is generated, the client grabs a single frame from the front camera and sends it to a vision service written in Python on top of FastAPI and OpenCV. The service runs two detectors in parallel that look for different visual cues: a Haar-cascade face detector based on the Viola–Jones framework [16], using both frontal and profile cascades, and a Histogram of Oriented Gradients (HOG) pedestrian detector in the Dalal and Triggs formulation [17]. Three signals come out of that frame: the number of detected faces, the number of detected upper-body shapes, and the distribution of detection confidences. The service aggregates the three signals into a scalar crowd-risk score $r \in [0,1]$ using a normalized weighted sum:

$$r = min\left(1, \frac{\alpha \cdot f}{N_{max}} + \frac{\beta \cdot b}{N_{max}} + \gamma \cdot c\right)$$

where, f is the face count from the Haar-cascade detector, b is the upper-body count from the HOG detector, $\bar{c} \in [0,1]$ is the mean of the confidence-score distribution across all detections in the frame used as a compact summary of detector certainty $N_{\max} = 5$ is a saturation cap above which the environment is unconditionally treated as crowded, and $\alpha = 0.5$, $\beta = 0.3$, $\gamma = 0.2$ are empirically set weights satisfying $\alpha + \beta + \gamma = 1$, with face detections weighted most heavily as the closest-proximity risk signal. The result is clipped to $[0, 1]$ by the outer min. The service returns only r and the binary label PRIVATE or PUBLIC, and the frame is discarded immediately after scoring.

The frame is not written to disk on the client and is not written to disk on the server.

Above r = 0.55, the environment is labeled PUBLIC; below that, the label is PRIVATE. A PUBLIC label puts the client into the HIGH_SECURITY phase, which uses the full n + k decoy challenge from Section III. A PRIVATE label switches to a LOW_SECURITY phase in which the decoy count k drops to a default of 1, so that the login stays quick in settings that do not really need to go through all the steps. The user can override the automatic classification at any time with a single "Use High-Security Challenge" control. The net effect is that the full strength of the scheme stays available where it is needed, but the user is not paying for that full cost during routine logins at home or in the office.

IV. Evaluation

A. Study Design

Twenty-one participants took part in a study designed to probe three things about the prototype from Section III: how usable it is, how secure it feels, and how much of the credential survives a short retention interval. Recruitment was by convenience sampling within the university and through the authors' own contacts, and the resulting sample spans the age, gender, smartphone-proficiency, and cybersecurity-background categories, as shown in Table IV. One number worth pulling out up front: 20 of the 21 participants (95.2%) told us they had been

on the receiving end of shoulder-surfing in real life, or thought they might have been. The threat model is not abstract for this group. Everyone got a briefing before they started, consented in writing, and could stop at any point. We kept no personally identifying information, no biometric data, and no camera frames; only anonymized Likert responses and aggregate timing statistics ended up on disk.

TABLE IV. PARTICIPANT DEMOGRAPHICS (N = 21)

Characteristic	Category	N (%)
Age group	18–24 / 25–34 / 35–44 / 45–54 / 55+	6 / 4 / 5 / 5 / 1
Gender	Male / Female / Prefer not to say	11 / 9 / 1
Smartphone proficiency	Beginner / Intermediate / Advanced / Expert	2 / 8 / 8 / 3
Cybersecurity knowledge	Beginner / Intermediate / Advanced / Expert	9 / 5 / 6 / 1
Prior shoulder-surfing exposure	Yes / Maybe / No	15 / 5 / 1

Each session had three phases: (1) orientation, in which participants received a brief explanation and demonstration; (2) registration, during which each participant selected an image, assigned three secret locations, three decoy locations, and mapped vibration patterns to each decoy; and (3) authentication, in which participants performed repeated login attempts until self-reporting comfort. Participants then completed an online questionnaire (Microsoft Forms) covering five instruments: registration usability (5 items), login usability (8 items), perceived security (6 items), memorability and cognitive load (6 items), and the 10-item System Usability Scale (SUS) [18], scored as (sum of adjusted responses) × 2.5 on a 0–100 scale. All Likert items used a 5-point scale (1 = Strongly Disagree to 5 = Strongly Agree).

B. Results

1) Usability: Registration items scored between M = 4.19 and M = 4.33, with 81–86% agreement across all items (Table V). Login usability scored higher overall. All 21 participants agreed they could clearly feel the vibration cues (M = 4.67, 100%) and that the process became faster after multiple attempts (M = 4.71, 100%). First-attempt login success was lower (M = 3.76, 61.9% agreement), reflecting a short but real learning curve: 42.9% of participants became comfortable in 1–2 attempts, 52.4% in 3–5, and 4.8% in 6–10 (estimated mean ≈ 3.1 attempts).

2) Perceived security and memorability: The security-perception block scored highest overall, with item means between 4.52 and 4.86 (Table VI). Every participant agreed that a nearby observer could not work out the real password, every participant agreed the scheme would hold up across multiple sessions of shoulder-surfing, and 95.2% said they would trust the scheme for sensitive uses such as banking. After 24 hours with no practice, 76.2% of participants still felt they could recall both the secret locations (M = 3.95) and the vibration-tap mappings (M = 3.90). Cognitive load looked manageable: 81.0% said the mental effort was acceptable next to a PIN or pattern lock, and all 21 said they could feel the vibration cue reliably even with background noise. The weakest item in the block was daily practicality at M = 3.76 (57.1% agreement), which suggests that the scheme suits high-value logins better than the dozens of routine unlocks every smartphone user does in a day.

TABLE V. USABILITY RESULTS: REGISTRATION AND LOGIN (5-POINT LIKERT; N = 21).

Item	M	SD	Agree / SA (%)
Registration Usability			
Registration process was easy to understand	4.19	0.93	85.7%
Selecting secret locations was intuitive	4.33	0.73	85.7%
Defining decoy locations was straightforward	4.19	0.68	85.7%
Mapping vibration patterns to tap styles was clear	4.29	0.85	85.7%
Confident I could remember my password after registration	4.24	1.09	81.0%
Login Usability			
Login process was easy to complete	4.52	0.60	95.2%
Could clearly feel the vibration cues before decoy taps	4.67	0.48	100.0%
Could distinguish between different vibration patterns	4.62	0.59	95.2%
Responding with the correct tap style after vibration was natural	4.48	0.81	90.5%
Time required to log in was acceptable	4.29	0.85	85.7%
Did not feel frustrated during the login process	4.43	0.60	95.2%
Successfully logged in on first attempt	3.76	1.14	61.9%
Process became faster and more natural after multiple attempts	4.71	0.46	100.0%

TABLE VI. PERCEIVED SECURITY, MEMORABILITY, AND COGNITIVE LOAD (5-POINT LIKERT; N = 21).

Item	M	SD	Agree / SA (%)
Perceived Security			
Vibration cues make the system more secure than PIN/pattern	4.71	0.46	100.0%
A nearby observer could NOT determine my real password	4.86	0.36	100.0%
Decoy taps effectively hide which taps are my real password	4.67	0.58	95.2%
System would resist shoulder-surfing across multiple sessions	4.76	0.44	100.0%
Per-session randomization adds meaningful security	4.67	0.48	100.0%
Would trust system for sensitive apps (banking, email)	4.52	0.60	95.2%
Memorability and Cognitive Load			
Could remember secret locations after 24 hours (no practice)	3.95	0.97	76.2%
Could remember vibration-tap mappings after 24 hours (no practice)	3.90	0.94	76.2%
Mental effort acceptable compared to PIN/pattern	4.05	0.92	81.0%
Did not feel mentally overloaded while using the system	4.14	0.79	81.0%
System would be practical for daily use on my smartphone	3.76	0.89	57.1%
Could perceive vibration feedback even in a noisy environment	4.57	0.51	100.0%

3) System Usability Scale (SUS): Twenty of 21 participants completed all SUS items. The mean SUS score was 78.62 (SD = 9.88, median = 80.0, range = 57.5–100.0). This exceeds the commonly cited average of 68 and, on the adjective scale of Bangor et al. [19], falls between the "Good" (≈71.4) and "Excellent" (≈85.5) ratings — closer to "Good" — corresponding to an acceptable system. Eighty per cent of participants (16/20) exceeded the 68 average, and 40% (8/20) scored at or above 80.3. SUS item-level means and the overall score are presented in Table VII.

TABLE VII. SUS ITEM-LEVEL MEANS AND OVERALL SCORE (N = 20)

SUS Item	Mean	Direction
Q1. I would like to use this system frequently	4.19	Positive
Q2. I found the system unnecessarily complex	1.81	Negative
Q3. I thought the system was easy to use	4.14	Positive
Q4. I would need technical support to use this system	1.76	Negative
Q5. The various functions were well integrated	4.29	Positive
Q6. There was too much inconsistency in this system	1.60	Negative
Q7. Most people would learn this system very quickly	3.90	Positive
Q8. I found the system very cumbersome to use	1.95	Negative
Q9. I felt very confident using the system	4.33	Positive
Q10. I needed to learn a lot before I could get going	2.00	Negative
Overall SUS Score (0–100 scale)	**78.62**	**Good (Acceptable)**

4) Inferential statistical analysis: Beyond the descriptive statistics reported above, we applied inferential tests to assess whether the usability and perceived-security results differ significantly from established neutral or benchmark values. Because the Likert items are ordinal and the sample is modest, all per-item comparisons were verified with the non-parametric one-sample Wilcoxon signed-rank test against the scale midpoint (neutral = 3), and the SUS comparison used a one-sample t-test against the widely cited benchmark of 68 [19]. A significance level of $\alpha = 0.05$ was used throughout, and Holm–Bonferroni correction was applied within each instrument block to control the family-wise error rate across multiple comparisons. Effect sizes are reported as Cohen's d for the SUS comparison and as the matched-pairs rank-biserial correlation for the Likert items.

The mean SUS score ($M = 78.62$, $SD = 9.88$, $N = 20$) was significantly higher than the 68 benchmark, $t(19) = 4.81$, $p < 0.001$, with a large effect size (Cohen's $d = 1.07$), confirming that the observed usability advantage is unlikely to be an artifact of sampling variation. Every perceived-security item differed significantly and positively from the neutral midpoint after correction (all $p < 0.001$); for example, agreement that a nearby observer could not determine the real password ($M = 4.86$, $SD = 0.36$) yielded $t(20) = 23.7$, $p < 0.001$, and agreement that the scheme would resist shoulder-surfing across multiple sessions ($M = 4.76$, $SD = 0.44$) yielded $t(20) = 18.3$, $p < 0.001$. The two 24-hour memorability items also exceeded the neutral midpoint significantly, recall of secret locations ($M = 3.95$, $SD = 0.97$), $t(20) = 4.5$, $p < 0.001$, and recall of vibration–tap mappings ($M = 3.90$, $SD = 0.94$), $t(20) = 4.4$, $p < 0.001$, though with smaller effect sizes consistent with the wider spread on these items. We note that these tests establish that responses depart significantly from neutrality and from the SUS benchmark; because the study used a single within-subjects condition, they do not constitute a between-conditions comparison against a baseline authentication scheme, which we identify as a limitation and a direction for the controlled comparative study described in Section VII.

C. Discussion

The SUS score landed at 78.62, roughly ten points above the often-cited 68 baseline, which is a fair indication that the haptic-decoy mechanism is not paying for itself with a usability overhead. Earlier visual-obfuscation schemes, such as WIW, show what the alternative looks like; there, the obfuscation tends to wear on the user [6]. Pairing the high security-perception ratings (item means 4.52 to 4.86) with a SUS rated "Good" on the Bangor et al. adjective scale, the data suggests we have the design tradeoff roughly right: better resistance to observation, without a meaningful hit on how the scheme feels to use. The 61.9% first-attempt success rate is what one would expect from any new interaction; the relevant follow-up signal is that all 21 participants agreed the process got faster and more natural after a few tries, which is what learnability looks like in practice. Twenty-four-hour retention came out at 76.2% for both the secret locations and the vibration-tap mappings, so the credential survives an overnight gap without an external aid. The strongest single block of the dataset was security perception: nobody in the sample thought a watcher could recover the password, and 95.2% said they would use the scheme for things like banking. Set next to the SUS, the picture is consistent. Participants found the scheme usable, they thought it was secure, and they put it in the part of the design space we actually targeted, namely, mobile authentication in public.

Set against the other haptic schemes, this one ends up in a slightly different corner of the design space. Feel Vibration [9] runs a challenge–response exchange entirely over a covert haptic channel, which is elegant, but it puts the whole credential into vibrations the user has to remember with no spatial anchor to lean on. HapticLock [12] was built for eyes-free use and is single-modal; it never set out to address shoulder-surfing of the on-screen interaction. The continuous-authentication work of Cao et al. [13] and HandKey [14] goes in a different direction altogether — passive vibration biometrics rather than a chosen credential, with the deployment and revocation issues that follow from any biometric (a vibration signature cannot really be reset once it leaks). Our scheme keeps the user on the familiar ground of a picture password, then adds the covert haptic decoy channel, per-session randomization, the environment-aware adaptive phase, and uses the image file itself as a second factor. From the user side, it is one continuous interaction; from the security side, it is a two-factor authenticator with built-in observation-resistance, reached without an SMS code, a second device, or new hardware.

V. LIMITATIONS

A. Limitations and Use Cases

The proposed scheme improves resistance to shoulder-surfing without requiring privacy screen protectors, which themselves degrade screen visibility for the legitimate user. That said, the design has a few limitations. It depends on the device having a working vibration motor; on phones where the motor is weak or has been disabled in software, the cue can be hard to feel. New users also need a short period of practice before the response to a vibration pattern becomes automatic. Some environmental factors matter as well. If the phone is sitting flat on a surface instead of being held, the vibration is muffled enough that some users will miss it.

The scope of the present evaluation is deliberately bounded, and two boundaries should be read alongside the results. First, the study sample (N = 21) was recruited by convenience from within the university and the authors' contacts; it is therefore not statistically representative of the broader smartphone-using population and may over-represent technically literate users, so the usability and perceived-security figures are best read as indicative rather than population-level estimates. Second, by design, the present study measures usability, memorability, and perceived security, and does not include an adversarial component in which participants observe and then attempt to break one another's login sessions. Accordingly, the shoulder-surfing-resistance claim is grounded in the quantitative single-session bounds derived in Section III — including the explicit multi-session inference threshold of Section III D.4 — together with participants' perceived security, rather than in a measured human-observer attack rate. We treat the controlled adversarial study — with a larger, more diverse, and randomly recruited sample drawn from real public-transport and public-space users, a between-subjects comparison against an established baseline scheme, and a protocol in which observers attempt to reproduce a victim's credential after watching one or more sessions — as the natural next step that converts the theoretical resistance argument into direct empirical evidence, and it is the primary direction for our ongoing work.

Despite these limitations, the system can be useful in several scenarios. It is particularly suitable for smartphone authentication in public environments where shoulder-surfing attacks are more likely to occur, such as airports, public transportation, or cafes. The system can also be used to protect access to sensitive mobile applications, such as banking or enterprise systems. Overall, the approach is most effective for touchscreen devices that support vibration feedback and are used in environments where visual observation attacks are a potential risk.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

This study describes a vibration-aided picture-based authentication scheme that puts secret locations and decoy locations together on the same image and uses covert vibration cues only for the decoy taps. The pieces – a client–server architecture, a structured data model, and a randomized challenge generator – combine to obscure the distinction between real and decoy taps from a visual observer, while keeping the implementation thin enough to run on commodity smartphones and other haptic-capable devices. The current design depends on a working vibration motor and on spatial and temporal thresholds that have been tuned for the prototype hardware; if those parameters are misconfigured, the scheme will either be less usable or less secure than intended.

B. Future Work

A few directions are already in scope. The most immediate is per-user calibration: we want the server to build a short profile of each user's tap timing so that the spatial and temporal thresholds can be moved up or down for that user without altering the threat model. Past that, we plan to try the prototype on wearables, smartwatches, and haptic-capable game controllers; the architecture should already accommodate them. Two algorithm-side refinements are in scope as well. The first is smarter decoy interleaving, including density, clustering, and context-sensitive placement. The second is a stronger crowd-risk classifier for the adaptive authentication phase, ideally one that runs without a cloud round-trip. On the hardening side, we plan to add client-integrity attestation and server-side anomaly detection that flags improbable tap-location or timing patterns before any of them turn into a successful attack.

ACKNOWLEDGMENT

This work was conducted as part of Senior Projects 1 and 2 (CCCY 411) at the College of Computer Science and Engineering, University of Jeddah, under the supervision of Dr. Faisal Alsubaei.

REFERENCES

[1] A. Bhardwaj et al., "Shoulder Surfing Protection Mechanisms: A Systematic Literature Review," arXiv preprint arXiv:2411.18380, Nov. 2024.

[2] F. S. Alsubaei and A. E. Abuhussein, "Vibration-Aided Picture-Based Authentication System and Method," U.S. Patent 12,393,663 B1, Aug. 19, 2025, Assignee: University of Jeddah.

[3] X. Suo, Y. Zhu, and G. Owen, "Graphical passwords: A survey," in *Proc. 21st Annual Computer Security Applications Conference (ACSAC)*, 2005, pp. 463–472, doi: 10.1109/CSAC.2005.27.

[4] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin, "The design and analysis of graphical passwords," in *Proc. 8th USENIX Security Symp.*, Washington, D.C., USA, Aug. 23–26, 1999, pp. 1–14.

[5] R. Dhamija and A. Perrig, "Déjà Vu: A user study using images for authentication," in *Proc. 9th USENIX Security Symp.*, Denver, CO, USA, Aug. 2000, pp. 45–58.

[6] S. Man, D. Hong, and M. Matthews, "A shoulder-surfing resistant graphical password scheme – WIW," in *Proc. Int. Conf. Security and Management*, Las Vegas, NV, USA, Jun. 2003, pp. 105–111.

[7] G. W. Bin, S. Safdar, R. Akbar, and S. Subramanian, "Graphical authentication based on anti-shoulder surfing mechanism," in *Proc. 2nd Int. Conf. Future Networks and Distributed Systems*, New York, NY, USA, Jun. 2018, pp. 1–6, doi: 10.1145/3231053.3231073.

[8] I. Maiti, M. Jadliwala, J. He, and I. Ray, "RandomPad: Usability of randomized mobile keypads for defeating inference attacks," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2017, pp. 1–7.

[9] W. Luo, B. Lan, X. Wan, Z. Liu, Y. Zeng, and J. Ma, "Feel vibration: Challenge-response mobile authentication with covert channel," in *Proc. IEEE Int. Conf. Communication Technology (ICCT)*, Oct. 2020, pp. 1089–1096, doi: 10.1109/ICCT50939.2020.9295824.

[10] J. Thorpe and P. C. van Oorschot, "Human-Seeded Attacks and Exploiting Hot-Spots in Graphical Passwords," in *Proc. 16th USENIX Security Symp.*, Boston, MA, USA, Aug. 2007.

[11] L. Y. Por, I. O. Ng, Y. L. Chen, J. Yang, and C. S. Ku, "A systematic literature review on the security attacks and countermeasures used in graphical passwords," *IEEE Access*, vol. 12, pp. 53408–53423, 2024, doi: 10.1109/ACCESS.2024.3386259.

[12] E. Freeman, S. Sherlock, and S. Brewster, "HapticLock: Eyes-Free Authentication for Mobile Devices," in *Proc. 23rd Int. Conf. Multimodal Interaction (ICMI)*, Oct. 2021, doi: 10.1145/3462244.3479921.

[13] H. Cao, H. Jiang, K. Yang, S. Chen, W. Wu, and J. Liu, "Data-Augmentation-Enabled Continuous User Authentication via Passive Vibration Response," *IEEE Internet of Things Journal*, vol. 10, no. 15, pp. 13614–13627, Aug. 2023, doi: 10.1109/JIOT.2023.3264274.

[14] X. Liu et al., "HandKey: Knocking-Triggered Robust Vibration Signature for Keyless Unlocking," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 4910–4925, May 2024, doi: 10.1109/TMC.2023.3298389.

[15] National Institute of Standards and Technology, "Secure Hash Standard (SHS)," FIPS PUB 180-4, U.S. Department of Commerce, Aug. 2015, doi: 10.6028/NIST.FIPS.180-4.

[16] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. 2001 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, Kauai, HI, USA, Dec. 2001, pp. I-511–I-518, doi: 10.1109/CVPR.2001.990517.

[17] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. 2005 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, San Diego, CA, USA, Jun. 2005, pp. 886–893, doi: 10.1109/CVPR.2005.177.

[18] J. Brooke, "SUS: A 'quick and dirty' usability scale," in *Usability Evaluation in Industry*, P. W. Jordan, B. Thomas, B. A. Weerdmeester, and I. L. McClelland, Eds. London, U.K.: Taylor & Francis, 1996, pp. 189–194.

[19] A. Bangor, P. T. Kortum, and J. T. Miller, "An Empirical Evaluation of the System Usability Scale," *International Journal of Human-Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008, doi: 10.1080/10447310802205776.