International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Smart Innovations in Computer Science and Applications
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

# EasyCare: A UI/UX-Centered Mobile Application for Streamlined Home Repair and Maintenance Services

Vrushika Urade
Department of Science and Technology, G.H.Raisoni Skill Tech University, Nagpur, India

## ABSTRACT

My cousin called me last year saying her AC stopped working and she had no idea who to call. She tried three numbers she found online — one didn't pick up, one asked for advance payment before even seeing the problem, and the third quoted something on the phone that doubled by the time the job was done. No receipts. No accountability. Just frustration. Honestly that experience stuck with me and it's a big reason why I started working on EasyCare.

EasyCare is a mobile app I put together for home repair and maintenance — AC work, plumbing, electricals, painting, tiles, that kind of thing. UI/UX wasn't something I tacked on at the end; I started there. My thinking was simple — if someone's pipe is leaking at 10pm, they shouldn't need to figure out the app. It should just work.

Before any wireframe, I sat down with actual people — homeowners, renters, professionals, older folks who live by themselves. Just asked them to walk me through what happens when something breaks. I listened. I took notes. From those conversations I built personas, made rough sketches, then prototypes, then tested them. Every session revealed something that needed fixing. That process of testing, breaking, fixing — that's what the final design came from.

Trust kept coming up. Every person I spoke to mentioned it in some form. You're letting a stranger into your bedroom, your bathroom, your kitchen — that's not nothing. So I made sure the app shows verification status, past ratings, and job history right where you're making the decision. Not buried three screens deep. Right there. Pricing visible before you tap confirm. What you see is what you pay.

When I tested prototypes with users, the feedback was mostly positive — people moved through bookings faster, said they felt more confident, and described the whole thing as less stressful than what they're used to. The technicians I spoke to also liked the idea of having a dashboard to manage their work instead of relying on phone calls and WhatsApp messages. That matters to me — it's not just about making things easier for customers, the workers need a better system too.

**Keywords:** EasyCare, UI/UX Design, Home Repair Services, Mobile Application, User Experience, Service Booking System, Human-Computer Interaction, Digital Service Platform, On-Demand Services, Usability Testing.



## Introduction

My neighbour Priya had water dripping from under her kitchen sink. This was last July. She called four people — two didn't pick up, one said he'd come and didn't, and the fourth showed up, fixed it in forty minutes, and charged her nearly twice what he quoted. No bill, nothing on paper. She paid

because what else was she going to do. I've sat with enough people sharing almost the exact same story that it stopped feeling like bad luck and started feeling like a system that just doesn't work.

Think about what ordering food looks like now versus calling a plumber. On Swiggy you see the restaurant, the rating, the delivery guy's name, a live map of where your order is. You booked an electrician? You called a number. Maybe he picked up, maybe not. Maybe he showed up, maybe not. Nobody tracks anything. Nobody's accountable to anything. My parents dealt with repairs this way. My grandparents did too. Nothing changed. Which is genuinely weird given how much else has.

Urban Company exists. So do a few others. And honestly they've helped. But if you read through their Play Store reviews you'll find the same things repeating — price quoted was not price charged, technician cancelled day-of, couldn't get support when something went sideways. These aren't rare complaints. They fill pages. The apps moved the process online but the underlying problems — transparency, accountability, reliability — didn't go away.

EasyCare came from wanting to do this differently — not just put the same old process on a screen, but actually think about what a person needs when they open the app. What are they feeling right then? What question do they need answered first? What would make them feel okay about confirming a booking with someone they've never met?

What the app actually does: you pick a service category, select your specific problem, choose a time, confirm where you are, see the price, and pay. AC repair, plumbing, electrical, painting, tiles — all covered. No phone calls. No negotiating. The whole thing sits in one place and the price you see before confirming is the price on your invoice.

For technicians — and I want to be clear that this side of it matters just as much to me — the app provides a proper dashboard. Right now a lot of skilled workers operate with no system at all. They get jobs through word of mouth, manage schedules in their head or over WhatsApp, and have no way to show potential clients their track record. EasyCare gives them structured booking management, earnings tracking, and a rating history that actually means something professionally.

This paper covers the research behind EasyCare, how the design developed, what the testing showed, and where things still need work. The broader point I'm trying to make is that fixing a broken industry doesn't require building something complicated — it requires building something that actually fits how people think and what they actually need.


How EasyCare Works: A User Journey Overview

## Literature Review

Before starting the design work I read through research on usability, trust in digital platforms, and mobile design. Some of it confirmed things I already had a feeling about. A few things genuinely surprised me and changed how I approached certain decisions.

The usability literature is pretty consistent on cognitive load — basically, the more mental effort an interface demands, the more likely people are to abandon it. This isn't theoretical. When people are stressed, decision-making gets harder. A person dealing with a broken AC or a flooded bathroom doesn't want to spend mental energy figuring out which menu to tap. They want an obvious path forward. That's what pushed me toward a step-by-step booking flow with one decision per screen, rather than a single long form.

Research on trust in online platforms is really important for this specific context. Several studies found that when credibility signals — ratings, verification badges, service history — are visible and easy to find, users are much more likely to complete transactions. When those signals are buried in submenus, conversion drops even if the information technically exists. In home services, trust matters more than almost any other factor because the technician is physically entering someone's private space. I made sure verification badges and ratings are visible on the booking confirmation screen, not behind an extra tap.

Service design research introduced me to thinking about what users see versus what runs behind the scenes. Good platforms make the backend work invisible — matching, notifications, invoicing all happen without the user needing to manage any of it. That principle shaped how I thought about the flow between the customer-facing interface and the technician dashboard.

Here's something I noticed testing with real people — most of them were holding their phone in one hand, usually standing up, sometimes already dealing with the thing that's broken. Tapping a small button while slightly panicked is harder than it sounds. Text that requires zooming in is annoying enough that people give up. Important info that's below the fold gets missed entirely. These things seem small but they add up. Service apps that skip these basics tend to have very clear signs of it in their low ratings.

I spent a good amount of time going through reviews of Urban Company and similar platforms — not skimming, actually reading through pages of them. Pricing that changed after confirmation, cancellation policies buried where nobody would find them, no clear way to escalate when something went wrong. Every single one of these is a design problem with a design solution. The information exists — it was just placed at the wrong moment, or treated like fine print when it should have been front-page. That exercise was probably as valuable to me as reading academic papers about what to do right.

**Problem Statement**

Home repair in India runs mostly on informal trust — and informal trust breaks down constantly. That's the actual problem. Not that the technology doesn't exist, not that skilled workers aren't available, but that there's no structure holding the whole thing together in a way that works reliably for either side.

When something breaks at home, most people go through the same chaotic process. They dig up an old contact, ask a neighbour, search on Google, try a few numbers. Whatever they find, there's no way to verify if the person is actually qualified. No ratings to check, no job history to look at, no credentials you can confirm. You just take your chances and hope it works out.

Pricing causes more frustration than almost anything else in this space. A technician quotes one number on the phone, arrives, does the work, and then the actual bill is higher — sometimes significantly. Additional parts, extra labor time, a 'service charge' that wasn't mentioned before.

Nothing is in writing so there's no real way to dispute it. Customers feel taken advantage of even when it's not intentional. The system just doesn't support honest, clear pricing — so misunderstandings happen regularly.

Scheduling is another failure point that people deal with more than they should. You block off part of your day, the technician says he'll come between 11 and 1, and at 12:30 he sends a message saying he's stuck somewhere. Or he just doesn't show up and doesn't message at all. There's no system holding anyone accountable. For working professionals managing tight schedules, this is genuinely costly, not just annoying.

Before the visit even happens, communication is usually poor. Most customers don't know the technical names for what's wrong — they just know something isn't working. The technician gets a vague description, comes unprepared, realizes he needs different parts, and has to come back a

second time. Both sides waste time on something that a proper pre-visit conversation would have prevented.

Safety is also a real concern that doesn't get talked about enough. Inviting someone into your home — especially if you live alone — requires a level of trust that the current system can't really support. There's no formal background verification, no way to check credentials, no accountability trail. People who live independently or are in a vulnerable situation have very limited options for protecting themselves in this process.

From the technician side, things aren't much better. Skilled workers — good ones — have no real way to build a professional reputation beyond their immediate circle. They get inconsistent work, have no organized system for managing their schedule, can't track what they're earning in any structured way, and have no platform to showcase their skills. A talented electrician two streets over might as well not exist if nobody in your building has heard of him.

EasyCare is trying to fix all of this at once — not one feature at a time, but by redesigning the entire experience from the ground up, starting from what actually goes wrong and working backwards from there.

Challenges in Current Home Repair Services

**Objectives**

The clearest way I can put this — I didn't start with a list of features and work backwards to justify them. I started with specific problems that real people described to me and worked out what the system needed to do to actually solve them.

First thing was getting a real understanding of what frustrates people about booking home repair services. Not guessing, not assuming — actually talking to homeowners, renters, working professionals, older residents. The same issues came up repeatedly across different types of people: not knowing who you're dealing with, not knowing the actual cost until after, not being sure the person will show up. Those three things set the direction for everything else in the design.

The interface needed to work without instructions. Not 'easy to learn' with a tutorial — just obvious. Someone stressed about a problem at home shouldn't need to explore around to figure out how to book a service. That meant clear service categories, a linear booking flow where the next step is always apparent, and nothing that requires reading explanatory text to understand.

Trust needed to be designed in, not implied. Verification badges visible before confirming. Ratings and job history on the same screen where you make the booking decision. Pricing shown clearly before any payment, not after. These aren't separate trust features — they're responses to the specific moments in the booking journey where users lose confidence and drop off.

Pricing transparency was its own specific goal. The system needed to display what a service costs before you commit. Fixed rates for standard work, honest ranges for more complex jobs. Any additional charges documented digitally. Auto-generated invoice after job completion. This directly addresses the most common complaint about existing home repair services — and it makes the platform accountable in a way informal bookings never are.

The technician side needed to be a real system, not an afterthought. Dashboard with organized bookings, earnings tracking, rating visibility, availability management. A platform where only customers benefit doesn't sustain itself — the workers need to get genuine value from it too, otherwise supply dries up and the whole thing collapses.

Usability testing had to be built into the process with defined metrics. Task completion time. Error rates. Whether users could find specific information — pricing, cancellation policy — without being guided. Satisfaction ratings. The point wasn't to confirm the design was good. It was to find where it wasn't, so those things could actually be fixed before treating it as done.


EasyCare: UI/UX-Centered On-Demand Home Repair

**Proposed System**

EasyCare works as a two-sided platform — one experience for customers booking services, a different one for technicians managing their work. They run on shared infrastructure but each side is designed specifically around what that group needs to do.

For customers, the home screen shows clear service categories — AC Repair, Plumbing, Electrical, Painting, Tile Fitting. Each category breaks down into specific subcategories. Under Plumbing: leak repair, pipe installation, bathroom fitting. Under Electrical: wiring fault, switchboard replacement, fan installation. The structure means you navigate to what you need rather than having to describe it. No technical knowledge required.

Booking is broken into steps — select service, pick the specific issue, choose date and time, confirm address, review pricing, pay. One thing per screen. Progress shown at the top. Each step is narrow enough that there's really only one correct path forward. This was a deliberate choice based on what the usability research showed about reducing cognitive load in stressful situations.

Before confirming, the technician profile is displayed — photo, experience, certifications, completed jobs, average rating, verification badge. This is visible on the confirmation screen itself, not somewhere you'd have to navigate to separately. The placement was intentional. The trust signals need to appear when users are making the decision, not before or after.

Pricing is fixed for standard services, estimated range for complex ones. If something extra comes up during the job, it has to be documented in the app before being added — no verbal add-ons, nothing appears on the final invoice that wasn't approved. After the job, the invoice generates automatically and gets sent to the customer. That paper trail changes the accountability dynamic entirely.

Once a booking is confirmed, tracking begins. Notifications when technician is assigned, when they're on the way, when they arrive. GPS shows their location in real time. This was consistently the

feature users responded to most positively in testing — just knowing someone is actually coming and seeing where they are removes a lot of anxiety.

In-app chat lets customers and technicians communicate before the visit. Share a photo of the problem, describe what's happening, ask what to expect. This pre-visit exchange catches mismatches between what the customer needs and what the technician is prepared for — which is what causes most of the repeat visits and extra costs in traditional setups.

For technicians, the dashboard is the whole system. Upcoming jobs listed with customer details and addresses. Earnings tracked per job and over time. Ratings visible after each completed service. Availability updated easily. It replaces the informal, scattered way most service workers currently manage things — and it gives them something they've never had before: a professional track record they actually own.

After every job, both sides rate the experience. Customer ratings go onto the technician profile and stay visible. This isn't just a quality mechanism — it changes the culture of accountability. When your ratings are public and directly tied to how much work you get, you behave differently.


**System Architecture and Methodology**

The system runs on a three-layer structure. The top layer is what users see — the mobile interface for customers and the dashboard for technicians. The middle layer handles all the logic that runs behind the scenes: matching customers with technicians, processing bookings, sending notifications, managing payments, generating invoices. The bottom layer is the database — accounts, booking records, profiles, transactions, ratings. Payment details and home addresses are encrypted.

Most of my work happened at the interface layer, which is where UI/UX design lives. The methodology was iterative by design — meaning the interface changed substantially based on what each round of testing revealed, not based on what I assumed would work.

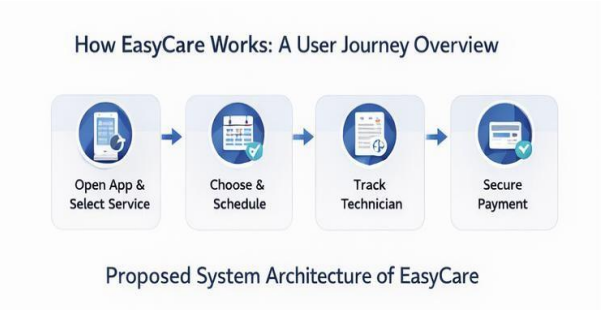
I started by talking to people. Not structured surveys — just open conversations with homeowners, tenants, professionals, and older residents about how they currently handle home repair situations. I wanted to understand the experience, not just collect data points. The same frustrations came up across almost every conversation: vague pricing, unreliable scheduling, no way to verify who you're dealing with. Those three things became the design priorities.

After those conversations I went through existing platforms in detail. Used them, read their reviews, mapped where users consistently ran into problems. Hidden pricing, cancellation policies nobody could find, support that was hard to reach — these weren't random complaints, they had specific design explanations. Pricing information was placed too late in the booking flow. Cancellation policy was buried in terms. Support routing was unclear. Each one became a specific thing to get right in EasyCare.

User personas came from the interview research. Three main types: a professional in their late twenties who needs things sorted fast with minimum friction; an elderly resident who is not very comfortable with apps and needs the interface to be self-explanatory; someone living independently who has particular concerns about safety when letting workers into their home. Journey maps for each persona identified where anxiety peaks and where confidence drops in the current process.

Wireframes first — basic structural sketches to test navigation logic before any visual design decisions. Small group feedback on wireframes, then revisions, then high-fidelity prototypes for formal usability testing. Participants completed specific tasks: book a plumbing service, find the pricing for AC repair, locate the cancellation policy, reschedule an existing booking. I timed each task, noted where people hesitated or went the wrong direction, and recorded satisfaction ratings.

Between rounds, changes were made based on what the data showed — not what I thought might be better. Cancellation policy moved to the booking summary screen because round one showed nobody could find it in the menu. Payment confirmation screen redesigned because its different visual style made users momentarily uncertain whether they were still in the correct app. These were small changes with measurable impact on completion rates.




**Results and Discussion**

Testing gave a pretty honest picture of what was working and where things needed to change. I'll be direct about both.

On booking time — participants completed a full service booking in three to four minutes on average. That's a real improvement over the informal process. Calling around, waiting for callbacks, negotiating a price, confirming over WhatsApp — that can easily take twenty to thirty minutes, and sometimes ends without a confirmed booking at all. Even in a controlled testing environment, that difference in time is meaningful.

Navigation errors dropped clearly between round one and round two. In the first round, subcategory labels were confusing several participants — they weren't sure which option covered their specific problem and kept ending up in the wrong section. After relabelling and adding small icons, almost everyone navigated correctly on the first attempt in round two. That's a direct result of actually watching where people go wrong rather than assuming the logic that made sense to me would make sense to them.

The trust features — verification badges, ratings, job history on the confirmation screen — got the strongest responses. Several participants said unprompted that seeing those details made them more comfortable confirming the booking. One person said it felt more legitimate than what they were used to with other apps. That's exactly the response I was going for. Trust doesn't happen automatically — it has to be earned through specific design decisions, and those decisions have to appear at the right moment.

The GPS tracking feature got positive responses across the board. Users consistently described it as removing the anxious waiting feeling — 'at least I can see he's actually on the way.' That's a small thing technically but a significant thing emotionally, and emotional experience is a huge part of whether someone uses a platform again or not.

Upfront pricing got mentioned repeatedly in feedback. Multiple participants said knowing the cost before confirming gave them confidence they don't usually feel when booking repair services. A few said they'd recommend the app specifically because of that feature. When the thing people fear most about a service — getting overcharged — is directly addressed in the design, that matters to them.

Not everything worked. The payment confirmation screen confused people in both testing rounds because it looked visually different from the rest of the app. That inconsistency made users momentarily unsure they were still in the right place. Booking history was hard to find — people expected it on the main screen but it was inside a profile submenu. Both of these are real issues that need to be fixed, not minor details.

Technician dashboard testing was smaller scale but the responses were positive. Workers described having one place to manage everything as genuinely different from how they currently operate. Rating visibility was seen as motivating — they could see their score and connect it to how much work they were getting. That link between performance and opportunity is something most of them don't have access to right now.

**Conclusion**

Home repair services in India work the way they do not because people prefer it that way, but because nobody has built a better alternative that actually fits how people behave and what they need. EasyCare is an attempt to do that — starting from real user frustrations rather than from a feature wishlist.

The design decisions in EasyCare each have a specific reason. The step-by-step booking flow exists because people under stress make worse decisions when overwhelmed. The trust signals are on the confirmation screen because that's where the decision happens, not in a profile section you'd have to navigate to. Pricing is shown upfront because post-service billing surprises are the single biggest driver of distrust in this industry. These aren't aesthetic choices — they're responses to documented problems.

Trust ended up being more central than I expected going in. Almost every design decision connects back to it in some way. How technicians are displayed, where pricing appears, when notifications go out, how invoicing works — all of it either builds or undermines the user's confidence in the transaction. Getting that right is more important than any individual feature.

Transparent pricing is probably the biggest single improvement EasyCare makes over existing alternatives. The informal pricing system in home repair is where trust goes to die. When you know what you're paying before you confirm, and when any changes have to be documented in the app, the dynamic shifts completely. Disputes become rare. Customers feel respected. Technicians get paid fairly without negotiating every time.

The technician experience matters just as much as the customer experience, even if it gets less attention in discussions like this. Giving skilled workers organized tools, performance visibility, and a professional track record they can build over time — that's not just good for them individually, it raises the overall quality of service the platform delivers. You can't have a reliable consumer experience without a reliable supply side.

There are things to fix. Payment screen consistency, booking history accessibility, mobile layout refinement for smaller screens. These are real gaps, not cosmetic things. But the core structure works — the research validates the approach, and the testing shows it achieving what it set out to achieve. The next step is building this out properly.

**Future Scope**

There's a lot of room to build on what EasyCare does. Some of these feel like near-term improvements; others are more ambitious but worth planning toward.

Predictive maintenance is the one I'm most interested in developing first. The app already tracks service history. Using that data to send reminders — 'your AC hasn't been serviced in eight months, summer is coming' — shifts the platform from being something you open when things go wrong to

something that helps you prevent things from going wrong. That's a meaningfully different value proposition.

AR-based preliminary assessment is something I want to explore. The idea is that a user could point their camera at a problem area — a leak, a damaged switchboard, a cracked tile — and get a rough indication of what kind of service they need and a ballpark cost range, before committing to a booking. Reduces unnecessary appointments and gives users more information going in.

Geographic expansion is the logical growth path. New cities mainly require onboarding local technicians and calibrating pricing. The infrastructure handles it. Multilingual support is something I'd want to add fairly early — designing English-first already limits reach in most of India, and that's a problem worth solving sooner rather than later.

Smart home integration becomes more relevant as more Indian homes get connected devices. If a water sensor detects moisture or an AC unit logs a fault, EasyCare could surface a relevant service suggestion automatically. That would make the platform genuinely proactive rather than something people only remember when something breaks.

Annual maintenance subscriptions make sense from a user perspective and a business perspective. Predictable costs for homeowners, predictable income for technicians, predictable revenue for the platform. It's the kind of feature that converts occasional users into long-term ones.

Safety enhancements — voluntary live recording with consent from both parties, emergency SOS, more rigorous credential verification — would be particularly valuable for users who live alone. These aren't edge case features; they address concerns that come up regularly when you talk to people about using home service platforms.

Eventually, insurance integration could allow repair documentation to flow directly into insurance claims without the user needing to do anything separately. That would turn EasyCare into something that's useful not just for booking but for managing the financial side of home ownership. That's the long-term direction worth building toward.

**References**

1. D. Norman, The Design of Everyday Things, Revised and Expanded Edition. New York, NY, USA: Basic Books, 2013.
2. S. Krug, Don't Make Me Think: A Common Sense Approach to Web Usability, 3rd ed. Berkeley, CA, USA: New Riders, 2014.
3. J. Nielsen, "Usability Engineering," San Francisco, CA, USA: Morgan Kaufmann, 1993.
4. ISO 9241-210, "Ergonomics of human-system interaction — Human-centred design for interactive systems," International Organization for Standardization, 2019.
5. A. Cooper, R. Reimann, and D. Cronin, About Face: The Essentials of Interaction Design, 4th ed. Indianapolis, IN, USA: Wiley, 2014.
6. B. Shneiderman et al., Designing the User Interface: Strategies for Effective Human-Computer Interaction, 6th ed. Boston, MA, USA: Pearson, 2016.
7. J. Lazar, J. Feng, and H. Hochheiser, Research Methods in Human-Computer Interaction, 2nd ed. Cambridge, MA, USA: Morgan Kaufmann, 2017.
8. Urban Company, "Service Platform Overview and Operations Model," Industry Case Study Report, 2023.
9. Nielsen Norman Group, "Trust and Credibility in Digital Interfaces," UX Research Report, 2022.
10. McKinsey & Company, "The Future of On-Demand Service Platforms," Industry Insights Report, 2023.

11. Usha Kosarkar, Gopal Sakarkar, Shilpa Gedam , "Revealing and Classification of Deepfakes Videos Images using a Customize Convolution Neural Network Model", International Conference on Machine Learning and Data Engineering (ICMLDE), 7th & 8th September 2022, 2636-2652, Volume 218, PP. 2636-2652, https://doi.org/10.1016/j.procs.2023.01.237