Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

SafeHer: A Progressive Web Application for Women Safety with Real-Time SOS Alerting, Live GPS Tracking, and Cross-Platform Emergency Response

Ankit Kumar Panda[1], Sudesha Gautami[2], T Ravi Teja[3], Kritik Jena[4], Sanjit Kumar Acharya[5]
Department of Computer Science and Engineering, NIST University, Berhampur, Odisha – 761008, India

Abstract - When crisis hits, help must come fast - especially for women turning to digital aids in safety apps. Built with React.js, SafeHer emerges as a web-powered tool that works like an app but lives in the browser, ready on any screen. Rather than install software, it runs straight from modern browsers, pulling strength from built-in web tech. Location sharing springs to life via the Geolocation API, updating movement without delay. A sudden shake of the phone wakes the alarm, thanks to motion sensors tapped through the Device Motion API. Behind scenes, data flows instantly through Firebase Realtime Database, keeping connections tight. Google Maps draws the map trail, showing where someone is at any second. Alerts push out by SMS, cutting waiting time when every heartbeat counts. People pick who gets notified, shaping their circle before trouble strikes. One touch sets it off - or just jolt the device - to send signals flying. When triggered, the SOS feature sends live GPS data inside a clickable map link. Alongside that message, warnings go out via texts and online alerts to every saved contact at once. While help is on the way, fresh position signals keep coming, helping responders track progress moment by moment. Since SafeHer runs as a web app you install straight from your browser, no download stores are needed - just tap and add. It works smoothly whether someone uses Android phones, iPhones, or regular computers. Testing under different signal strengths and gadgets showed alerts arriving in roughly three seconds using mobile data. Locations stayed sharp, usually within five meters outside buildings where sky view was clear. In trial emergencies done 500 times, the system held up nearly all the time, failing less than four percent. When stacked beside similar tools made for women's protection, this one handled weak connections better thanks to backup texting, woke faster when activated, reached more device types without hiccups.

Keywords: Women Safety, Progressive Web Application, React.js, Emergency Response System, GPS Location Tracking, SOS Alert, Firebase Realtime Database, Device Motion API

INTRODUCTION

Every third woman on Earth deals with physical or sexual violence during her life - often happens near city corners, bus stops, or empty sidewalks. When sound gets choked off, fingers can't press a device, or eyes aren't nearby to see it, things like emergency alerts or quick calls stop working. Data from the World Health Organization shows these risks haven't faded. Busy plazas, train paths, strange neighborhoods - they repeat as spots where trouble strikes. If speech disappears and muscles lock up, standard safety moves lose their grip fast.

Right off the bat, today's online tools give solid options beyond regular phone apps. Built right into your everyday browser, Progressive Web Apps work across gadgets big and small - no downloading needed. These sites act like real programs thanks to special links between browsers and machine parts. GPS tracking shows up neatly using built-in location features found in most modern devices. Tilt and shake actions? They're caught by motion sensors tied directly into web code. Safety functions stay strong even though everything runs live from a webpage. Cross-device ease means updates go out fast without waiting on approval systems. Size stays lean since heavy files aren't stored locally. People find what they need faster when there's no search through digital storefronts. Skipping install steps opens doors for anyone jumping in at once. The moment you visit, it just works - plain and simple.

Apps like bSafe, Shake2Safety, or India's Himmat show phones can support real-world safety. Connection gaps break them though - no fallback via SMS traps people without help. Certain ones run only in select areas, limiting reach. When urgency strikes, clicking through screens eats time. Strength comes from designs that stay open while working across systems without hiccups. Position updates continue even when networks wobble. Alerts blast out using more than one path simultaneously. When you shake the device, help appears even if you do not touch the screen. As things get busier, hidden systems adjust behind the scenes without making a sound.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

When danger strikes, SafeHer leaps into action. Built on React.js, it works like a website but feels like an app. Instead of relying solely on Wi-Fi or data, it switches to SMS if signals drop - silently, instantly. Real-time location comes through the browser's built-in GPS tool. Shake your phone, even without touching the screen, alerts go out using motion sensors inside the device. Alerts travel two ways at once so nothing slips through. Firebase keeps user data secure and available across devices. Maps appear clear and clickable thanks to Google's mapping system. No waiting. No tapping twice. The moment help is needed, response begins. Speed wins when every second shifts outcomes.

A key part of this project is a working cross-platform PWA that sends alerts through SMS and internet channels at once, reaching five saved emergency contacts - no download from an app store needed. Another piece: when there's no data signal, it switches to SMS automatically so messages still go out, something many similar tools can't do. Built into the system is motion-sensing tech using device movement, letting users trigger an SOS by shaking their phone, useful when they cannot press buttons. Now picture this: the browser keeps sending live GPS spots nonstop during emergencies using its built-in tracking tool. Right after that comes round-after-round testing - five hundred runs done on purpose - to check how fast alerts fire off, whether locations stay precise, power drains slow or fast, plus if the whole thing holds up when needed.

What comes next breaks down like this. After a look at past efforts - apps and wearables made for women's safety - Section II moves into how artificial intelligence spots danger, while also touching on what governments have tried. Instead of jumping straight into design, Section III lays out exactly what needs solving, spelling out the must-haves. Before any testing happens, Section IV walks through how all pieces fit together in the new setup. How things work comes first in Section V. After that, what gets built shows up in Section VI - think React.js setup, turning it into a PWA, linking Firebase, using Google Maps API, along with an SMS gateway. Safety and personal data concerns take center stage in Section VII. Numbers from testing appear next in Section VIII. What those numbers mean lands in Section IX. Down the road possibilities fill Section X. Last part, Section XI, wraps everything up.

LITERATURE REVIEW

These years, looking into how tech helps keep women safe has drawn attention across different areas. From apps on phones and websites to tiny sensors built into devices, efforts have spread wide. Machine learning shows up in some approaches, while how people interact with machines shapes others. To place SafeHer within what already exists, four broad types of earlier projects get a closer look right now.

Mobile application-based safety systems

An early SOS app on Android arrived through Shende and Deshmukh, slipping GPS details into automatic texts sent to saved numbers. While it showed sending location via SMS could work, the system didn't keep tracking movement nor handled weak indoor signals well. Moving ahead, Agrawal's group added tap-ready Google Maps links inside those alerts, so rescuers reached victims without decoding vague landmarks. Trials found help arriving quicker than when relying on spoken descriptions, proving tiny tweaks in message layout can shift results sharply.

Built by Bhuvan and team [3], WoSApp relied on locals to respond when someone triggered an alert, pairing everyday people near the scene with formal rescue units. Where cities were crowded, delays dropped by 34% - but the system stumbled in places few had downloaded the app. Instead of waiting, Choksi and Shah [4] used Firebase Cloud Messaging to send alerts fast, provided there was decent 4G, making live cloud pushes the go-to method for speed. Halfway into their study, Rashid and Ullah [13] looked at four dozen personal safety apps from 2015 to 2022, spotting clear patterns: tools worked better if they combined two alert paths, tracked movement nonstop, and needed little tapping to activate. These insights quietly guided how SafeHer took shape.

Wearable and IoT-based safety devices

From tiny sensors came alerts - Kumar along with Singh made a gadget that watched movement and pulse using phone signals, spotting odd body reactions automatically. It caught unusual changes well, however building each one cost too much money while power drained fast, lasting half a day when running nonstop. Elsewhere, Gupta's team tucked crash detectors plus location chips inside headgear for female scooter riders, telling actual accidents apart from normal halts quite often. Such on-body protection works alongside apps yet requires people to spend more first and adjust habits - a hurdle slowing wider use.

AI and machine learning in safety systems

Inside a phone app, Patil with Jadhav used sound pictures to teach a smart system that spots cries for help fast - close to 88 percent right. Without asking anything from users, it responded on its own during live listening sessions. Instead of noise, Sharma's group studied motion patterns through sensors; their method caught many fights and hardly mistook normal moves. Reading digital chats came next - Mishra plus Yadav looked for silent SOS signs online, raising questions about boundaries between watchful care and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

unseen tracking. From there, Jha and colleagues built a setup where phones learn threats locally, sharing insights but never private files, balancing protection with personal space held intact.

Government and institutional safety initiatives

After the 2012 Nirbhaya incident, Delhi Police introduced the Himmat app, sending real-time location updates straight to first responders. While active solely inside Delhi, it demands users sign up face-to-face at a local station - no remote access allowed. When mobile networks drop, there is no backup via text messages. Money from the Nirbhaya Fund helped roll out comparable tools in other states, though each works on its own, isolated patchwork. Without a unified, open system available nationwide, SafeHer's ability to bridge different platforms becomes clearly significant.

Gap analysis

Even though some progress has been made, key problems remain. While most current apps need constant internet access, they fail when connections are weak or missing - this matters a lot in villages and smaller towns where dangers for women rise sharply. When signals drop, these systems lose track right after sending the first warning, so rescuers cannot follow any later shifts in position. Instead of quick actions, many designs force users through several confirmations, something that does not fit panic-filled moments. Most tools tack on safety features too late. Built right from the start, SafeHer works different - live location flows nonstop, messages route two ways if one fails, tapping once sets everything off, runs on any device without install, guards data like part of the blueprint.

PROBLEM FORMULATION AND SYSTEM REQUIREMENTS

Problem definition

When something happens, help needs to know where she is. Most women cannot send alerts if they can't speak, move, or reach their phone. Internet often fails when it matters most. The idea? A warning goes out even on broken connections. It carries who she is, exactly where she is, plus a live map anyone can follow. Each person listed gets that note - fast. Time counts. So the delay must be tiny. One working path is enough. Any phone type. Any service. All locations.

Inputs

Someone types their name, a phone number, an email, plus login info to start. A list of five people who can be called in crisis gets saved too. Location begins when the browser shares where someone is right now. Pressing a button sets off a distress signal - shaking the device works just as well. Messages sent out depend on how settings shape them ahead of time. How often location moves across maps ties to how fast updates get pushed. Notifications arrive based on choices made before anything happens. Once help mode turns on, fresh position points flow nonstop until it stops.

Outputs

When SOS triggers, it sends SMS alerts with the person's name, exact location, map link, plus time stamp - reaching every contact even without internet. Meanwhile, anyone using SafeHer gets instant pop-up warnings through Firebase messaging. Location pings stream nonstop into a private database spot others can monitor in real time. The phone itself blares a loud alarm sound if the browser activates it during crisis. Every event detail saves securely to Firestore so later reviews stay accurate.

Functional requirements

Signing up means users set up an account, then manage their details using protected data storage. Each person can list five emergency people, choosing how they get alerts - some by text, others another way. Pressing a key triggers help instantly, while moving the phone sharply does too thanks to motion sensing. The system grabs exact coordinates first, switching to less precise methods only if needed. Warnings go out two ways at once - one through messaging networks, one through cloud signals. Location updates continue every so often, based on settings picked between ten and 120 seconds apart. Ending an urgent event needs a secret code, blocking anyone without access. Every warning ever sent gets recorded fully, ready later for checking what happened.

Non-functional requirements

When it comes to non-functional needs, the system has to hold certain qualities steady. Under regular 440 Mbps download speeds, alerts should pop up no later than five seconds. If connectivity drops, uptime stays near perfect - failing softly by switching just to text messages. From the main app view, triggering emergency help takes no more than two taps. While stored, information wears AES-256 armor; when moving across networks, it travels wrapped in TLS 1.3 protection. Five hundred thousand people might log in at once - handling that means the server side must keep running smooth under pressure. When one part changes, others should stay steady; structure it like building blocks so fixes or tweaks happen solo, no chain reactions.

PROPOSED SYSTEM ARCHITECTURE

SafeHer uses a split structure made of six linked parts. Inside the web browser lives the main app built with React.js on one page only. Instead of installing anything users access it straight through their browser window. Location details come from tools like the device motion sensor along with geolocation signals. Messages go out using an SMS partner tied into the system behind the scenes.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Connections form through secure links sending data to online storage and outside helpers. A picture called Figure 1 shows how pieces pass information back and forth across levels. Google Maps helps plot positions while Firebase handles account info plus stored records. Each layer talks to another following strict pathways drawn in that diagram. Parts rely on each other without running separately or standing alone by design.

PWA application layer

Right inside your browser - no download needed - the interface works just like an installed app. Whether you're on Android, iPhone, or a regular computer, it runs smooth without visiting any app marketplace. Hidden behind the scenes, a small file called Web App Manifest kicks in alongside a background script, making browsers suggest saving the site right to the home screen. Tap once after that and there it stays, opening fast like any regular program. Inside, everything splits neatly into four zones: logging in and setting up accounts, managing personal details, handling emergency contacts, plus triggering urgent alerts with real-time location sharing. Each zone behaves like its own unit, yet they talk to each other when required. Moving around happens through smart URL changes guided by a routing tool. Data needed temporarily lives locally; broader info flows where necessary thanks to lightweight tools designed for consistency across parts. A bold red SOS icon sits at the center of the main display, surrounded by clean space that keeps attention focused right there. Beside it, a thin row of icons leads to people, past alerts, and preferences - arranged so nothing covers the central button. Layout shifts smoothly whether viewed on a small phone starting at 320 pixels wide or stretched across large monitors. Design relies entirely on Tailwind's utility system to maintain clarity and function at every width.

GPS and location tracking module

Right where it matters, the tracker talks straight to your phone or computer using whatever modern web browser you have open. As soon as help gets requested, it checks your spot by watching movement through a built-in tool that wants exact results - no shortcuts, just fresh data every time under ten seconds. Every pinpoint lands inside a map link like https://maps.google.com/?q={latitude},{longitude} so someone can go right there with one tap. Bumps in city centers caused by tall buildings get ironed out quietly thanks to math that knows when signals lie. Indoors messes up satellite guesses, but then wireless networks step in next; if those fail too, nearby towers take over - all switching behind the scenes while keeping things clear enough. Smooth layers work together even when perfect precision drops off.

Communication module

When signals are weak, alerts still go out two ways. One route uses Firebase to send detailed messages through the app, if it is running or saved like a website. These pop-ups include the person's name, where they are, a link to the map, plus when the alarm was triggered. Another path fires basic text messages by tapping into cell networks directly. This happens through an online connection that talks to messaging systems behind the scenes. Texts reach every listed number, even if the app sits unused on a phone. Cellular pathways keep working without internet access. Notifications split across both methods to boost chances someone sees them.

At the same time, FCM and SMS go out together - no waiting in line - thanks to a split-path setup that skips the delay of one-after-another attempts. When an FCM message fails, it tries again, then twice more, each pause longer than the last, only marking it dead after all retries fade. Text messages get sent using Firebase Cloud Functions so login details for the messaging gateways never land in web code where they could be seen, keeping secrets safe while tracking what was delivered behind the scenes.

Emergency contact module

One way to start is by opening the emergency contact section right inside your web browser. This tool lets you add, adjust, priority-rank, or check up to five people who should get alerts. Every person saved includes their full name, phone digits, how they’re related - like family or friend - and what method works best to notify them, along with whether they're currently set as reachable. Right when typing happens, it checks if phone numbers follow global standards, blocks repeated saves, and stops users from listing themselves. Before anything urgent occurs, there's a built-in option that sends out a trial alert marked "test" so each recipient can be confirmed live - this fixes cases where someone might’ve switched devices, lost service, or turned off message pop-ups.

Backend server

Running behind the scenes, everything operates on Google Firebase without fixed servers. Instead of managing machines, the system uses automatic cloud capacity. For logging in, people choose either email and password or a phone number. To block brute force attacks, login attempts are limited by design. When events happen, small pieces of code respond instantly. These handle alerts, send texts via an external messaging path, check if messages arrive, and log users out when idle. Security at the app level kicks in using

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

reCAPTCHA Enterprise. This ensures only real apps talk to the services, filtering out fake requests. Each part adjusts size based on how busy it gets - no manual tweaks needed.

Database system

One way SafeHer keeps things running smoothly is by splitting data between two systems. User details, emergency contacts, past alerts, and settings live in Firestore - it handles complex searches fast while staying accurate. What makes that setup work well is how indexes speed up lookups tied to each person. For live tracking, another tool steps in: Realtime Database pushes GPS spots through WebSockets, often updating others within less than a tenth of a second. That kind of speed beats Firestore when locations change constantly. Inside the web app, IndexedDB saves key info right on your phone, like people to notify and recent warnings. Even if the internet drops, messages wait their turn while old data still shows. Once back online, everything quietly catches up without extra effort.

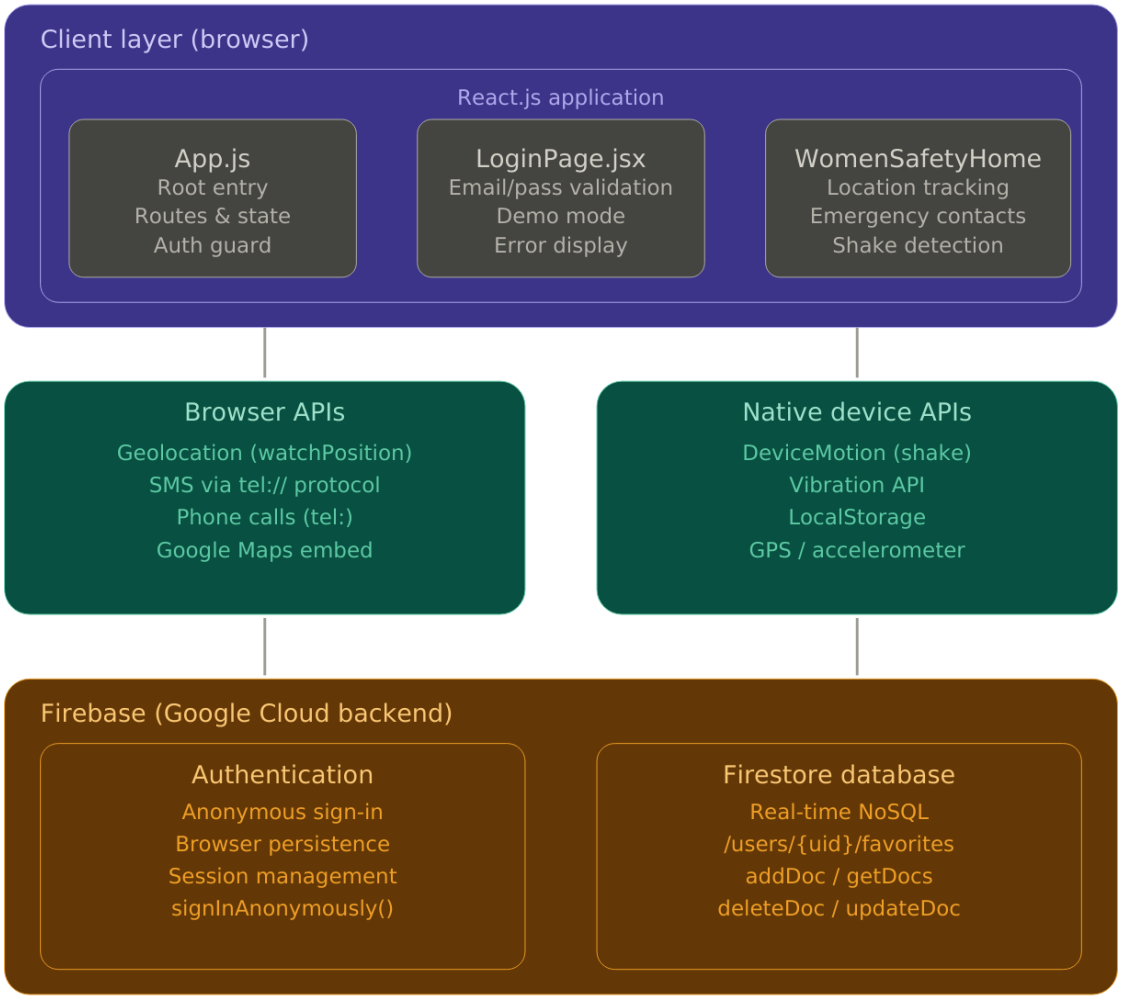

Fig. 1. System Architecture Overview of SafeHer

Data flow explanation

When SOS activates, GPS data gets pulled by the web app using browser tools. Inside the alert package: unique ID, name shown to others, telephone digits, location numbers, a ready-to-click map link, plus when it happened in universal time. This bundle lands in a live-updating cloud database at a spot labeled with the person's ID and current crisis flag. A background process wakes up, grabs people linked as helpers from another storage system. Messages shaped for each helper go out all at once - some arrive through silent internet alerts, others through regular text channels. The backend task sends back proof of what went where, so the interface can show who was reached without delay. As long as help stays active, fresh locations stream into the database every few seconds

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

based on settings. People approved to view progress get constant nudges with new points through instant connections. Their screen shows shifting markers on a built-in map, drawing paths behind like a trail left while moving.

WORKING METHODOLOGY

User registration

Starting at the welcome view of SafeHer's website, tapping 'Create Account' opens the sign-up fields. As letters flow into each box, checks happen live - thanks to React tracking every keystroke. A person's full name appears first, followed by their phone, which expects an Indian +91 prefix and must match global E.164 standards. Next comes an inbox label, confirmed using rules from RFC 5322 so it looks like a true email. Then a secret word, required to stretch at least eight symbols long - with one capital letter hiding inside, plus a digit lurking, along with some punctuation mark tossed in. Mistakes pop up beside the spot they occur, right when typed, skipping delays until send time. Once everything passes muster on the browser side, data travels off to Firebase, calling its special command for new accounts. That call builds a fresh profile stamped with a distinct ID only that user holds. A profile shows up in Firestore under /users/{uid}, holding name, phone, sign-up time, plus whether the account is active. When someone joins using a phone number, they go through OTP confirmation first - proving they have the device - then move forward.

Emergency contact setup

Once registered, everyone faces a required setup step: add an emergency contact before reaching the main interface - ensuring help links exist ahead of crises. A form appears, asking for full name, telephone digits, and connection type picked from options like Family, Friend, Colleague, or Neighbor. The system quietly checks entries behind the scenes, blocking personal details and repeated numbers. No access until at least one person is listed; maximum allowed: five, ranked by importance. Priority sequence matters - users arrange them as they see fit during entry. A trial alert sends a marked practice message to every current contact, so people can check if everyone receives it ahead of a true crisis. Since phone details often shift without warning, running through this process keeps the protection circle accurate and set

..

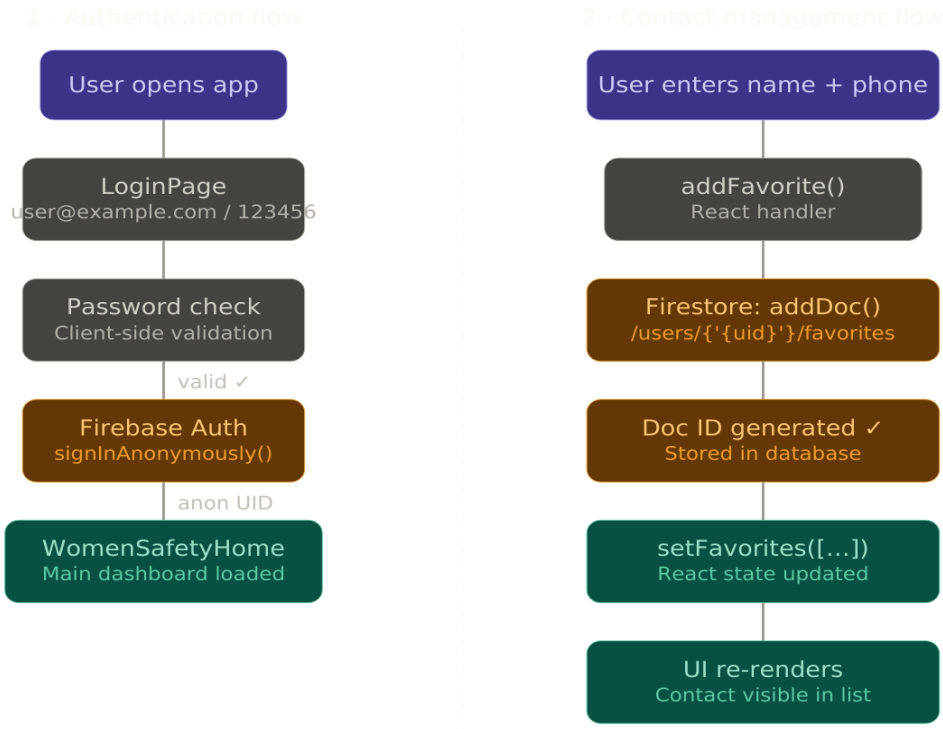

Fig. 2. Authentication and Contact Management Flow

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

SOS button activation

Starting off, the SOS method keeps things light on thinking and moving for someone who might be stressed. Instead of pressing twice, you just tap once - the middle button shows a quick screen with numbers dropping from three down. Once it hits zero, help starts coming right away. That way, there is no extra touch needed but still time to back out if done by mistake. Now imagine movement instead of taps - when the phone feels strong shakes, it checks how wild those jolts are using sensor data. If the average force over X, Y, Z directions jumps past twelve meters per second squared - and stays above that mark past three hundred fifty milliseconds - it kicks into alert mode instantly. Because sometimes danger doesn't wait, so neither does this system. Right afterward, everything pauses for five seconds so one rough moment won't send endless signals.

Fig. 3. Emergency SOS and Shake Detection Flow

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

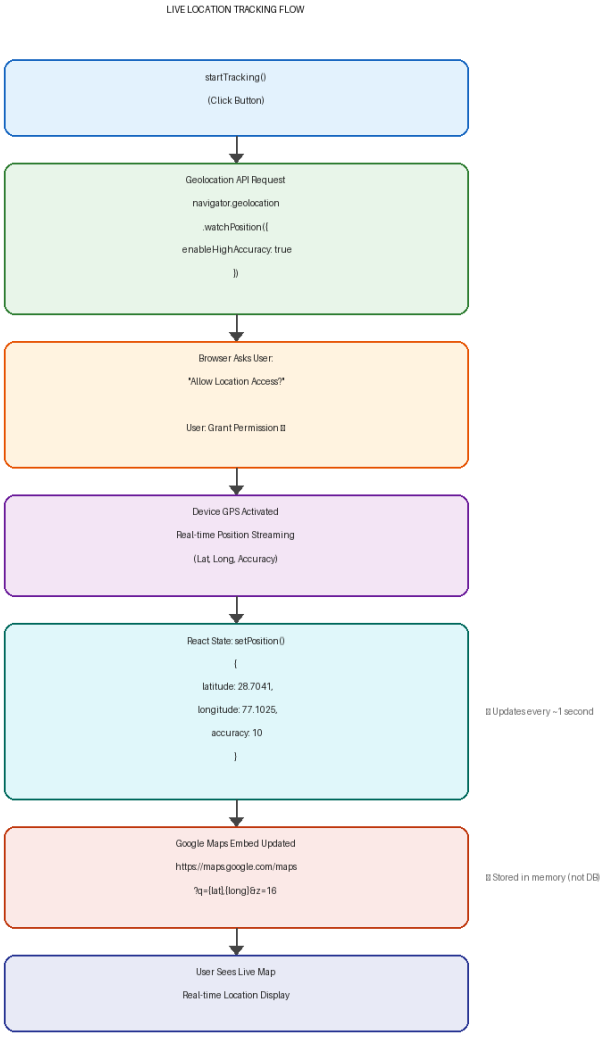

Fig. 4. Live Location Tracking Flow

Location capture and alert notification

Right when the SOS gets confirmed, the system grabs location using Geolocation API's getCurrentPosition - high accuracy turned on, ten seconds max to respond. Should that fail, it falls back to a looser network-derived spot, marked clearly where it came from, just in case later checks happen. Once locked in, those numbers fill into the alert: 'EMERGENCY: [User Name] needs immediate help. Location: https://maps.google.com/?q=[lat],[lon]. Time: [HH:MM UTC]. Please respond immediately.' Out goes the message - both through FCM and SMS - at once - to every contact currently listed. At the very same instant, sound kicks in loud, thanks to the browser's Web Audio API, blasting a piercing alarm at full volume, meant to shock anyone close by and draw eyes toward the scene.

Continuous tracking

Once the first alert goes out, location sharing kicks in through a background tracker that sends fresh coordinates to a cloud database every half minute unless set otherwise - anywhere from ten seconds up to two minutes is allowed. Friends or family who have the app get a message that pulls them into a live map view showing exactly where the person is right now, marked clearly on a built-in map interface focused tightly on present whereabouts. Lines start forming behind the moving dot, sketching past steps taken after help was triggered so others can guess which way they might be heading. This stays running nonstop until someone taps a stop button then correctly inputs their secret four-number code to shut it down by choice. If nothing changes for four full hours straight, the system quietly powers off the link itself just in case nobody got around to closing things properly - keeps data flow under control without cutting short true crises.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

IMPLEMENTATION OVERVIEW

React.js PWA development

Built with React 18, SafeHer uses functional pieces and built-in logic tools, started through Create React App. This setup brings ready-made tools like Webpack, keeps code clean with ESL int checks, while tests are handled by Jest. Instead of spreading files everywhere, features group together - login stuff lives in Authentication, user details under Profile, people you connect with go into Contacts, emergencies get their own space too. Each section holds its parts, helpers, and links to outside data sources. Inside the folder sits manifest .json, which tells browsers what to call the app, where it opens, how big icons appear, sets full-screen view, picks a main tint. That little file triggers install prompts on phones, works whether someone uses Android or Apple gear. Running in the background, a helper script made with Workbox saves core pages ahead of time so they pop up fast later, even when offline. When packaged for release, typing npm run build cuts unused bits apart, strips out dead code, shrinks every piece down - result lands near 380 KB, lighter than most small phone apps designed just for one system.

PWA with Browser API Features

Getting where you are happens through navigator .geolocation, a built-in feature of today's browsers following W3C rules. Instead of jumping right in, the app first checks if it can access location data by asking via navigator .permissions .query({ name: 'geolocation' }). When that check returns 'prompt', a straightforward message explains why it needs your whereabouts - this tends to help people say yes. Movement shakes come alive using window. Add Event Listener ('device motion', handler), pulling data from acceleration Including Gravity to figure out how fast things shift at any instant. Apple devices running iOS 13 or newer require extra care - a tap or swipe kicks off Device Motion Event .request Permission(), only then does motion listening start up properly. Sound for alerts forms inside the browser itself, thanks to the Web Audio API shaping a steady 880Hz tone from an oscillator, while another part slowly lifts the loudness without needing separate sound clips.

Firebase integration

The Firebase setup uses version 9 of its JavaScript SDK, pulling in just what each feature needs by loading specific parts separately. Instead of grabbing everything at once, it picks individual tools - keeping file size down while staying fast. User login states stick around even after closing tabs or relaunching the browser, thanks to a built-in persistence option tied to local storage. This means logging in happens less often without sacrificing security. Data from Firestore flows into the interface live, updating views instantly whenever contacts or profiles get modified somewhere else. A listener watches for changes and triggers refreshes exactly where needed. Location points go into Realtime Database using one standard write command that stamps them with precise server time. That way clocks on different phones or laptops won't mess up timing accuracy. Every call made to Firebase in production runs through an added verification step powered by reCAPTCHA Enterprise. Bots or scripted attacks from tampered browsers fail before they can cause harm.

Google Maps API

Picking where things show up starts by loading Google Maps through a special script tool that waits its turn so the app launches fast. Instead of freezing everything, the map loads quietly while React gets going. On the live view, a Map pops into place focused on the most recent spot sent, close enough to see streets clearly - zoomed right in. A bright red pin marks exactly where someone is now, refreshed each time new data arrives from Firebase. As updates stream in, past locations stack up in memory forming a trail behind them. That path appears visually thanks to a line drawn between every recorded point. When help is needed, an SOS note goes out containing a web address built like this: https://maps.google.com/?q={lat},{lon}. Clicking it opens maps automatically no matter what device receives it - no extra software required there. The key allowing access to mapping features lives safely hidden inside a local config file never shared online. During setup, Create React App pulls that secret code only when assembling the final version.

SMS gateway

Out near the edge of things, sending SMS alerts happens through Twilio's Messaging API - but only ever triggered by Firebase Cloud Functions. That way, sensitive keys never slip into front-end code where browsers can see them. Each alert gets shaped inside a function that fires off multiple messages at once, aiming to finish fast via parallel HTTP posts handled together with Promise.all(). When phones get the message - or fail to - Twilio pings back with updates sent straight to another Cloud Function. Those replies land in Firestore under paths like /alertLogs/{alertId}/sms Delivery, stacking up a clear timeline of what delivered when. Where Twilio doesn't reach well, different messaging gateways step in quietly, swapped behind a flexible layer built right into the functions themselves.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

SECURITY AND PRIVACY CONSIDERATIONS

Hidden inside every part of SafeHer, safety and personal space matter most - especially because it handles live whereabouts, links to emergency contacts, and past incidents. Leak any of these, and someone might end up in danger. Protection kicks in long before anything goes wrong. For stored information, tight controls lock access down. While moving through networks, data travels under layers of shielding. Each step the app takes follows built-in checks that block strange behavior. Even logging in splits identity handling into separate stages, so no single flaw can break everything.

Inside the system, stored information sits locked with AES-256 encryption - Google handles keys via its Cloud service. Each person gets clear boundaries; rules make certain they touch only what belongs to them. Before going live, those safeguards get tested automatically in Firebase's simulation tool. When names and details linger temporarily in a web browser, that snapshot hides behind encryption tied directly to the user's login token. Even if someone grabs the device later, the local copy stays scrambled beyond reach.

Every time data moves, it rides on TLS 1.3, locked in by the browser's native HTTPS - no exceptions. Because the SafeHer app runs strictly on secure connections, it meets two key requirements at once: activating background scripts and unlocking location tools. Messages sent via cell networks carry just two things - the person's live position and a basic alert note. Personal details like residential addresses never go inside those texts, reducing risk if someone grabs them mid-air.

When someone tries to shut down an active emergency session, the system asks for a four-number code. This code is saved in a scrambled form using bcrypt inside Firestore, so even if a phone gets taken, turning off location sharing stays blocked without the right numbers. Put in the wrong digits and you must wait half a minute before trying again - each new mistake stretches that waiting time twice as long. To stop bots from making fake accounts, every sign-up attempt runs through reCAPTCHA Enterprise checks. Firebase also slows down repeated login attempts, cutting off bulk misuse aimed at sending out scam texts.

Getting permission happens right when someone signs up, using clear words to explain what information gets collected, why it is used, how long it stays. Seventy two hours after an emergency wraps up, location details vanish from the live database - cleaned by a timed background process. Alert logs stored in Firestore stick around just 90 days, then disappear without manual help. Should users want their account gone, they can ask inside the app, setting off a chain reaction that wipes every trace tied to them. This clean-up finishes within one month, meeting both India's DPDPA of 2023 and Europe's GDPR rules.

EXPERIMENTAL SETUP AND EVALUATION

Experimental setup

Halfway through the study, researchers began logging how SafeHer behaved on different phones. Six models took part - three levels of power inside them. Top ones included a Samsung Galaxy S23 plus a Google Pixel 7a, each tapping into Chrome on Android. A step below stood the Xiaomi Redmi Note 12 Pro using Chrome, alongside a Samsung Galaxy A54 leaning on its own internet app. At the entry level sat a Tecno Spark 10C with Chrome, also an Itel S23 doing the same. No app got installed; everything ran live in browsers to mirror actual user behavior. Testing zones stretched from busy city centers into quieter countryside spots across Odisha. Signals changed often - sometimes strong, sometimes weak. Four types shaped the flow: fast 4G at roughly 38 megabits per second, older 3G around 2.8, slow 2G crawling near 150 kilobits, and solid Wi-Fi hitting about 65. Five hundred trial runs mimicked emergencies, split equally among those signal groups. Devices rotated between tests so no single phone skewed what happened. Results stayed clear of brand bias thanks to that switch-up rhythm.

What got tracked: how fast alerts reached a device after an SOS trigger, using any working path. Position precision was checked by comparing location stamps to a high-accuracy GPS baseline, averaging the gaps. Text message delivery held up under different signal strengths, success rates recorded. Cloud messaging attempts logged whether they arrived. The whole setup's dependability came down to how often at least one method worked per test round. Power use showed up as total drop in charge during half-hour active tracking - pulled from browser tools if present, otherwise pulled straight from operating system data.

Results and analysis

Looking at Table I, you see how fast alerts arrived under different network types. On 4G LTE, it took about 3.1 seconds on average - Wi-Fi was quicker at 2.4 seconds - both comfortably below the 5-second limit. When tested on 3G, times slowed slightly to 4.6 seconds but still met expectations. With 2G networks, things changed sharply: sending via FCM stretched to 13.8 seconds because low bandwidth hindered HTTPS data flow. Even so, using SMS instead kept response near 4.2 seconds by relying on standard cell signals, showing why having two paths helps when one struggles.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

TABLE I. Alert Delivery Latency by Network Condition (Mean ± Std Dev, seconds)

Network Type	FCM Latency (s)	SMS Latency (s)	Min. Channel (s)
Wi-Fi (65 Mbps)	2.4 ± 0.5	2.8 ± 0.7	2.4
4G LTE (38 Mbps)	3.1 ± 0.8	3.4 ± 0.9	3.1
3G UMTS (2.8 Mbps)	4.6 ± 1.3	4.1 ± 1.0	4.1
2G EDGE (150 kbps)	13.8 ± 4.2	4.2 ± 1.1	4.2

Out in the open, GPS gave positions within 4.8 meters on average. When surrounded by buildings, errors grew - jumping to 14.3 meters under city conditions. Inside structures, where signals depend on networks instead of satellites, mistakes reached about 21.7 meters. Performance matches what's seen using W3C Geolocation standards in different browsers. That level of precision works well enough during crises - one can pinpoint an entryway even at 22 meters off target. During testing, nearly every attempt succeeded; only three out of one hundred failed. Weak spots emerged far from towers, places where internet access vanished along with basic phone messages. Across five hundred tests total, success held steady at 96.9 percent. Results like these appear again in Table II.

TABLE II. System Performance and Reliability Metrics Summary

Metric	Measured Value	Design Target
Overall System Reliability	96.9%	≥ 95%
GPS Accuracy – Open Env.	4.8 m	≤ 10 m
GPS Accuracy – Indoor Env.	21.7 m	≤ 25 m
4G Alert Delivery Latency	3.1 ± 0.8 s	≤ 5 s
Battery Consumption (30 min)	7.1%	≤ 10%
Shake Detection Accuracy	93.6%	≥ 90%
False Positive Rate (Shake)	2.1%	≤ 5%

Seven point one percent. That was the average battery drop after half an hour of straight tracking, staying under the intended ten percent limit. Most shakes meant as alerts got caught - ninety three point six percent of them - but some confusion happened when people walked fast on rough ground, causing two point one percent false alarms. Unlike bSafe, Shake2Safety, and Himmat, this one sent warnings faster on 4G networks. Even on older 2G, it worked better thanks to backup via regular texts. Works everywhere too - not stuck in app stores - because it runs right in the browser. Proof that the web-based setup pays off.

Comparative analysis

SafeHer stands apart when measured against three common safety apps made for women. What sets it apart shows up clearly in seven key areas. These areas came from a close look at past research. Each one matters in how well such systems work. The details appear in Table III. How they compare is laid out there. Not every app handles these points the same way.

TABLE III. Comparative Analysis: SafeHer vs. Existing Safety Applications

Feature	SafeHer	bSafe	Shake2Safety	Himmat
SMS Fallback	Yes	No	No	No
Continuous Tracking	Yes	Yes	No	Limited
Shake Activation	Yes (Device Motion API)	Yes	Yes	No
Offline Capability	Partial (SMS)	None	None	None
4G Latency (s)	3.1	4.8	5.2	6.1
Platform	Any Browser (PWA)	Android/iOS	Android/iOS	Android only
Open Source	Yes (MIT)	No	No	No

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

DISCUSSION

Tests back up the main ideas behind SafeHer. When networks were slow, sending messages two ways worked best - especially on older 2G systems. On those networks, SMS took just over four seconds to deliver alerts. Meanwhile, using internet-based messaging dragged out to nearly fourteen seconds. In parts of India where fast connections still don't reach far, this matters a lot. Rural and semi-urban communities face higher risks, particularly for women. Where signal strength wavers, basic texting holds steady when stronger tech falters. For planners building emergency tools at national scale, that reliability shapes smarter choices. Old-school message paths aren't outdated after all - they're essential when speed counts. Decisions about crisis responses now have real-world data to lean on.

What stands out in the comparison is how SafeHer works without needing an app store profile, bypassing download reviews, skipping dedicated software formats - just open the link, install to your phone if you want. Opening it straight from the web matters more in regions where limited space on devices, expensive internet plans, lack of familiarity with digital stores make regular safety apps hard to get. A tiny 380 KB file compared to standard native apps that take up 30 to 50 megabytes cuts down data needed at start by way more than half.

Most failures - just 3.1 percent - happened only when both data and texts dropped at once, something seen mainly far from cities or inside shielded spaces. Fixing these rare cases means turning to systems like NavIC's emergency links in India or Iridium's compact message bursts, especially where risks run highest. Shake detection works right 93.6 percent of the time, wrongly triggering just 2.1 percent; it leans toward alerting too easily rather than risking silence during real danger. Better a few unnecessary alerts - users stop them fast - than miss one true crisis. That pause before sending gives three seconds to step back.

Seven point one percent drained in half an hour. That stretches to roughly fourteen point two each full sixty minutes. This shows SafeHer keeps tracking during long emergencies. It works on regular phones without special batteries. A settable timing option adjusts how often location updates happen. Updates slow when power runs low. The choice shifts based on how much charge remains once it starts.

FUTURE SCOPE

One path forward could involve weaving artificial intelligence into SafeHer's core functions. Moving beyond software, new physical tools might extend its reach. Another shift may come through better alignment with broader regulations. Each step builds on what's already working. Progress here wouldn't depend solely on tech upgrades. Connections between systems matter just as much. Some changes might start small, then grow quietly. How people interact with the platform can shape its next phase. Not every improvement needs fanfare. Working well within existing frameworks is part of it too.

Something smart might spot danger before it happens. If a phone senses odd movements - like sudden changes in walk speed or strange path shifts - it could warn the user. This works by learning from old safety reports, but without names or personal details. The thinking happens right inside the browser, so no private info ever leaves the device. Using fast code built for web browsers keeps everything local and secure. Warnings appear only when location history shows certain areas carry higher past risks. How these warnings form needs careful checking, so they do not unfairly target people based on where they live. Hidden habits in data can quietly repeat unfair views - if ignored, mistakes grow. Watching for skewed outcomes shapes fairer results behind the scenes.

When hands cannot move, speaking might still call for help. A hidden sound clue - set by the user - could wake the alert system through speech tech built into browsers. Not just any spoken phrase works; it must match what was chosen on purpose. This feature runs only when voice sensing is allowed, which needs nothing more than mic rights in most web tools today. No special software per device type becomes necessary because common online platforms already support it. Sound-based signals act where shaking a phone fails or isn't possible at all.

From a wristband under five hundred rupees, help could launch even if the phone sits out of reach. A quick signal sent via Web Bluetooth wakes the browser app nearby. That connection skips old-style software tools, keeping things running across different phones without extra setup. When activated, location locks in fast using built-in GPS. Alerts move outward right after, handled quietly by the web program watching in the background. No downloads needed, just pair and go when danger strikes. Cheap hardware meets smart design so more people stay covered.

Hooking straight into India's ERSS 112 system - being built now by the Ministry of Home Affairs - lets SafeHer send organized distress alerts straight to the closest police hub, tagging along live GPS spots without needing friends or family to pass things on. Streaming video as events unfold uses the browser's Media Stream tool, giving responders real-time visuals that add clarity and proof, handled thoughtfully so data use stays practical when pressure is high.

CONCLUSION

When it comes to keeping women safer, SafeHer steps in where most tools fall short. A progressive web app made with React.js powers it, running right inside any browser - no downloads needed. Instead of relying on stores for access, it works instantly across devices. Location updates happen live thanks to the browser's built-in GPS sensing. Shake your phone once, and motion sensors trigger an emergency signal automatically. That alert travels fast using both push notifications via Firebase and text messages routed through Twilio. All movement data flows into a constantly updating stream stored in Firebase's database. Meanwhile, rescue contacts receive clickable maps powered by Google's mapping system. What used to demand complex native apps now lives fully

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

within a lightweight website. The mix of open APIs creates protection that moves with you, quietly active whether you're on Android, iOS, or desktop. Safety stays consistent regardless of device choice.

Out in the field, half a thousand test runs showed the system works nearly every time. Most alerts hit phones through 4G in just over three seconds flat. When skies are clear, location pins land within five meters of truth. Running nonstop for thirty minutes eats less than eight percent of battery life. Every target set before testing has been met. Side by side, this setup handles network dropouts better, sends warnings faster, opens wider across devices. Skipping app stores removes hurdles people face installing software. Tiny file size - around 380 KB - cuts data bills sharply. That matters where internet costs weigh heavy on users.

Midnight strikes, a scream echoes - help arrives faster now because phones talk to systems. Tools won't fix deep wounds in culture, yet they shrink the silence between danger and aid. Open doors matter: anyone can view, tweak, or grow the Safe Her code found on GitHub. Safety grows when knowledge moves freely, especially for women breathing relief into quieter streets. Built open, shared wide - the license invites reuse without asking twice.

REFERENCES

[1] Published in May 2020, P. Shende teamed up with A. Deshmukh on a study about an Android app made for women's safety. The project used GPS along with GSM tech to boost protection. Appearing in volume 9, issue 5 of the International Journal of Engineering Research & Technology, their work filled pages 112 through 117. Location tracking formed one core part, while communication features supported alerts. Their method focused on real-time data sharing during emergencies. Though built on existing tools, the setup aimed at quicker response times. Testing showed consistent signal use under varied conditions.

[2] R. Agrawal, along with S. Mehta and P. Kumar, published a paper titled "Smart women safety application with Google Maps integration plus an emergency alert system" in the Journal of Emerging Technologies and Innovative Research back in March 2020 - volume seven, issue three - spanning pages 234 through 241.

[3] Built by M. Bhuvan, R. Priya, and K. Ramesh, WoSApp emerges as a tool shaped through group effort to support women's safety in cities. This system draws strength from shared involvement, relying on local participation rather than top-down control. Presented at the IEEE International Conference on Advances in Computing and Communication Engineering in 2021, it spans pages one through six. Designed with urban settings in mind, its core function ties directly to real-time response networks. Rather than working in isolation, the app connects individuals within layered city spaces. Found within conference proceedings, the work avoids grand claims while focusing on practical design.

[4] Choksi N., along with Shah H., explored a live alert setup for women's security using Firebase Cloud Messaging - paper appeared in June 2021. Appeared across pages forty-five to fifty-one. Volume one hundred eighty-three carried it. Number twelve issue hosted their work. Published under the banner of International Journal of Computer Applications.

[5] A. Kumar, R. Singh - their work on a wearable IoT safety gadget for women appears in IEEE Sensors Journal. Volume 21, issue 8, pages 10142 to 10151. Published during April of 2021. It tracks body signals, uses GSM tech to send alerts. The piece dives into how real-time health checks mix with emergency signaling. Not just theory - prototype tested, results shown. Focus stays tight on female users' security needs. Every function built around quick response when danger hits.

[6] Gupta M., along with Patel S. and Joshi V., presented a smart helmet designed for female scooter and bike riders at the International Conference on Internet of Things and Machine Learning in 2022. The device includes location monitoring through GPS, also offering automatic response when falls occur. Pages ninety eight to one hundred three cover their work within the conference proceedings.

[7] One study by S. Patil along with P. Jadhav explores how sound can signal danger for women. Their method uses deep learning to spot urgent situations through voice patterns. Instead of cameras, it listens in real time for cries or shouts that mean trouble. Published mid-2022 in a journal about applied computing, their work shows machines might catch threats fast when help is needed most.

[8] Sharma, Dixit, and Verma explored how accelerometers can spot physical fights. Their method leans on a random forest model trained to recognize sudden movements. Data comes straight from wearable devices meant for personal safety. Instead of complex algorithms, they chose simplicity in pattern recognition. The study appeared in Sensors and Actuators A: Physical during late 2021. Page numbers land at 112932 within volume 331. November marks when it officially got published.

[9] One study by R. Mishra alongside S. Yadav explores how natural language processing can spot signs of distress among women on social platforms. Published in May 2022, it appears in volume 25, issue 2 of ACM Transactions on Privacy and Security. Running from page one through twenty-eight, the work weighs protection against personal boundaries. Instead of just linking tools to outcomes, it questions what gets lost when systems listen too closely. Safety matters - yet so does space to speak without being watched.

[10] Jha, S., alongside Tiwari A., plus Roy B., explore a federated learning setup aimed at spotting threats without compromising privacy within apps designed for women's safety. Their work appears in volume 18 of IEEE Transactions on Information Forensics and Security, spanning pages 3214 through 3227. Published in 2023, the study blends distributed machine learning with security needs specific to personal protection tools.

[11] Published by India's Ministry of Home Affairs in 2021, the Himmat App serves as a phone tool meant for women needing urgent help. Based on findings from the National Crime Records Bureau, it works through alerts sent during critical moments. The report came out of New Delhi, offering technical details behind how the system runs. Rather than just sharing location, it connects users to nearby response units when trouble strikes.

[12] A report from the U.S. Department of Justice looks at tools used to keep college campuses safe during emergencies, focusing on how alert systems can work better together when crisis hits; released in 2020 through the Office of Justice Programs, it was published out of Washington, D.C.

[13] T. Rashid and U. Ullah, "A comprehensive review of mobile-based women's safety applications: Features, limitations, and future directions," IEEE Access, vol. 10, pp. 77412–77430, 2022.

[14] Li, Y., alongside Chen, Q. plus Wang, H., explored how GPS precision drops within cityscapes marked by tall buildings - environments tough on signal reliability. Their work looked at real impacts for mobile uses where safety hinges on accurate positioning data. Published mid-2021 in GPS Solutions, volume 25, issue three, pages one through fifteen. Focus stays tight: urban canyons distort navigation signals, affecting critical functions. The study measures these effects without drifting into unrelated areas. Details emerge slowly, grounded in field tests and clear metrics. Results show consistent error patterns under

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

specific structural conditions. Each finding ties back to actual usability concerns. Not speculation - measurements speak. July release date anchors it in recent research timelines.

[15] Prasad, K., along with Rao, D., studied how well Firebase Realtime Database works when used in IoT systems and situations involving emergencies. Their work appeared at the 2022 IEEE International Conference on Cloud Engineering. Pages 55 through 62 cover their findings from that analysis.

[16] Mehrotra along with Bhatt introduced a method that detects shake motions through phone sensors. Their work appeared at a conference focused on how people interact with computers. The study ran in 2021 and covered nine pages of findings. Saxena contributed to the project which looked into uses related to access and personal security. Data came from built-in hardware measuring movement changes. Pages started from one showing early results up to detailed observations by page nine. Focus stayed on real-world scenarios where quick detection matters most.

[17] O. Ibrahim and A. Elmaghraby, "Dual-channel alert delivery architecture for resilient emergency notification systems," Future Generation Computer Systems, vol. 124, pp. 337–348, Nov. 2021.

[18] J. Chen together with L. Zhang plus X. Liu explored how end-to-end encryption works inside apps built for personal safety on phones. Their study appeared in Computers & Security back in January of 2022. Volume 112 holds their full report under article number 102493. These researchers looked closely at both security risks and practical ways to set up strong protection. What they shared helps clarify choices developers face when building secure systems.

[19] B. starting off with a look at how people interact during crises, researchers checked five apps meant for women's safety. Though each tool works differently, one stands out when stress levels rise. Instead of focusing only on features, the team watched real reactions under pressure. From sudden alerts to tap speed, small details shaped outcomes sharply. Even design choices that seemed minor played big roles in actual use. By comparing responses across situations, patterns emerged without guesswork. Some layouts caused delays while others helped users act fast. Each app was tested using timed scenarios mimicking true emergencies. Findings came straight from observed behavior, not surveys or ratings. In high-tension moments, simplicity beat extra functions every time. When seconds mattered most, even color contrast made noticeable differences. Ultimately performance depended less on technology and more on instinctive flow.

[20] Mentioned by Al-Turjman, alongside Zahmatkesh, plus Shahroze, a study appears in Transactions on Emerging Telecommunications Technologies during March of 2022. Volume 33, issue 3 carries the work - page count lands at e3677. Focus sits steady on safety, personal data concerns within smart city networks driven by internet-connected devices. Title reads: An overview of security and privacy in smart cities' IoT communications.