APPS BY VIKAS

Meet Vikas

GenzLudo
Ludo with 3D characters and up to 8 players. A first-of-its-kind no-teaming-up feature keeps play fair. Try it - you'll love the difference.
Download on Google Play

Kachu
Too much scrolling straining your neck? Add your addictive apps and complete daily neck exercises to unlock them.
Download on Google Play

01 / UI & UX STATES

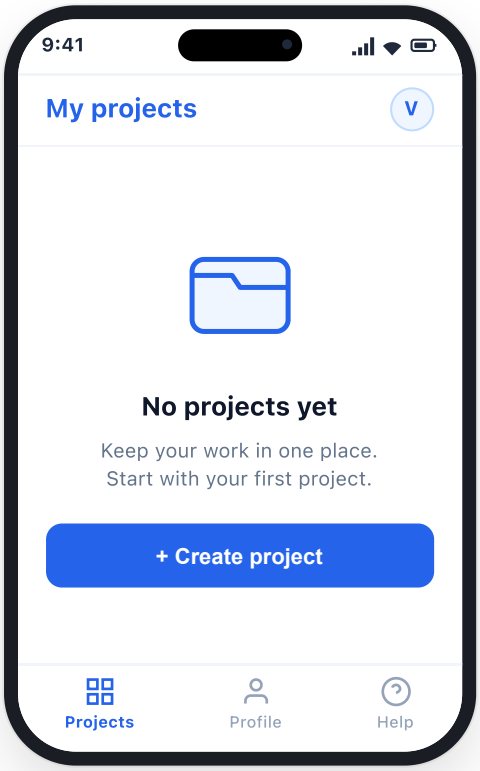
Empty State

Show what to do when there is no content yet.

WHEN TO USE IT

A new user opens a page before adding their first item.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. A clear message
Say what is missing.

2. A simple reason
Explain what the page is for.

3. One next step
Use a clear action button.

AVOID

- A blank screen.
- A button called "Click here".
- An error for a new account.

> **The user should quickly know:**
> ***"Nothing is here. I can add it."***

QUICK CHECK

Can a new user understand the page and take the next step?
Test it: tap "Create project". It should open the new project form.

Use this state only after loading has finished and no items exist.

02 / UI & UX STATES

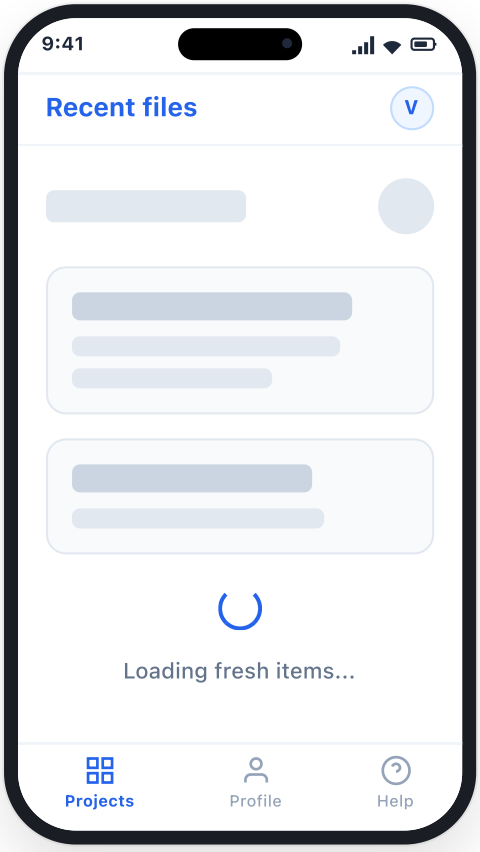
Loading State

Show that content is on its way without freezing the screen.

WHEN TO USE IT

The app is fetching data from the server or calculating results.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Skeleton placeholders
Match the shape of incoming cards and text.

2. A gentle shimmer
Shows active progress rather than a frozen UI.

3. Immediate layout hints
Prevents annoying layout shifts when data arrives.

AVOID

- A blank white screen.
- An unlabelled infinite spinner.
- Blocking the entire screen for a small list.

> **The user should quickly know:**
> ***"The app is working. My content will appear shortly."***

QUICK CHECK

Does the user immediately know that data is actively loading?
Test it: throttle network to Slow 3G and verify no layout shift occurs.

Keep skeleton box dimensions closely matched to the actual content.

03 / UI & UX STATES

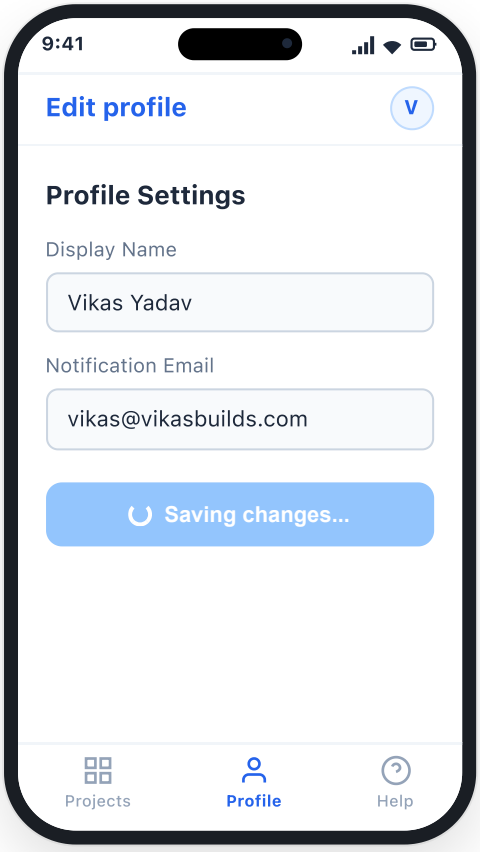
Saving State

Confirm the save request and stop repeated button taps.

WHEN TO USE IT

The user taps Save or Update and the server is storing changes.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. In-button spinner
Replaces button text with an animated saving icon.

2. Disabled interaction
Disables the button to prevent duplicate submissions.

3. Reassuring status
Informs the user that their edits are in flight.

AVOID

- Allowing multiple rapid taps on Save.
- Leaving the button completely unchanged.
- Closing the form before the server confirms save.

> **The user should quickly know:**
> ***"My changes are being stored right now."***

QUICK CHECK

Can a user accidentally tap Save multiple times in a row?
Test it: rapidly tap "Save" 5 times and check network tab for 1 single request.

Always re-enable the button with an error message if the save fails.

04 / UI & UX STATES

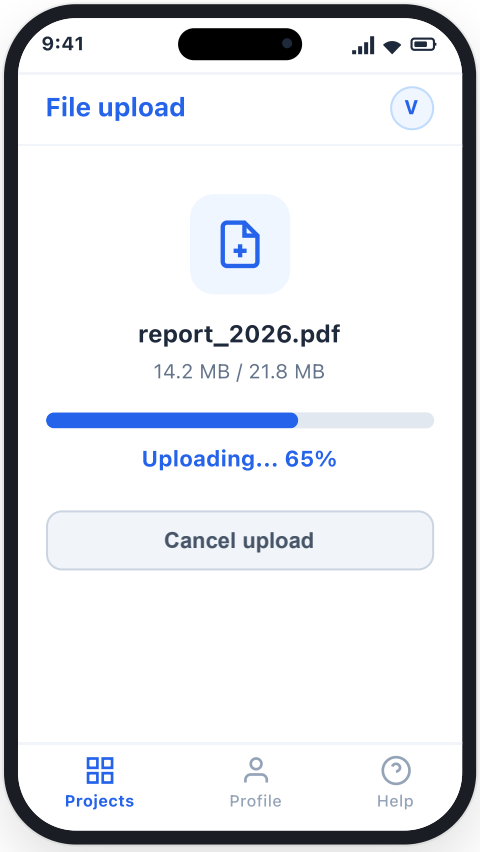
Progress State

Show real numerical progress for operations that take time.

WHEN TO USE IT

The user uploads large files, exports databases, or imports records.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Exact progress bar
Displays real percentage and transferred bytes.

2. File name and size
Confirms which item is currently transferring.

3. Clear cancel option
Allows users to stop the process if needed.

AVOID

- A fake timer bar that gets stuck at 99%.
- Hiding how much time or data remains.
- A transfer that cannot be paused or cancelled.

> **The user should quickly know:**
> ***"I know exactly how much has finished and how much is left."***

QUICK CHECK

Does the progress indicator reflect actual server byte transfers?
Test it: pause network and confirm progress holds rather than jumping fake numbers.

For actions taking longer than 10 seconds, display estimated time remaining.

05 / UI & UX STATES

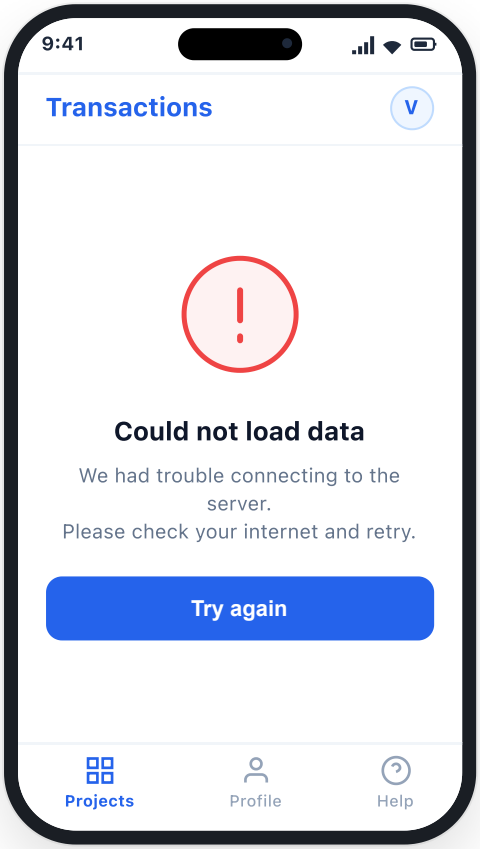
Error State

Explain what went wrong in plain words and offer an instant retry.

WHEN TO USE IT

A server crash, bad API response, or connection failure occurs.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Friendly explanation
Uses plain human English instead of HTTP 500 codes.

2. Actionable retry CTA
One prominent blue button to trigger a fresh request.

3. Help / support link
Provides an escalation path if the issue persists.

AVOID

- Technical jargon like "Internal Server Error 500".
- Blaming the user for a backend failure.
- A dead-end screen with no retry button.

> **The user should quickly know:**
> ***"Something failed, but I know how to retry immediately."***

QUICK CHECK

Can the user recover from this error with a single tap?
Test it: reconnect internet, tap "Try again", and confirm data loads.

Never show raw database logs, stack traces, or uncaught JSON to users.

06 / UI & UX STATES

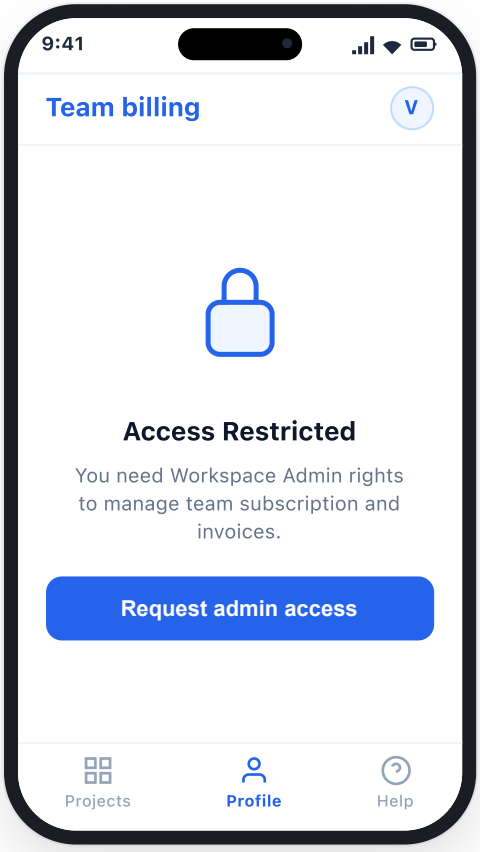
Permission Denied

Tell users why access is restricted and who can grant it.

WHEN TO USE IT

A team member visits an admin page, billing tab, or private file.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. The exact required role
Explains what permission level is missing.

2. Who to contact
Points users to their workspace administrator.

3. A safe exit path
Provides buttons to request access or go back.

AVOID

- A cold, cryptic "403 Forbidden" error.
- Pretending the page does not exist.
- Trapping the user with no navigation back home.

The user should quickly know:
"I understand why I cannot view this and who can give me access."

QUICK CHECK

Does the user have a direct way to request access or return home?
Test it: tap "Request access" and confirm admin receives an access notice.

Always identify which role is needed so users know who to contact.

07 / UI & UX STATES

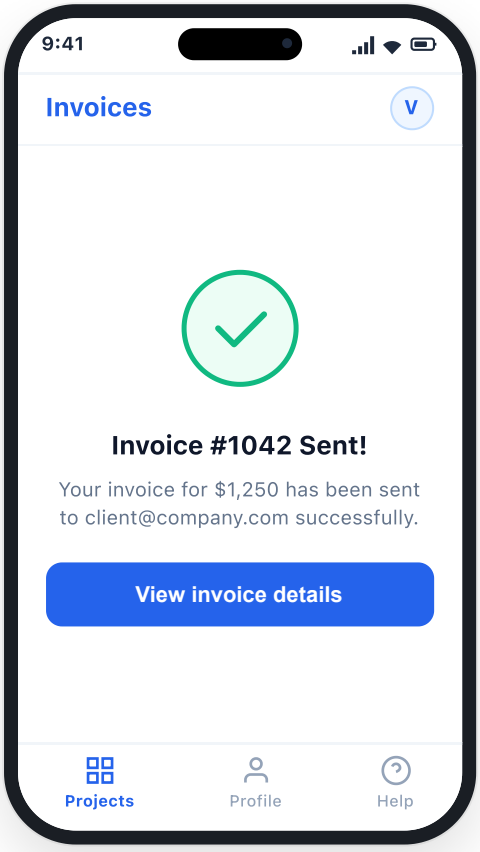
Success State

Celebrate completed actions and guide users toward the next step.

WHEN TO USE IT

A payment completes, an invoice is sent, or a new record is created.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Positive visual badge
A crisp green checkmark confirming success.

2. Key transaction summary
Shows receipt number, amount, or recipient.

3. Logical next action
Clear buttons to view the result or create another.

AVOID

- Silent success where the UI does not change.
- Vague confirmations with no reference numbers.
- A dead-end screen with no forward navigation.

> **The user should quickly know:**
> ***"My action succeeded and I have the confirmation details."***

QUICK CHECK

Does the confirmation display the exact reference details just created?
Test it: verify invoice number and recipient match the user's input.

For minor updates, use a brief floating toast instead of a full page takeover.

08 / UI & UX STATES

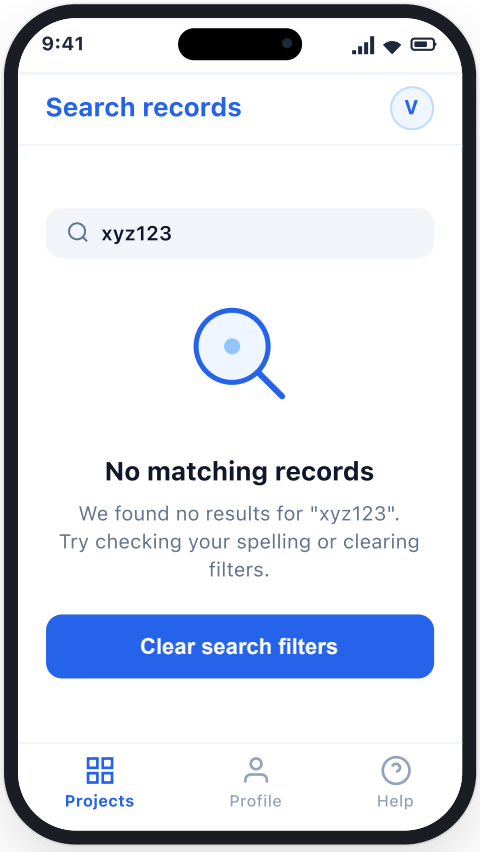
No Results State

Help users recover when their search or filters return zero matches.

WHEN TO USE IT

A search keyword or active filter combination yields no records.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. The exact search query
Displays the words and filters applied.

2. Helpful suggestions
Suggests broader keywords or alternative terms.

3. One-tap filter reset
A prominent blue button to reset all filters.

AVOID

- Confusing this with an Empty State (new account).
- Leaving an empty white space with no message.
- Forcing users to backspace every character manually.

> **The user should quickly know:**
> ***"Nothing matched my search, but I can easily reset it."***

QUICK CHECK

Does tapping 'Clear filters' immediately restore the full list?

Test it: tap "Clear search filters" and verify all items reappear instantly.

Always differentiate between 'zero records created' and 'zero search matches'.

09 / UI & UX STATES

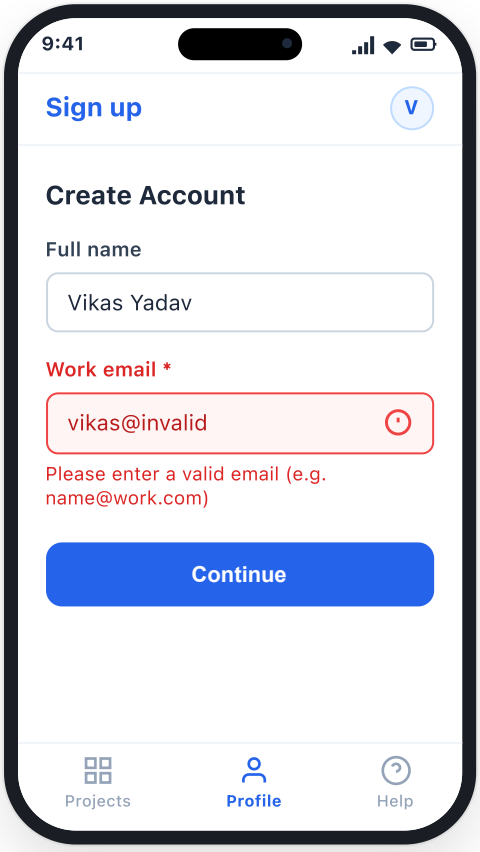
Validation State

Highlight errors inline right next to the offending input field.

WHEN TO USE IT

A user types an invalid email, weak password, or skips a required field.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. High-contrast red border
Clearly outlines the specific field with an error.

2. Actionable corrective text
Explains exactly how to fix the input below the field.

3. Live re-validation
Clears the error badge as soon as the user corrects it.

AVOID

- Flashing errors before the user even finishes typing.
- Generic messages like "Invalid input".
- Clearing the user's password or input on error.

> **The user should quickly know:**
> ***"I see exactly where the mistake is and how to fix it."***

QUICK CHECK

Does the error indicator disappear the moment the user corrects the input?
Test it: type a valid email format and confirm the red outline disappears.

Validate on field blur or form submit, not on the very first keystroke.

10 / UI & UX STATES

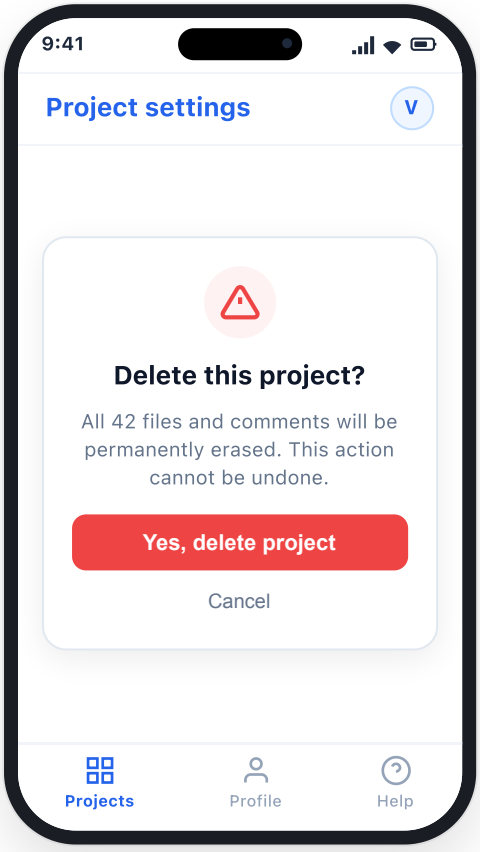
Warning State

Warn users before destructive or irreversible actions take place.

WHEN TO USE IT

Deleting an entire project, cancelling a subscription, or purging logs.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Prominent warning symbol
Amber/red badge highlighting permanent danger.

2. Specific consequences
Lists exactly what will be lost (e.g. 42 files).

3. Distinct button hierarchy
Destructive button in red, safe Cancel in neutral gray.

AVOID

- Vague questions like "Are you sure?".
- Hiding the real impact of the action.
- Making the delete button look like the default safe option.

> **The user should quickly know:**
> ***"I understand the permanent impact before making my choice."***

QUICK CHECK

Is Cancel the safest, most obvious button to prevent mistakes?

Test it: verify tapping outside the modal safely cancels the deletion.

For critical actions, require the user to type the project name to confirm.

11 / UI & UX STATES

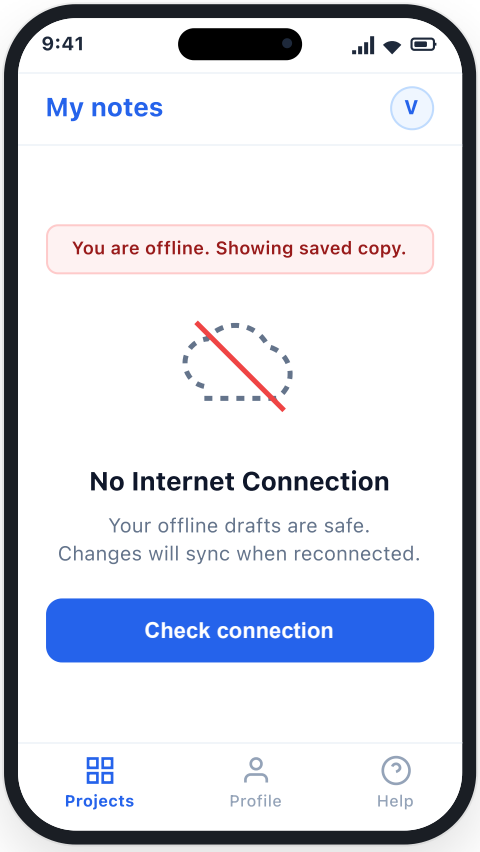
Offline State

Keep cached data readable and protect user work without internet.

WHEN TO USE IT

The device loses Wi-Fi or cellular network connection.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Persistent offline banner
Subtle top bar indicating offline mode.

2. Cached content display
Shows previously loaded records seamlessly.

3. Safe draft storage
Assures users that new edits are saved locally.

AVOID

- Crashing the app or showing a white blank screen.
- Discarding user input typed while offline.
- Showing an infinite spinner that never times out.

> **The user should quickly know:**
> ***"I am offline, but my work is safe and I can read my saved files."***

QUICK CHECK

Can the user read cached notes and write drafts without internet?

Test it: turn on Airplane mode and verify notes remain readable and editable.

Disable remote-only actions with clear tooltips while offline.

12 / UI & UX STATES

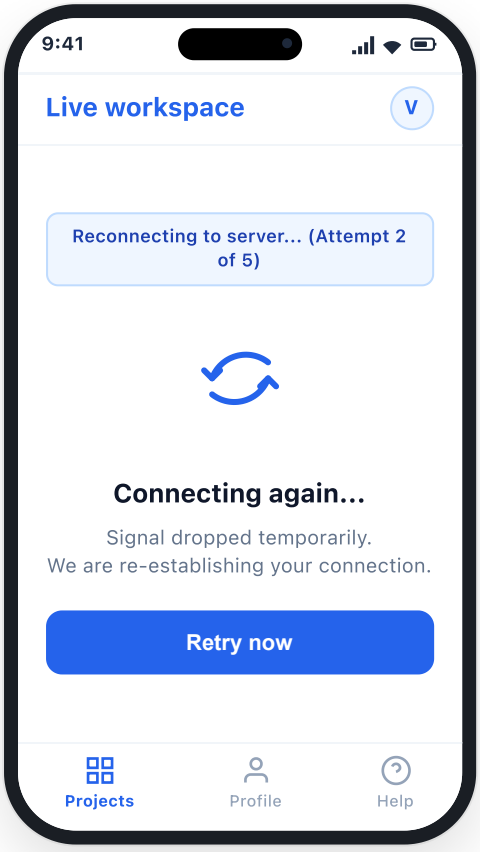
Reconnecting State

Show automatic reconnection status without alarming the user.

WHEN TO USE IT

The network drops briefly and the app attempts automatic reconnection.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Gentle status banner
Top notification showing active reconnect attempt.

2. Attempt counter
Shows current attempt number (e.g. Attempt 2 of 5).

3. Manual retry shortcut
Allows users to trigger immediate reconnection.

AVOID

- Flickering between connected and disconnected modals.
- Throwing full-screen error alerts for 2-second drops.
- Resetting open editor forms on reconnect.

> **The user should quickly know:**
> ***"The app is handling the network hiccup automatically."***

QUICK CHECK

Does the app seamlessly resume the session once internet restores?

Test it: turn Wi-Fi back on and verify banner disappears without page reload.

Use exponential backoff (2s, 4s, 8s) to avoid overwhelming your API.

13 / UI & UX STATES

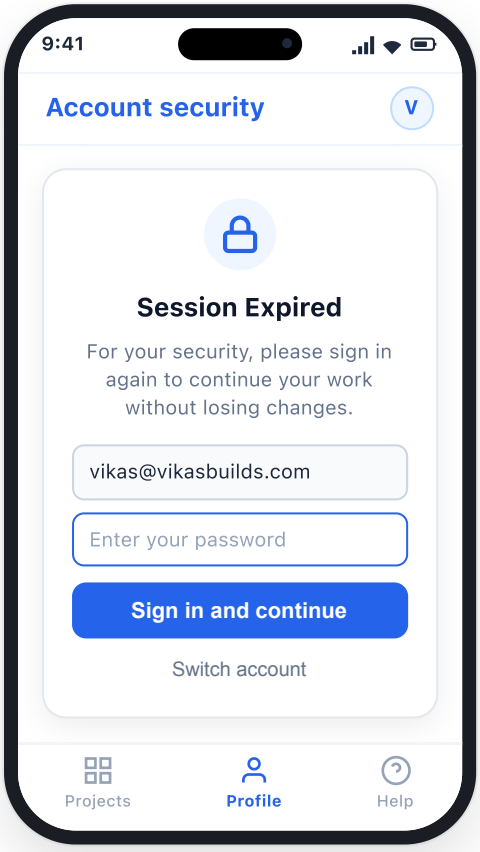
Session Expired

Ask for re-authentication while preserving unsaved work.

WHEN TO USE IT

A login token or security cookie expires due to inactivity.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Plain security reason
Explains session timed out to protect the account.

2. Pre-filled user email
Saves user time with their account pre-selected.

3. Seamless resume
Returns directly to the exact page and form after login.

AVOID

- Hard redirecting to home page and wiping unsaved input.
- Cryptic unhandled "401 Unauthorized" errors.
- Forcing users to start multi-step forms from scratch.

> **The user should quickly know:**
> ***"My session expired for safety, but I will not lose my work."***

QUICK CHECK

Does logging back in preserve typed inputs on the current screen?
Test it: let session expire, sign back in, and confirm form data is intact.

Store drafted form contents in local storage before displaying the login modal.

14 / UI & UX STATES

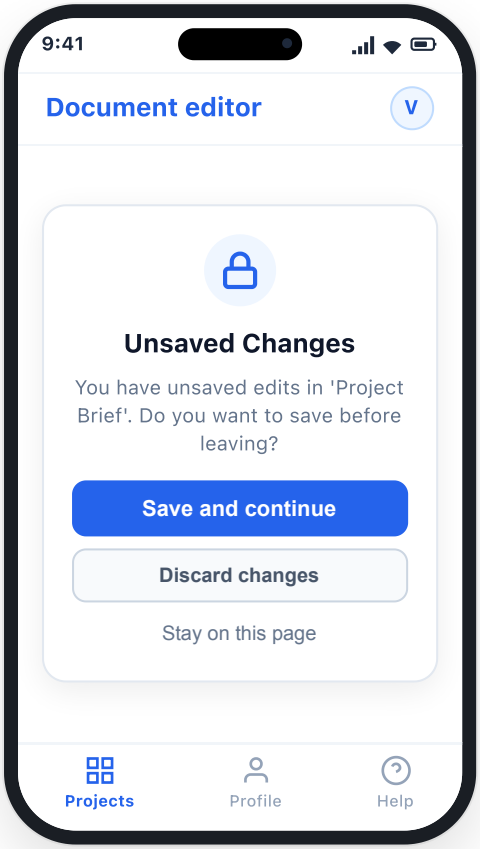
Unsaved Changes

Prevent accidental loss when navigating away from dirty forms.

WHEN TO USE IT

The user taps Back, closes a tab, or clicks a link with unsaved edits.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Explicit loss warning
States clearly that edits will be discarded if closed.

2. Name of modified item
Reminds users which document has pending changes.

3. Three clear choices
Save & continue, Discard changes, or Stay on page.

AVOID

- Letting users close forms with zero warning.
- Confusing ambiguous buttons like "Yes / No".
- Treating minor read-only page views as dirty forms.

> **The user should quickly know:**
> ***"I have unsaved edits and can choose whether to keep or discard them."***

QUICK CHECK

Does clicking 'Stay on this page' keep all typed text completely intact?
Test it: type text, tap Back, select Stay, and verify text is preserved.

Highlight modified fields in the background so users see what changed.

15 / UI & UX STATES

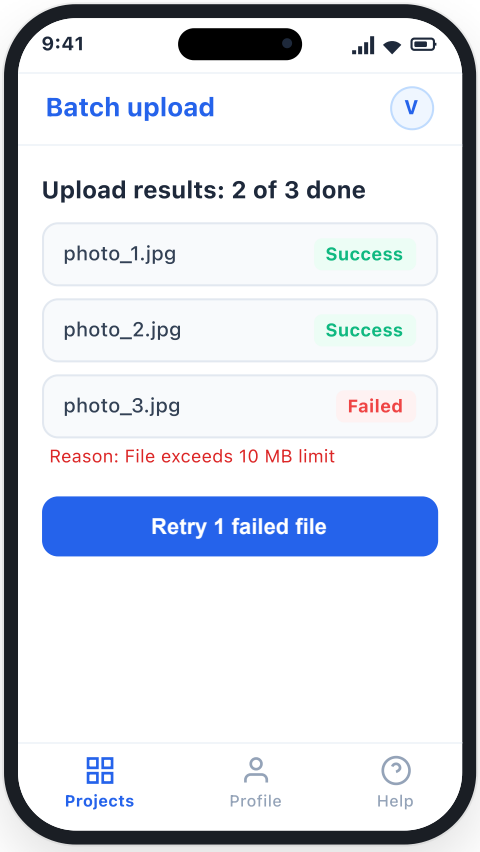
Partial Success

Distinguish successful items from failed ones in batch tasks.

WHEN TO USE IT

A batch action (e.g. uploading 10 images) finishes with 2 errors.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Clear tally
Displays exact counts (e.g. 2 succeeded, 1 failed).

2. Exact reason per failure
Lists why each failed item could not be completed.

3. Targeted retry button
A single button to retry only the failed subset.

AVOID

- Labeling the entire batch as complete failure.
- Forcing users to re-upload files that already succeeded.
- Hiding which specific records encountered errors.

> **The user should quickly know:**
> ***"Most files uploaded, and I only have to fix the one that failed."***

QUICK CHECK

Does clicking retry process only the failed records without duplicates?

Test it: tap "Retry failed" and confirm successful items are not duplicated.

Keep completed item IDs clearly confirmed so users know they are safe.

16 / UI & UX STATES

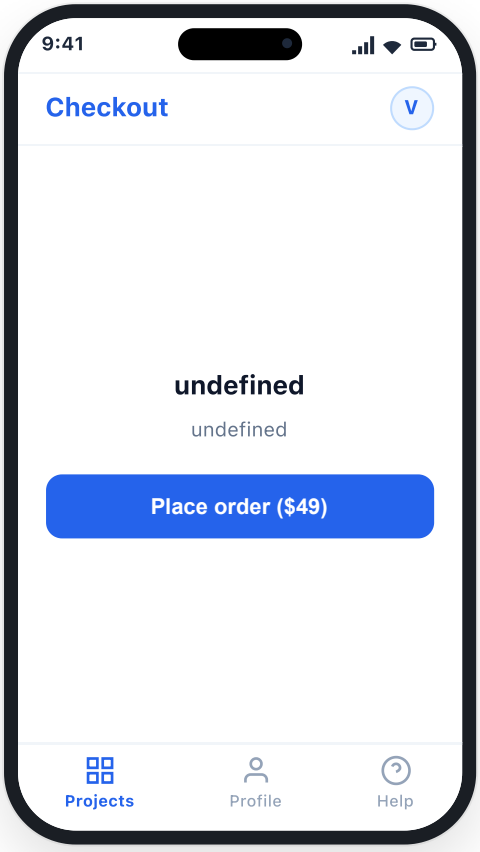
Disabled State

Explain why an action cannot be performed yet.

WHEN TO USE IT

A submit button is locked until all required fields are filled.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Reduced visual weight
Faded color (40% opacity) and not-allowed cursor.

2. Helpful inline hint
Explains what requirement is missing right above or on hover.

3. Instant activation
Button illuminates immediately once fields are satisfied.

AVOID

- Completely unresponsive "mystery" disabled buttons.
- Leaving users guessing which form field was missed.
- Hiding the button entirely from the page.

> **The user should quickly know:**
> ***"I know what is missing before I can click this button."***

QUICK CHECK

Does filling the missing field instantly activate the button?

Test it: complete the required field and check that button color turns active blue.

Prefer keeping buttons active and showing inline field hints on tap.

17 / UI & UX STATES

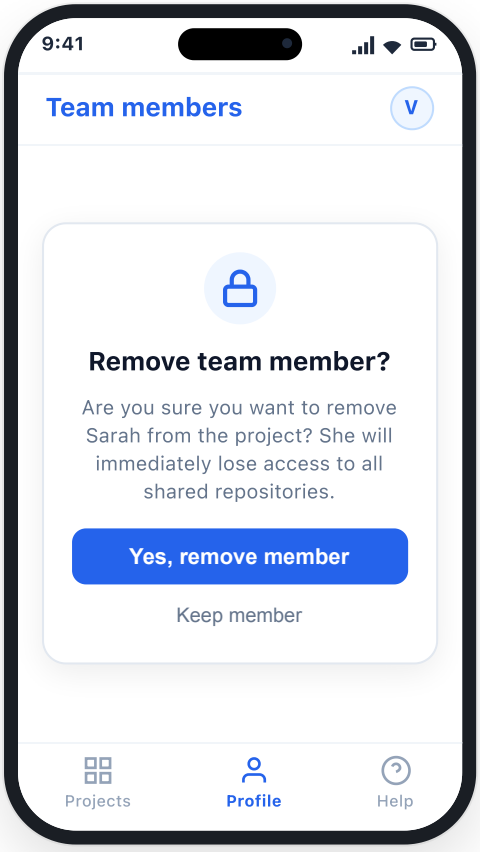
Confirmation State

Confirm deliberate human intent for high-impact actions.

WHEN TO USE IT

Revoking API keys, transferring team ownership, or deleting records.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Named subject
Explicitly names the item or person being affected.

2. Immediate consequence
Explains that access will be lost right away.

3. Clear action verbs
Uses "Remove member" instead of generic "OK".

AVOID

- Using confirmation popups for harmless everyday clicks.
- Ambiguous button labels like "Yes" and "No".
- Confusing confirmation with an error dialog.

> **The user should quickly know:**
> ***"I can double-check before making an important decision."***

QUICK CHECK

Can the user easily back out by tapping Cancel or pressing Esc?
Test it: press Escape and ensure the modal dismisses without any change.

Reserve confirmation dialogs only for irreversible or destructive actions.

18 / UI & UX STATES

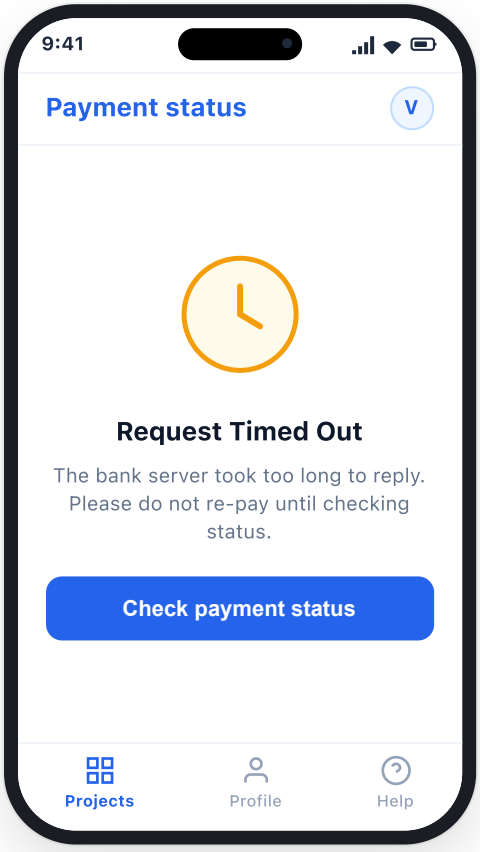
Timeout State

Prevent duplicate charges when external servers take too long.

WHEN TO USE IT

A payment gateway, bank webhook, or API call exceeds the deadline.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Explicit timeout notice
Explains the server exceeded the response deadline.

2. Duplicate caution
Warns users not to double-click or re-submit payments.

3. Status verification CTA
Offers a safe button to verify transaction receipt.

AVOID

- Letting spinners run indefinitely without timing out.
- Allowing users to accidentally double-charge cards.
- Crashing with unparsed connection drop errors.

> **The user should quickly know:**
> ***"The gateway was slow, but I should verify before trying again."***

QUICK CHECK

Does the UI prevent the user from triggering duplicate charges?

Test it: ensure the payment CTA is disabled while checking transaction status.

Include the transaction reference ID so support agents can look it up.

19 / UI & UX STATES

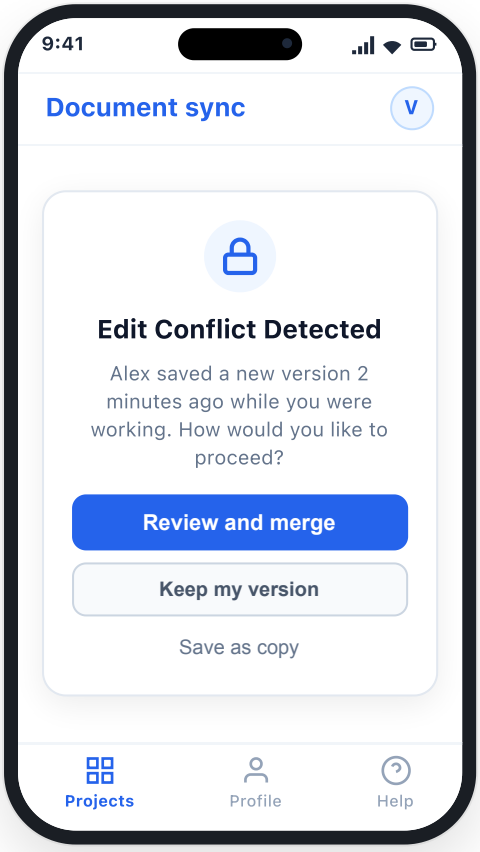
Conflict State

Protect concurrent edits when another person changes a record.

WHEN TO USE IT

Two collaborators edit the same task, article, or record at the same time.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Who edited and when
Names the teammate and timestamp of competing edit.

2. Side-by-side diff
Visual comparison between server and local edits.

3. Non-destructive options
Merge, overwrite, or save local version as a copy.

AVOID

- Silently overwriting someone else's work (last-write-wins).
- Erasing the current user's drafted edits.
- A cryptic database concurrency error.

The user should quickly know:
"There is an edit conflict, but my changes will not be wiped out."

QUICK CHECK

Can the user save their edits as an independent copy?
Test it: select "Save as copy" and confirm both documents exist safely.

Never overwrite records silently without notifying the user.

20 / UI & UX STATES

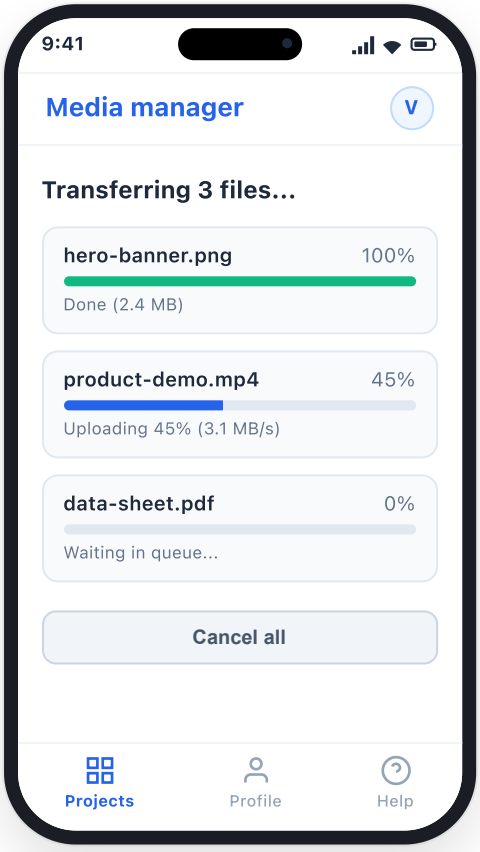
Upload / Download State

Provide visibility and individual controls for multi-file transfers.

WHEN TO USE IT

Users upload gallery images, attach project files, or download archives.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Per-file progress
Individual progress bars and transfer speeds.

2. Instant thumbnail preview
Visual recognition of images and document types.

3. Itemized control
Ability to cancel or retry individual files separately.

AVOID

- A single generic bar hiding individual failures.
- No option to abort a heavy transfer in progress.
- Leaving the user unaware of current network speed.

> **The user should quickly know:**
> ***"I can see exactly which files finished, are uploading, or waiting."***

QUICK CHECK

Can the user cancel a single heavy file without aborting the others?

Test it: tap 'X' on file 2 and confirm file 1 remains completed.

Show thumbnail previews for image and media uploads.

21 / UI & UX STATES

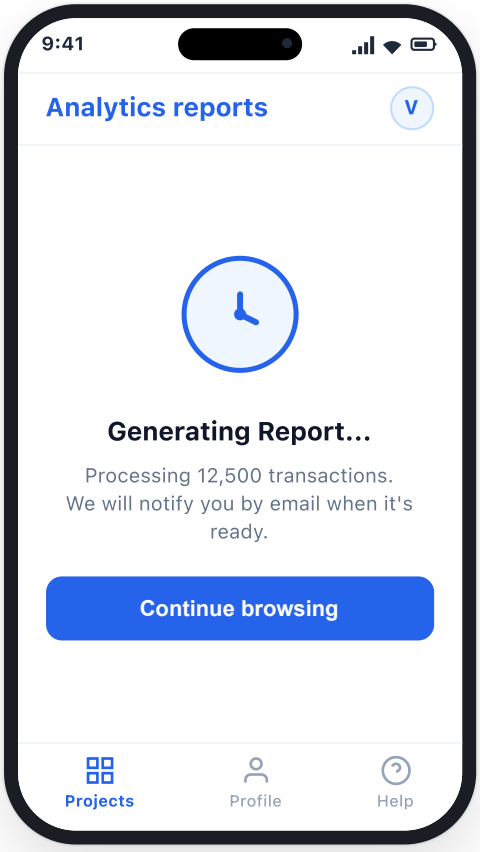
Background Processing

Let users continue other work while heavy tasks run in the background.

WHEN TO USE IT

Generating monthly financial reports, exporting ZIPs, or transcoding video.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Reassurance to leave
Tells users they can safely close or navigate away.

2. Estimated duration
Shares expected processing time (e.g. ~2 minutes).

3. Delivery notification
Explains how results will be delivered (email or in-app bell).

AVOID

- Forcing users to stare at a modal for 5 minutes.
- Cancelling the entire job if the user switches tabs.
- Failing to notify users when the job finishes.

> **The user should quickly know:**
> ***"The system is working in the background; I can do other tasks."***

QUICK CHECK

Can the user switch tabs without interrupting the background job?
Test it: navigate away during export and verify notification arrives when done.

Send an in-app toast or email when the background task finishes.

22 / UI & UX STATES

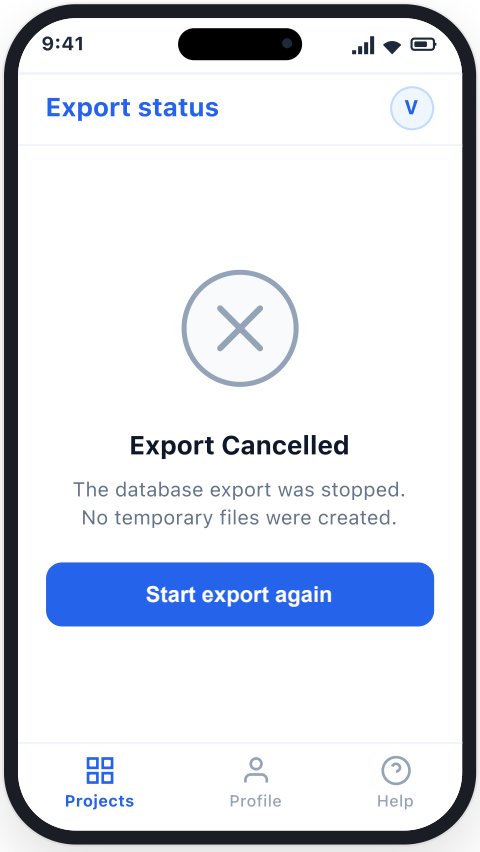
Cancelled State

Confirm when a process was aborted and clean up temporary state.

WHEN TO USE IT

The user cancels a file upload, stops an export, or declines a checkout.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Explicit cancellation notice
Confirms the cancellation completed cleanly.

2. Zero side-effects assurance
Assures that no partial or unwanted changes were saved.

3. Restart CTA
One-tap button to restart if cancelled accidentally.

AVOID

- Leaving users unsure if changes were partially committed.
- Leaving orphaned temporary files taking up disk space.
- Treating voluntary user cancellation like an application crash.

> **The user should quickly know:**
> ***"My cancellation went through cleanly and nothing was saved."***

QUICK CHECK

Does restarting the process initialize with clean default values?

Test it: tap "Start export again" and verify fresh configuration loads.

Immediately release allocated memory and network sockets upon cancellation.

23 / UI & UX STATES

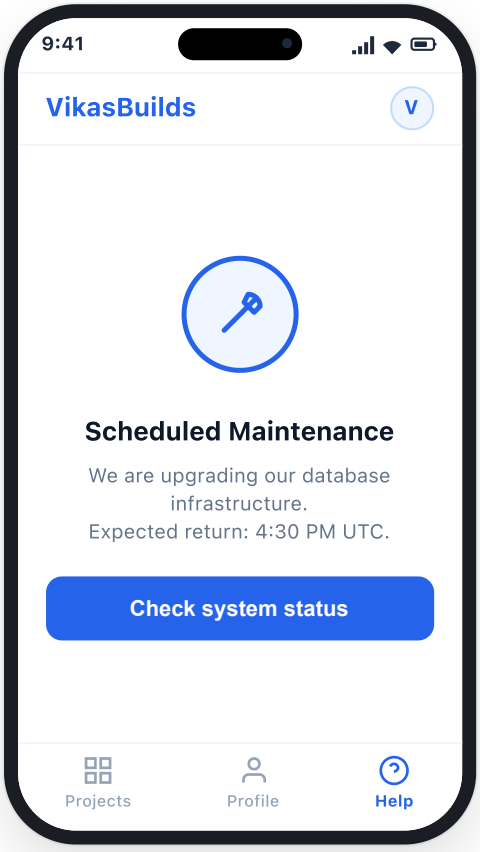
Maintenance State

Communicate planned downtime clearly with expected completion time.

WHEN TO USE IT

Scheduled database migrations, server maintenance, or system upgrades.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Friendly, calm explanation
Explains downtime is scheduled improvement, not a crash.

2. Estimated completion time
Shows specific return time in UTC and local timezone.

3. Status page link
Direct button to live status page (status.domain.com).

AVOID

- Serving generic 500 error pages during planned maintenance.
- Leaving users wondering if the app was shut down.
- Hosting the status page on the exact same offline server.

> **The user should quickly know:**
> ***"The app is down for planned work and will be back shortly."***

QUICK CHECK

Does the status link lead to an independently hosted status dashboard?
Test it: click "Check system status" and confirm it opens the live uptime tracker.

Cache maintenance pages at the CDN edge so downtime doesn't overload origin.

24 / UI & UX STATES

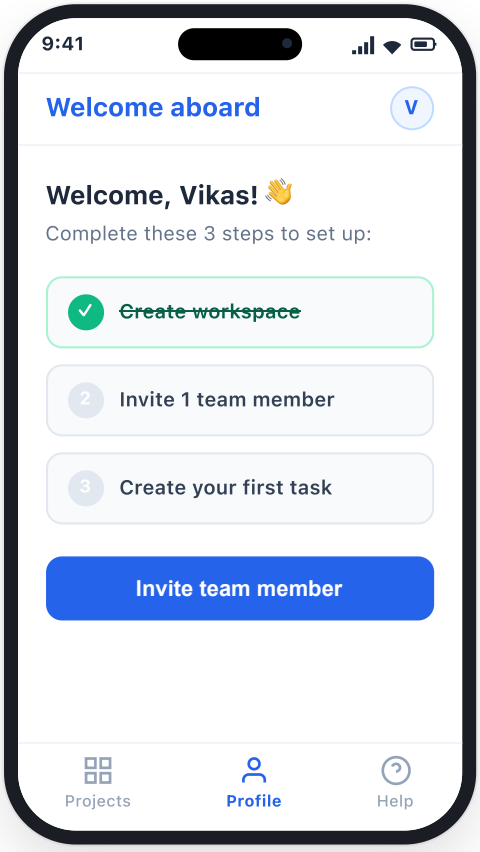
First-Time User

Welcome brand-new users with an actionable 3-step activation path.

WHEN TO USE IT

A user signs up and enters the dashboard for the very first time.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Warm personalized greeting
Welcomes the user by name with an encouraging tone.

2. Short 3-step checklist
Clear, bite-sized tasks to reach the first value moment.

3. Single prominent CTA
Focuses the user's attention on step 1 without overwhelm.

AVOID

- Dumping new users onto a totally empty, confusing screen.
- Bombarding them with 10 sequential modal tooltips.
- Hiding the skip button for experienced users.

> **The user should quickly know:**
> *"I know exactly where to start and what to do first."*

QUICK CHECK

Can the user dismiss the guide and explore freely if they prefer?

Test it: verify a clear "Skip for now" link exists beside the primary CTA.

Focus first-time guidance on reaching the 'Aha!' moment as quickly as possible.

25 / UI & UX STATES

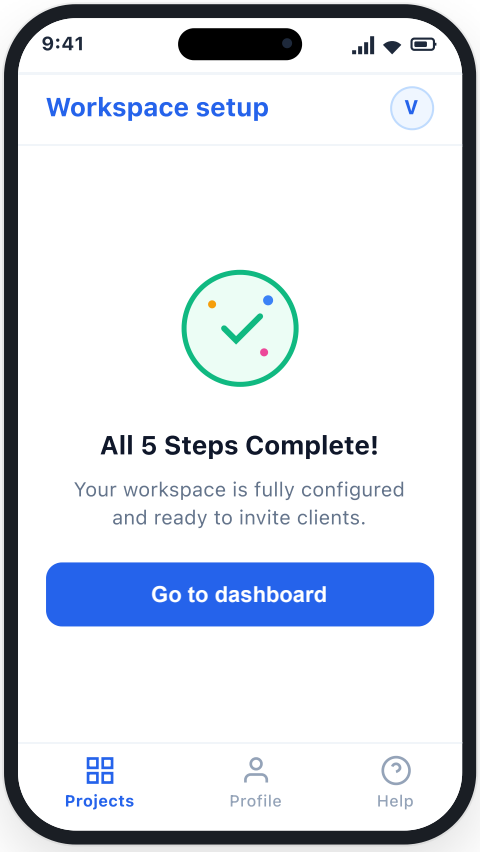
Completion State

Celebrate milestones and bridge users smoothly into the main workflow.

WHEN TO USE IT

All onboarding steps are finished, KYC is approved, or setup is complete.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Celebratory visual
A cheerful completion badge or subtle celebration graphic.

2. Summary of achievements
Confirms key milestones achieved (domain, billing, team).

3. Launchpad button
A prominent blue button opening the main active workspace.

AVOID

- An anti-climactic ending that leaves users asking "Now what?".
- Trapping users inside completed setup wizards.
- Heavy animations that freeze older mobile devices.

The user should quickly know:
"I have completed all requirements and am ready to get to work."

QUICK CHECK

Does the primary CTA take the user straight into their live dashboard?
Test it: tap "Go to dashboard" and confirm setup wizard does not show again.

Keep celebratory animations lightweight (CSS or SVG) so they never lag.

26 / UI & UX STATES

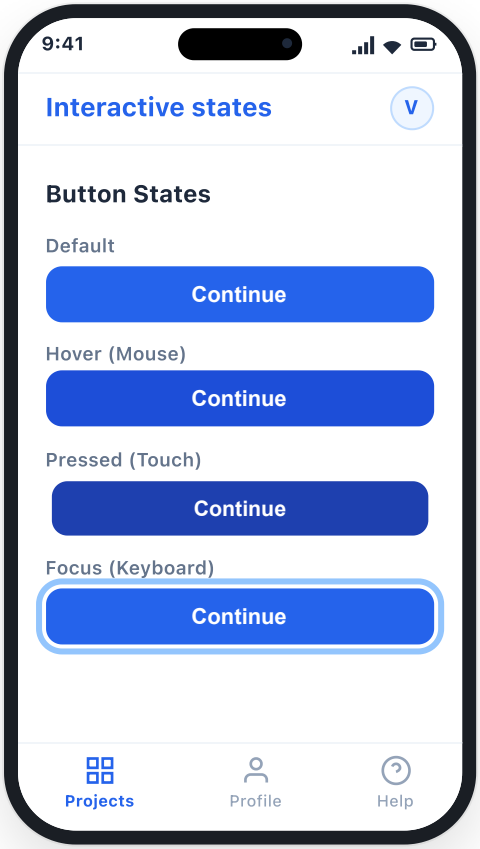
Hover / Focus / Pressed

Provide immediate tactile visual feedback across all input methods.

WHEN TO USE IT

Users hover with a mouse, tap with a finger, or navigate with a keyboard Tab.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Distinct hover state
Slight elevation and color darkening on cursor hover.

2. Immediate tap feedback
Depressed scale or active shade on touch down.

3. Accessible focus ring
High-contrast 2px/3px outline for keyboard accessibility.

AVOID

- Using `outline: none` without providing an accessible alternative.
- Zero visual response when tapping buttons on mobile.
- Hover effects that stick or get stuck on mobile touch.

> **The user should quickly know:**
> ***"I can clearly see and feel that my tap or keystroke registered."***

QUICK CHECK

Can you navigate and activate the button using only Tab + Enter?
Test it: press Tab on keyboard and verify high-contrast blue focus ring appears.

Never strip browser focus rings without adding an accessible high-contrast replacement.

27 / UI & UX STATES

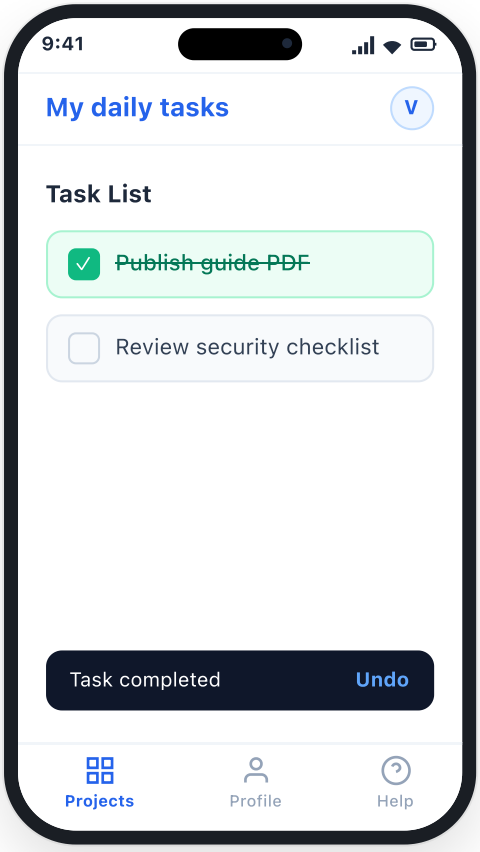
Instant Update & Undo

Update the UI immediately and roll back gracefully if the server fails.

WHEN TO USE IT

Liking a post, checking a task checkbox, or toggling settings (Optimistic UI).

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Instant state toggle
Checkbox marks instantly without waiting for network.

2. Quick Undo prompt
A bottom floating snackbar with an immediate Undo button.

3. Graceful rollback
Unchecks the box and alerts user if the server API rejects it.

AVOID

- Freezing the UI for 500ms on every checkbox tap.
- Failing silently without rolling back state on network error.
- Forgetting to provide an Undo option for quick mis-clicks.

> **The user should quickly know:**
> ***"The app feels lightning-fast, and I can undo if I made a mistake."***

QUICK CHECK

Does the checkbox revert smoothly if the network request fails?

Test it: simulate 500 server response and confirm task unchecks with an error toast.

Always keep the previous state in memory until the server confirms success.

28 / UI & UX STATES

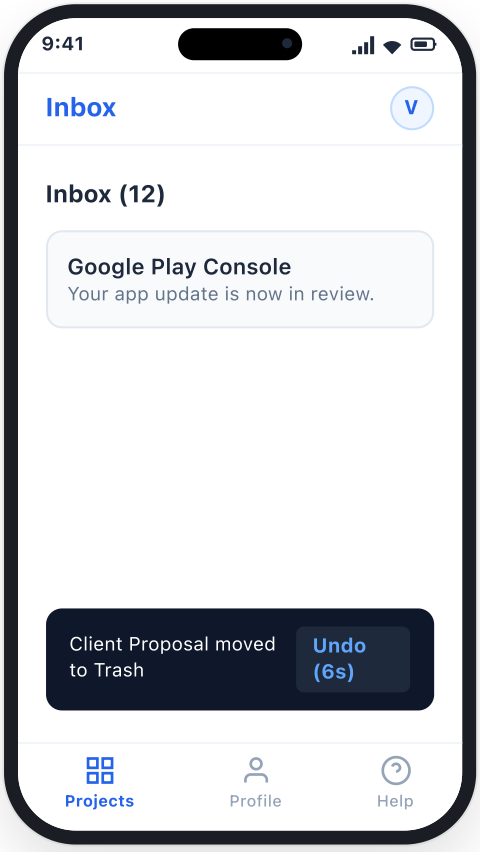
Undo State

Let users easily reverse destructive actions within a timed window.

WHEN TO USE IT

Moving an email to trash, deleting a task, or removing an item.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Instant action toast
Confirms which item was moved or archived.

2. Prominent Undo trigger
A high-contrast blue action button inside the snackbar.

3. Visible timer
Gives users 6 to 10 seconds before permanent commit.

AVOID

- Interrupting routine deletions with annoying "Are you sure?" popups.
- Hiding the Undo notification behind other screen layers.
- Auto-dismissing the Undo button after only 1 second.

> **The user should quickly know:**
> ***"I made a mistake, but I can restore it with a single tap."***

QUICK CHECK

Does tapping Undo immediately restore the deleted item to its place?
Test it: tap "Undo (6s)" and verify the item reappears in the list.

Undo is far superior to annoying confirmation dialogs for routine actions.

29 / UI & UX STATES

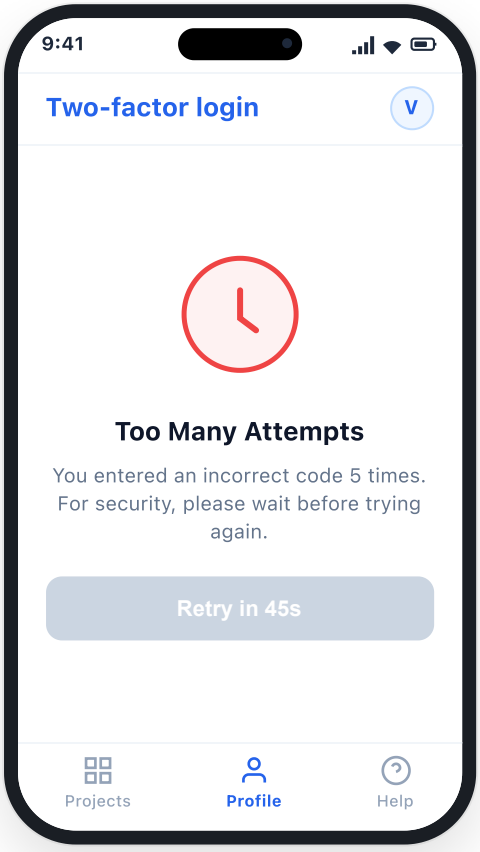
Too Many Requests

Enforce security rate limits with a visible countdown timer.

WHEN TO USE IT

Repeated failed login tries, spamming OTP requests, or API rate limiting.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Transparent reason
Explains rate limiting is a security measure to protect account.

2. Live countdown timer
Displays exact seconds remaining (e.g. Retry in 45s).

3. Auto-unlocking CTA
Button automatically re-enables when the timer reaches zero.

AVOID

- Silent lockouts where users click repeatedly without knowing why.
- Permanent lockouts with no reset pathway.
- Vague error messages like "System error occurred".

The user should quickly know:

"I need to wait a few seconds, and I know exactly when I can retry."

QUICK CHECK

Does the button automatically unlock once the countdown hits zero?

Test it: wait out countdown and verify button becomes active blue without page reload.

Include an alternative unlock route (e.g. 'Email login link') for locked users.

30 / UI & UX STATES

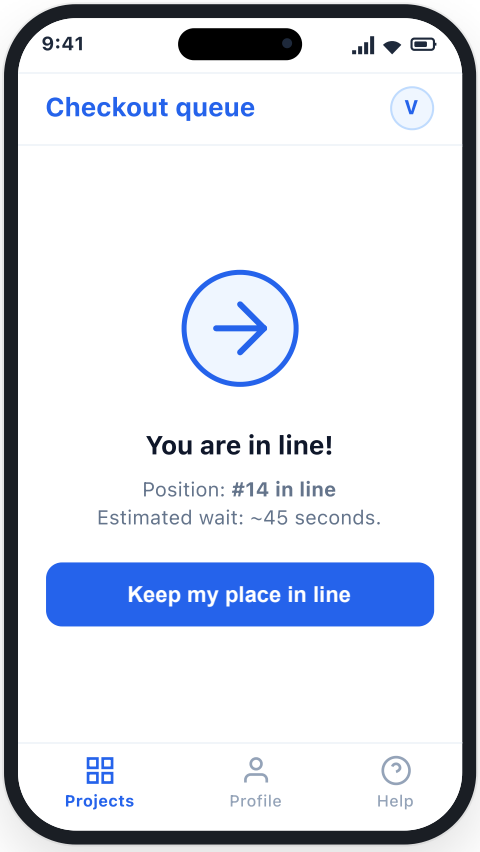
Waiting in Queue

Show real queue positions and estimated wait times during traffic peaks.

WHEN TO USE IT

High-demand flash drops, AI generation queues, or ticket checkouts.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Real queue position
Displays current spot in line (e.g. #14 of 80).

2. Realistic wait estimate
Calculates realistic wait time based on active throughput.

3. Reassurance to stay
Advises users to keep their browser tab open.

AVOID

- Fake spinning circles that pretend processing is instant.
- Losing a user's place in line if they accidentally refresh.
- Jumping queue numbers randomly backward.

> **The user should quickly know:**
> ***"I am holding my place in line and will be served automatically."***

QUICK CHECK

Does an accidental page refresh preserve the user's queue position?
Test it: refresh browser and check that session cookie retains queue spot #14.

Store queue tokens in session storage so reloads never lose queue place.

31 / UI & UX STATES

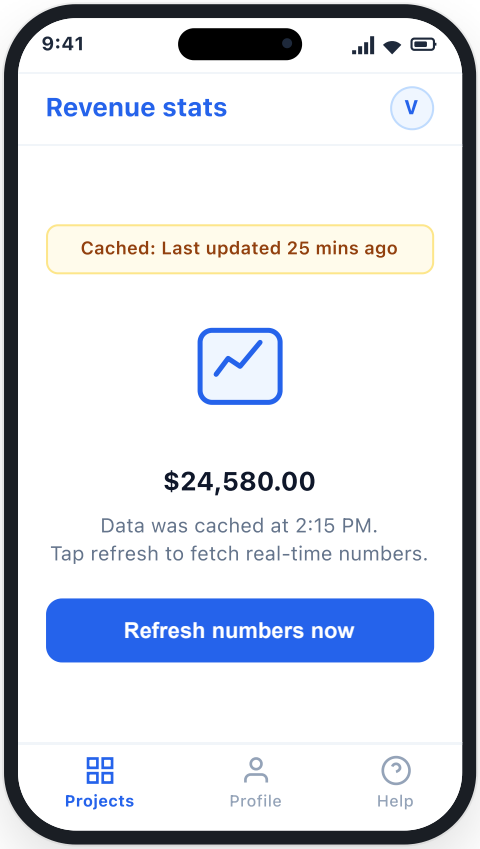
Old Data State

Indicate when displayed metrics are cached and offer an easy refresh.

WHEN TO USE IT

Cached analytics dashboards, financial rates, or offline read copies.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Plain cache timestamp
States exactly when data was last retrieved.

2. Manual refresh button
An obvious one-tap button to bust cache and pull fresh data.

3. Critical data warning
Highlights stale data if numbers affect business decisions.

AVOID

- Presenting stale, cached metrics as real-time live truth.
- Hiding timestamps in deep secondary settings menus.
- Disabling refresh buttons while users need fresh data.

> **The user should quickly know:**
> ***"This is a cached snapshot and I can refresh whenever I need fresh data."***

QUICK CHECK

Does tapping refresh update both the timestamp and displayed numbers?
Test it: tap "Refresh numbers now" and check that timestamp updates to "Just now".

Highlight stale indicators prominently if financial transactions are involved.

32 / UI & UX STATES

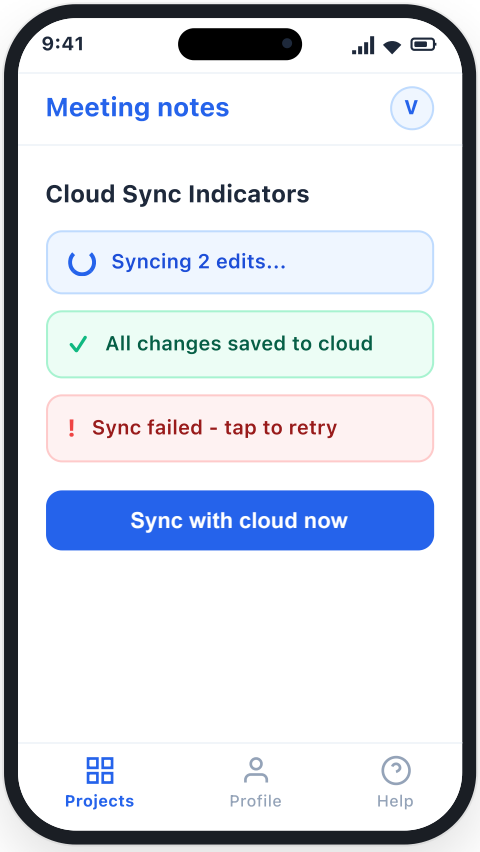
Sync State

Make cloud synchronization transparent, continuous, and reliable.

WHEN TO USE IT

Note taking apps, collaborative boards, and auto-saving cloud documents.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Continuous status icon
Cloud icon showing Syncing, Synced, or Error.

2. Cloud vs local distinction
Distinguishes between local device save and remote cloud sync.

3. Direct retry button
Allows users to trigger immediate re-sync if sync drops.

AVOID

- Claiming "Saved" when local edits haven't reached the server.
- Blocking typing or interaction while sync takes place.
- Silently discarding edits when background sync fails.

The user should quickly know:
"I always know whether my changes are safely stored in the cloud."

QUICK CHECK

Can the user continue typing freely while background sync is occurring?
Test it: type rapidly while observing the cloud sync indicator.

Always keep a local IndexedDB backup so sync delays never cause data loss.

33 / UI & UX STATES

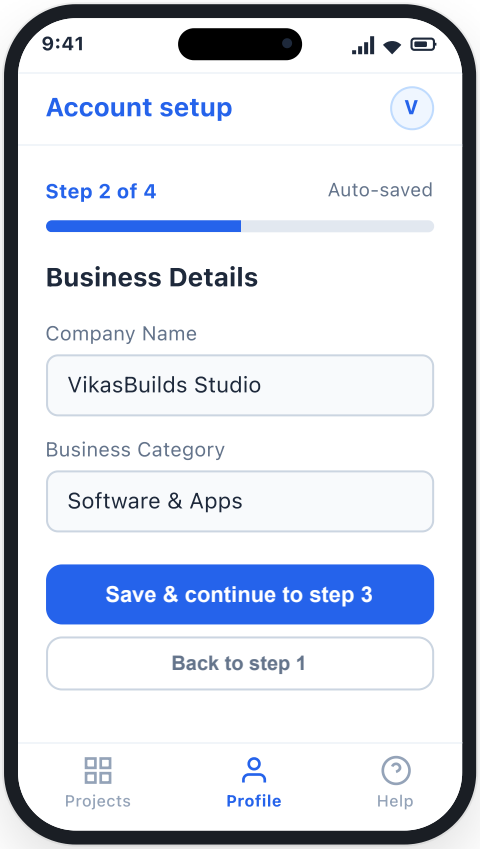
Multi-Step Form

Show clear progress and preserve entered data across every step.

WHEN TO USE IT

Long workflows like loan applications, onboarding, or checkout wizards.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Numbered progress indicator
Shows current step, total steps, and upcoming stages.

2. Safe Back button
Allows users to revisit earlier steps without wiping answers.

3. Per-step auto-save
Confirms each completed step is stored immediately.

AVOID

- Resetting all previous answers if the user taps "Back".
- Hiding how many steps remain in the overall flow.
- Forcing completion in a single sitting without draft save.

> **The user should quickly know:**
> *"I know exactly where I am in the flow and how much is left."*

QUICK CHECK

Does clicking 'Back' retain every single previously entered value?
Test it: go back to Step 1 and verify all text fields remain filled.

Auto-save each step to the database so users can resume later from any device.

34 / UI & UX STATES

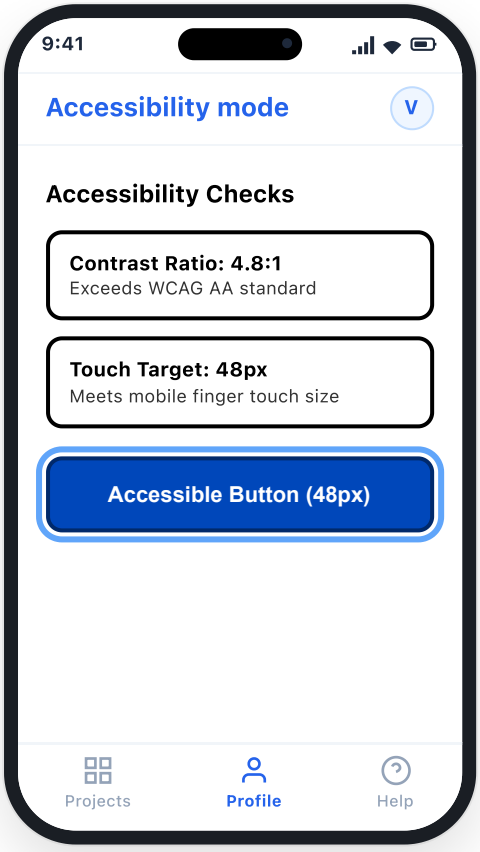
Accessibility State

Ensure clean readability, strong contrast, and full keyboard control.

WHEN TO USE IT

Meeting WCAG accessibility standards for screen readers, keyboard, and contrast.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. High contrast colors
Minimum 4.5:1 text-to-background contrast ratio (WCAG AA).

2. Large tap targets
Minimum 44x44px or 48x48px interactive touch boundaries.

3. Descriptive screen reader labels
Full `aria-label` text for every icon and action button.

AVOID

- Light gray text on white backgrounds that strains reading.
- Icon-only buttons with zero screen reader labels.
- Trapping keyboard focus inside popups or modals.

The user should quickly know:
"Everyone can read, navigate, and use this interface comfortably."

QUICK CHECK

Can the entire screen be navigated and read aloud with a screen reader?
Test it: enable VoiceOver / TalkBack and verify every button is announced clearly.

Always provide descriptive aria-labels for icon-only action buttons.

35 / UI & UX STATES

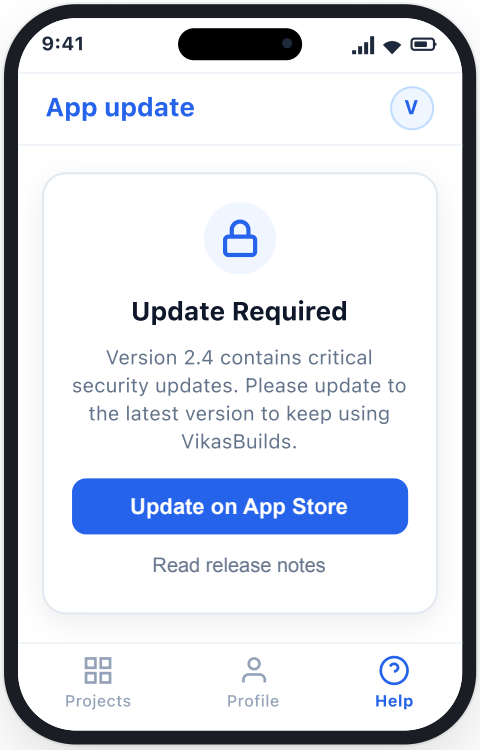
Update Required

Guide users smoothly to the app store for mandatory version upgrades.

WHEN TO USE IT

Breaking API deprecations, database schema changes, or critical security patches.

EXAMPLE / MOBILE APP



WHAT TO SHOW

1. Transparent upgrade reason
Explains why updating is required (e.g. security fix).

2. Direct App Store link
One tap opens the Play Store or App Store page directly.

3. Version & release highlights
Shows version number and bulleted improvements.

AVOID

- Letting the app crash abruptly without an update explanation.
- Trapping users on a broken screen with no store download link.
- Treating optional updates with disruptive mandatory lockouts.

> **The user should quickly know:**
> ***"I need to update my app, and one tap takes me straight to the store."***

QUICK CHECK

Does the button open the store app directly to the download page?
Test it: tap "Update on App Store" and confirm store listing opens immediately.

Differentiate between optional 'Update available' banners and mandatory gates.

More things worth exploring

Find more free guides at vikasbuilds.com/resources