UI/UX CASE STUDY · 2024 – PRESENT

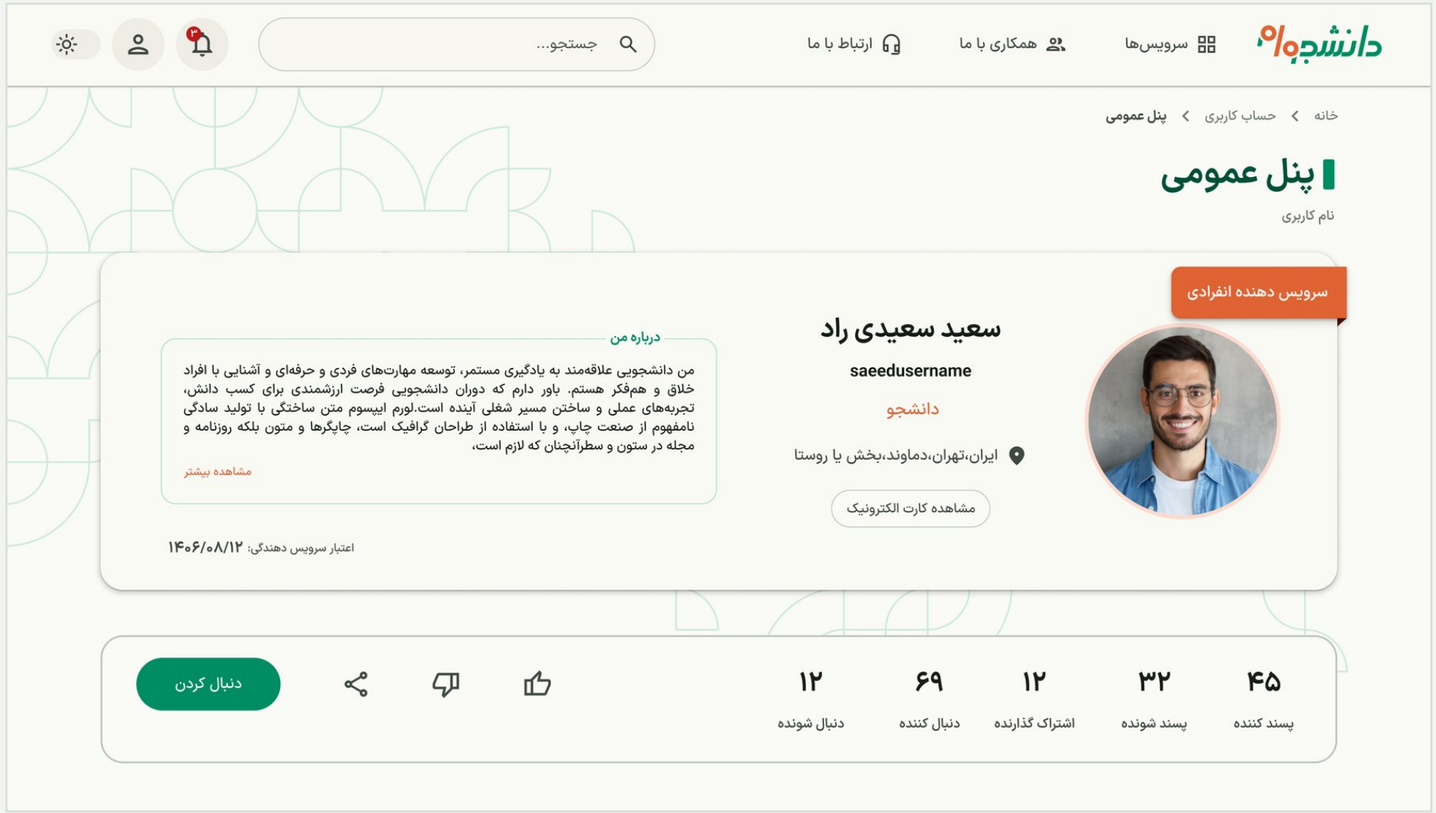
Daneshjooam

Turning dense requirements into calm, findable flows

A multi-service platform for students: discounts, rewards, support tickets, and personal private and public panels.

Role UI/UX Designer, Rahavard Hamrah Ofogh
Scope User panels, service pages, admin flows inside the user site
Platforms Desktop, tablet, mobile
Tools Figma, Material Design 3

Shared privately, on request. Screens use sample data.



The public profile: identity card, stats and actions, contact methods.

AT A GLANCE

Too much to fit, too little room to think

01

The problem

Dense requirements, lots of data, and two roles (user and admin) sharing the same pages.

02

My role

UI/UX designer for the user-facing side, across desktop, tablet and mobile.

03

What I did

Turned long pages into tabbed panels, replaced after-the-click errors with status before the click, and designed one profile layout that adapts to any amount of content.

04

The context

Many small services, each with its own evolving specification, and very detail-oriented stakeholders.

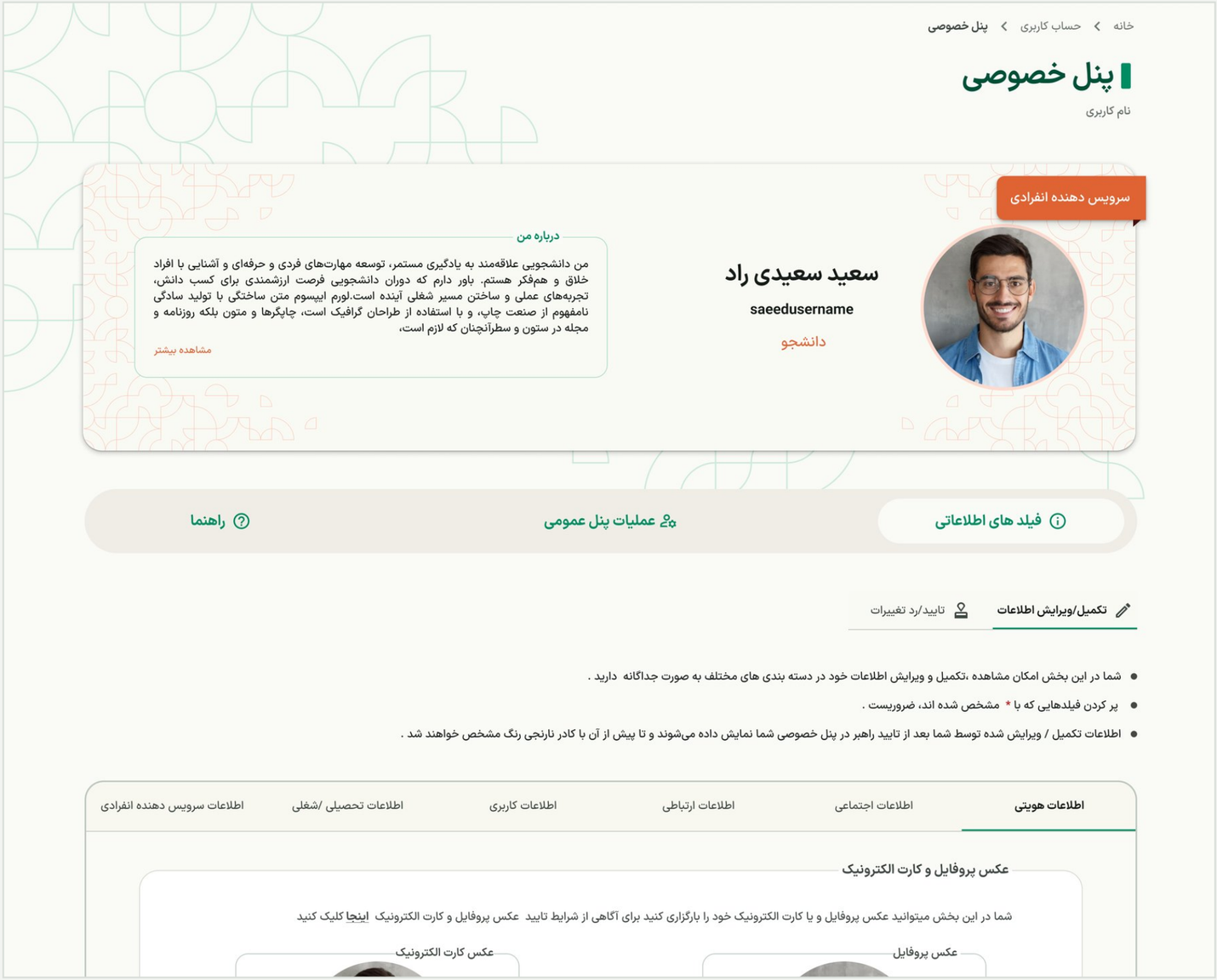
The rest of this document follows six parts. Each starts with the problem, then the decision, then the result. The deep dive (part 3) is the best single example of how I work.

PART 1

Information architecture under heavy requirements

The private panel is where users manage everything: profile fields, public-panel settings, requests, restrictions and guides. Every new requirement added another section.

- The first versions stacked everything on one long page. As the data grew, it became hard to scan and harder to act on.
- Over several iterations with my managers, we moved to **tabs grouped around what people come to do**: profile fields, public-panel operations, guide.
- Sub-tabs only where they reduce effort, never more than one level of hiding.



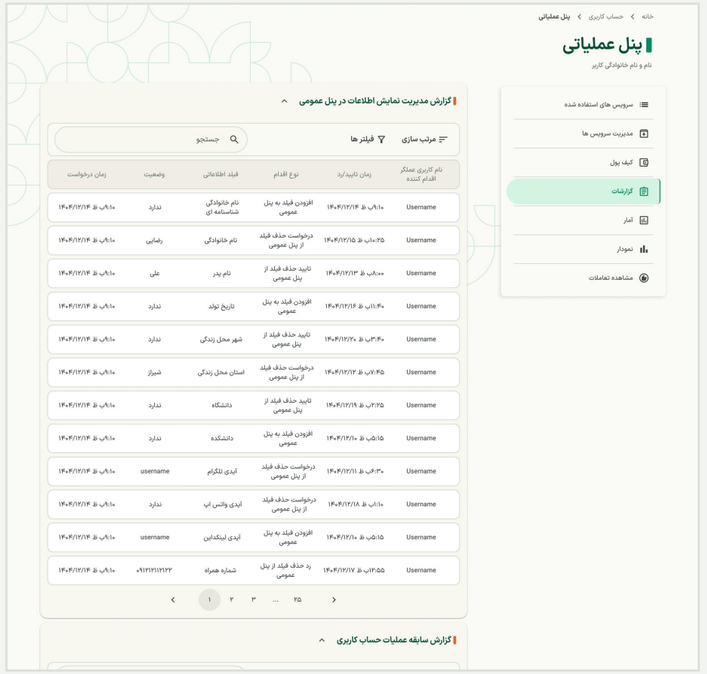
Private panel: main tabs, sub-tabs, and the fields of one category.

PART 1, CONTINUED

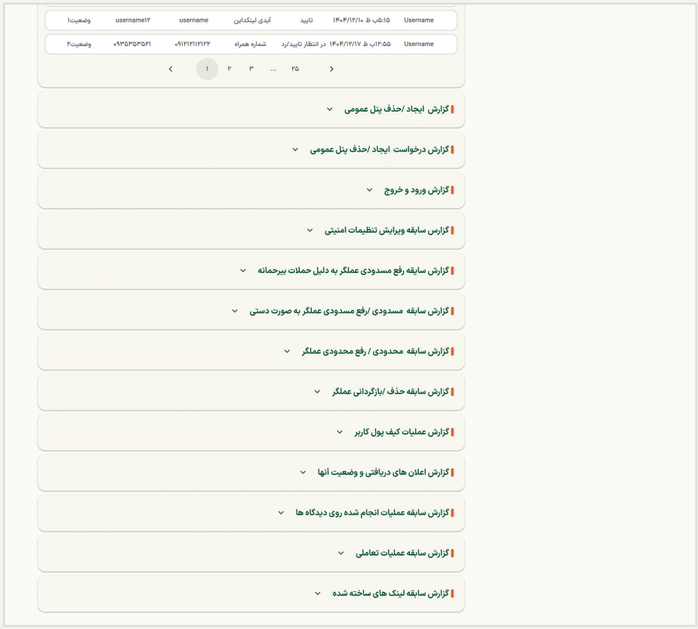
One rule for navigation, collapsible reports

Service operation panels hold a sidebar and a large number of detailed reports and statistics.

- **Separate pages get a sidebar item; sections inside a page become tabs.** One clear rule kept navigation predictable as the product grew.
- Each report sits in a **collapsible section**. People open what they need and keep the rest closed, so the page never turns into endless scrolling.



Sidebar, with one report open.



The remaining reports, collapsed.

PART 2

From difficult scenarios to reasonable paths

Some required scenarios didn't make sense as written. My job was to keep the business rule but change the experience.

Example: creating or deleting a public panel

The spec let users press a button and only then showed an error ("you have a pending request", "your membership doesn't allow this"). I proposed moving that knowledge **before the click**:

- the user's current status is always visible;
- if an action isn't possible, the button is disabled or replaced by a short explanation and the next step;
- confirmation dialogs only for real decisions, such as an irreversible delete.

I also caught a "Panel deleted" message that appeared *before* the user confirmed, and proposed splitting it into a question first and a success message after.

This redesign is currently under review with the team.

PART 2, ALSO

Designing against a moving target

The platform is split into many small services, and each has its own evolving specification.

- I kept every design in sync with the latest version of each spec and treated the docs as the source of truth.
- When specs contradicted each other, or contradicted good UX, I raised it with a clear alternative instead of silently choosing.

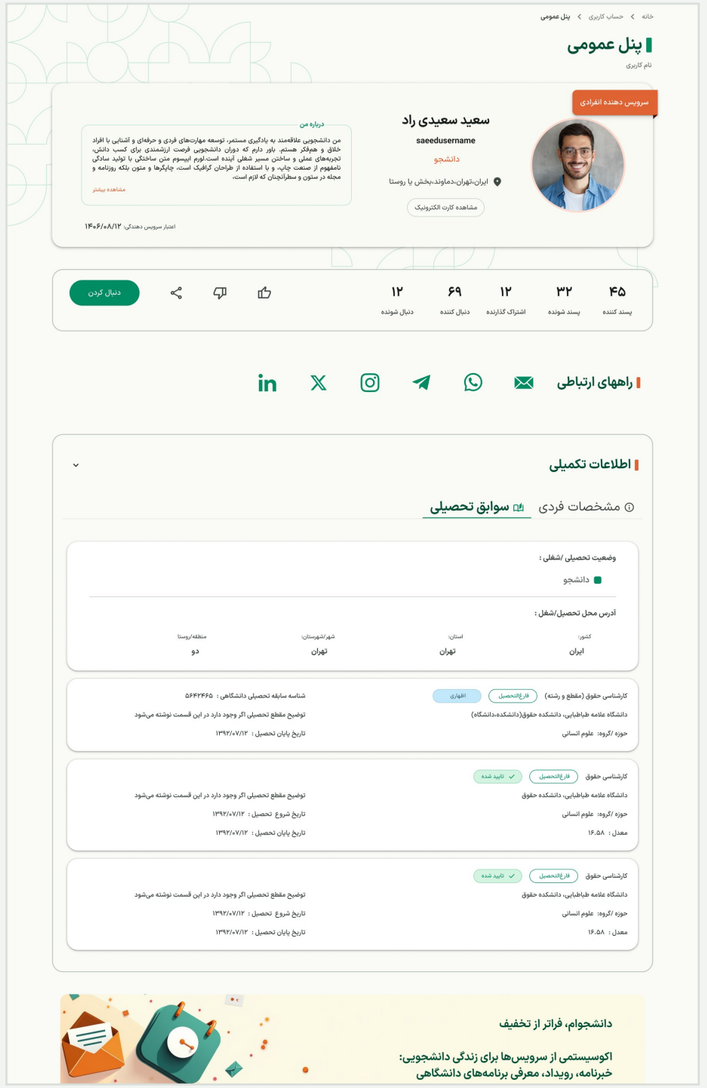
PART 3 · DEEP DIVE

Designing for variable content density

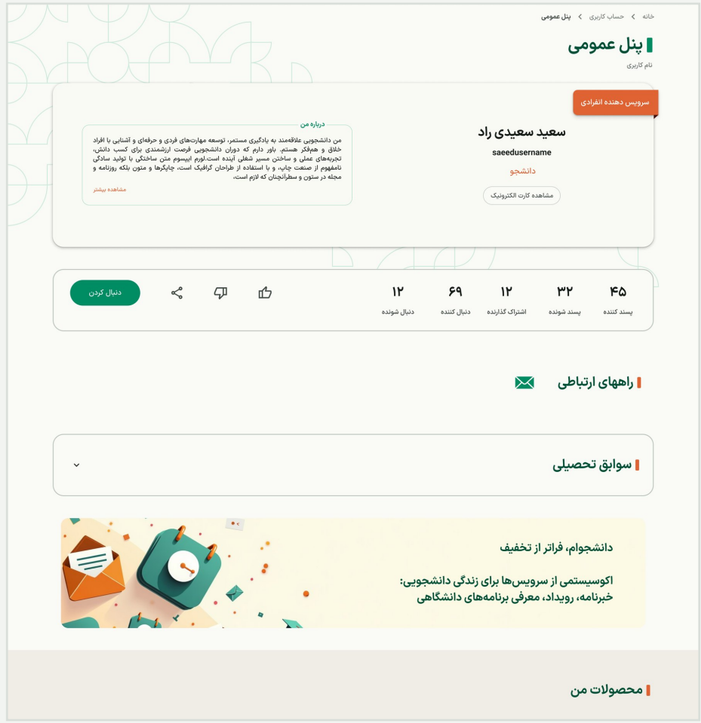
The public profile combines personal details, contact information, education and service-related content. Much of it is optional, and some sections exist only for service providers. One structure had to hold anything from an almost empty profile to a very rich one.

Constraints

- Required information came from product and reporting requirements; removing it was out of scope.
- Optional information stays under the user's control.
- Existing visual language and components had to be preserved.
- One design for minimal and rich profiles, no separate layouts.



Rich profile: additional information open, with tabs.



Minimal profile: one contact, one section.

PART 3 · PROBLEM 1

A lone icon has no context

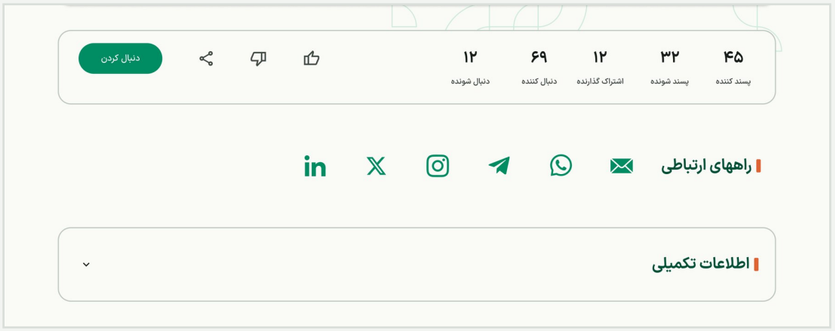
Email is shown by default; other channels appear only if the user adds them. The compact icon row worked well with many channels, but with only email, a single icon looked isolated.

Decision

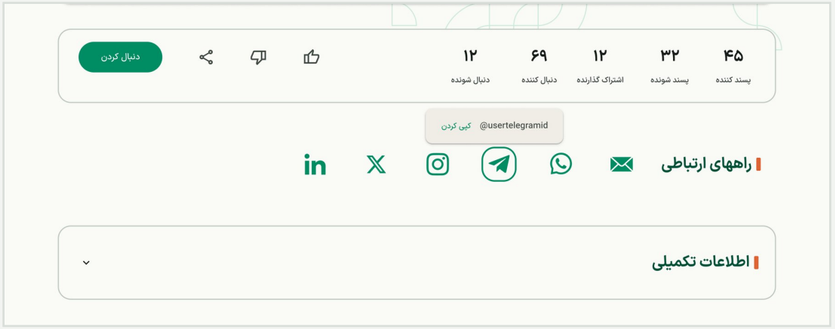
- Keep the compact icon pattern, and add an inline heading, **"Contact methods"**, on the same line. With one channel it gives context; with many, the layout stays compact.
- Click or tap opens a popover with the detail and a copy action. Hover shows only a tooltip with the channel name: interactive content never lives on hover.
- The selected icon uses the brand colour, consistent with the rest of the system.



One channel: heading + icon read as one unit.



Many channels: the same pattern, still compact.



Click opens the popover with a copy action.

PART 3 · PROBLEM 2

A tab bar with one tab

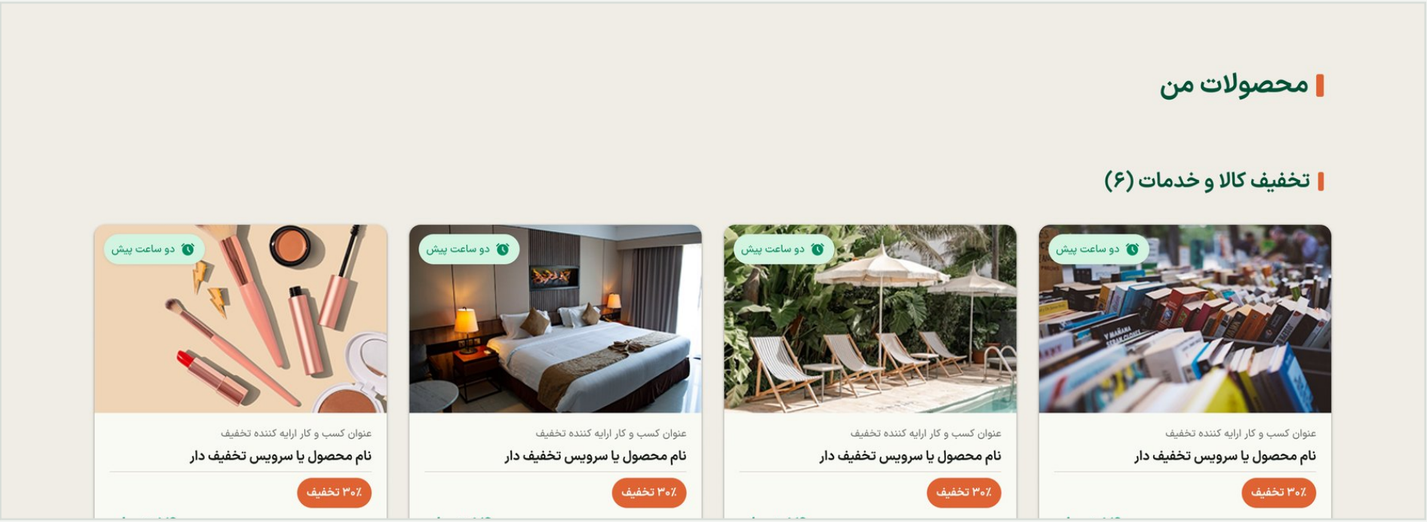
Service providers can have **My Products** and an optional **Other Information** section. With only one, a single tab is navigation with nothing to navigate.

One conditional rule

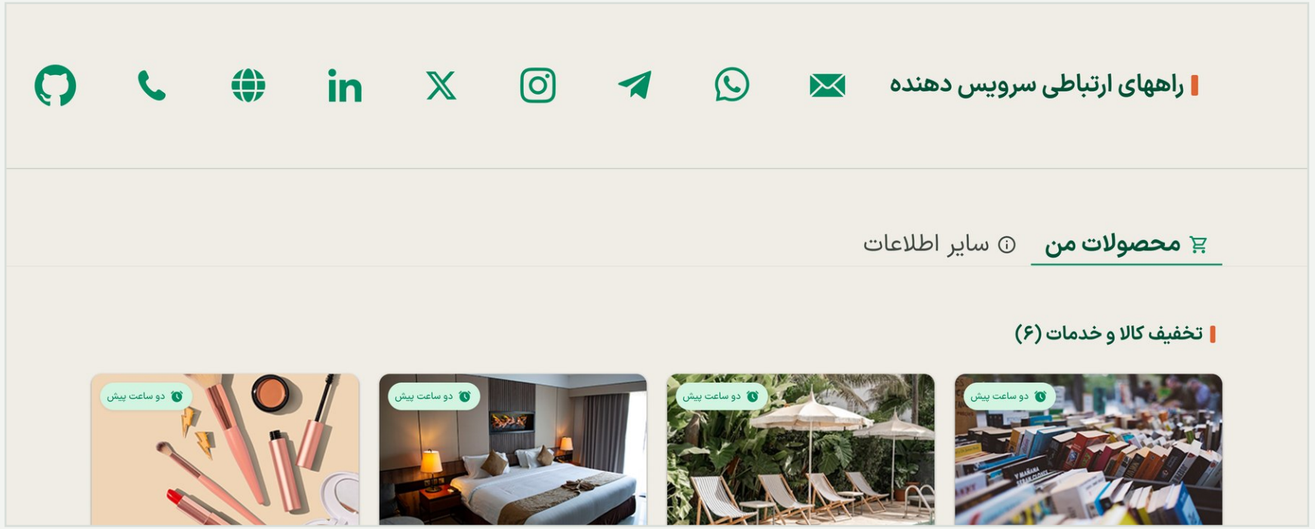
- **one section**: a standard heading, in the platform's existing style;
- **two sections**: the existing tab component;
- **no content**: the section is not shown.

Tabs start where the heading starts (the right edge, in this RTL interface), so content begins in the same place in every state.

The same rule runs inside **Additional information**, where I grouped secondary details (education, birthplace, military service status), closed by default.



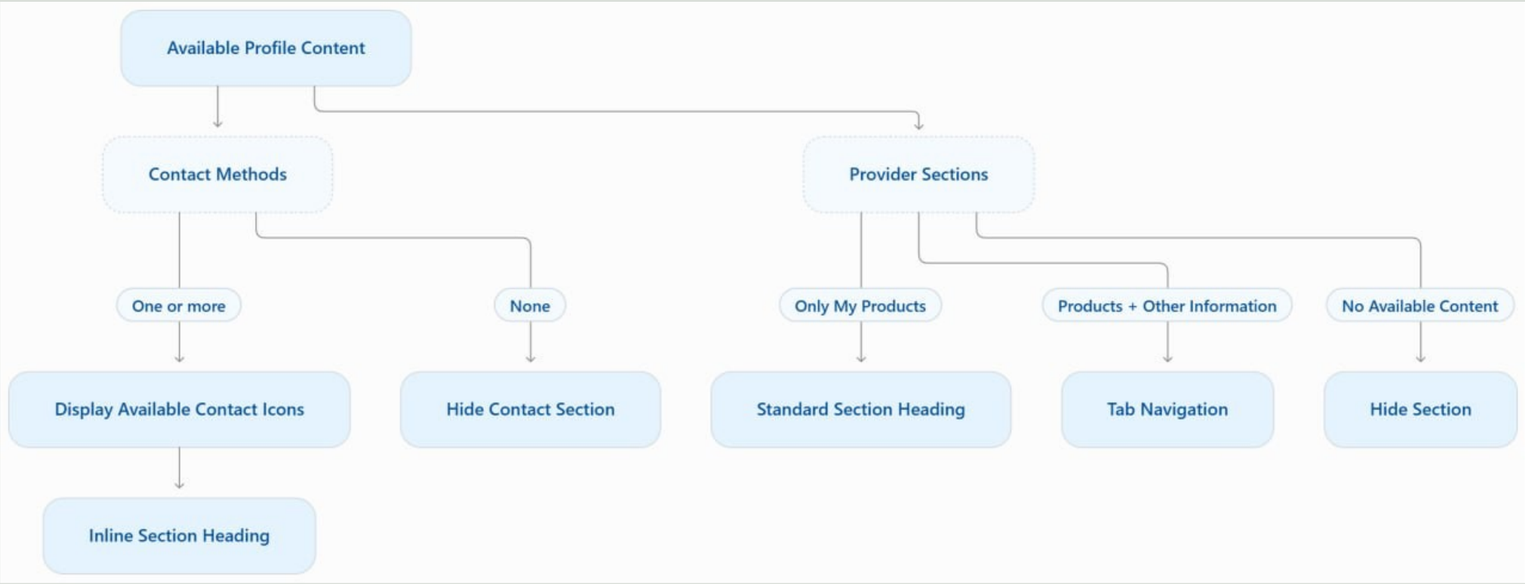
One section: a heading.



Two sections: tabs, starting at the same edge.

PART 3 · OUTCOME

Rules, not separate screens



The display rules for every combination of available content.

Consistency

Reuse existing patterns and behaviours across every profile section.

Adaptability

The interface responds to the content that exists, with no empty elements or controls.

Visual efficiency

Compact icons for contacts, but never without context.

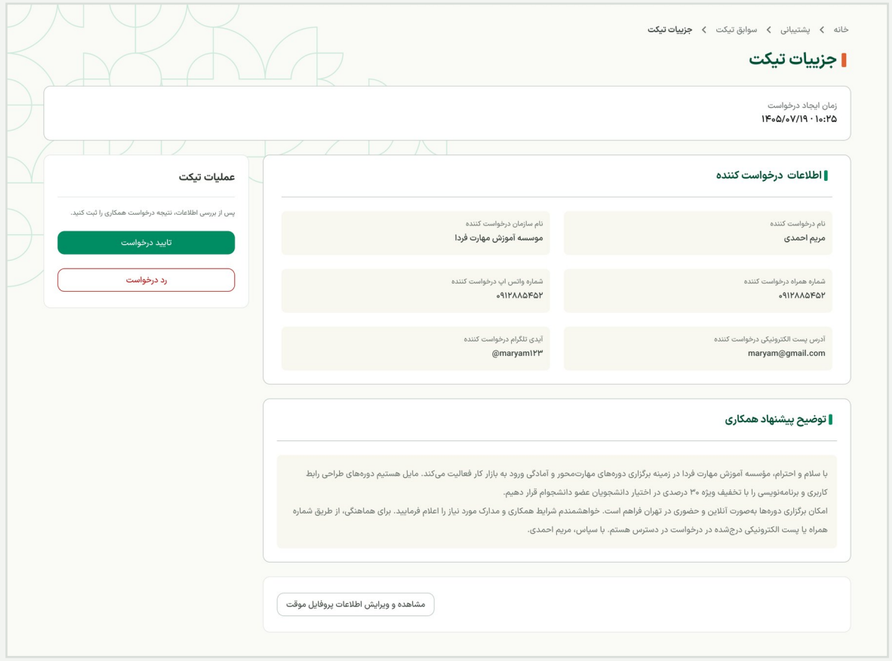
After more than ten iterations, the latest version is under review with the team.

PART 4

Two roles in one interface

The admin panel is a separate system with its own designer. But the team decided some admin actions work better **inside the user-facing site**, right where the user's data lives. Designing those fell to me, so the same pages serve the user and an admin with different permissions.

- **Ticket details** with a reply thread, internal notes and actions.
- A **searchable referral dialog** showing every department, so admins recognise instead of remember.



Ticket details, with the actions panel.



Referral dialog with search.

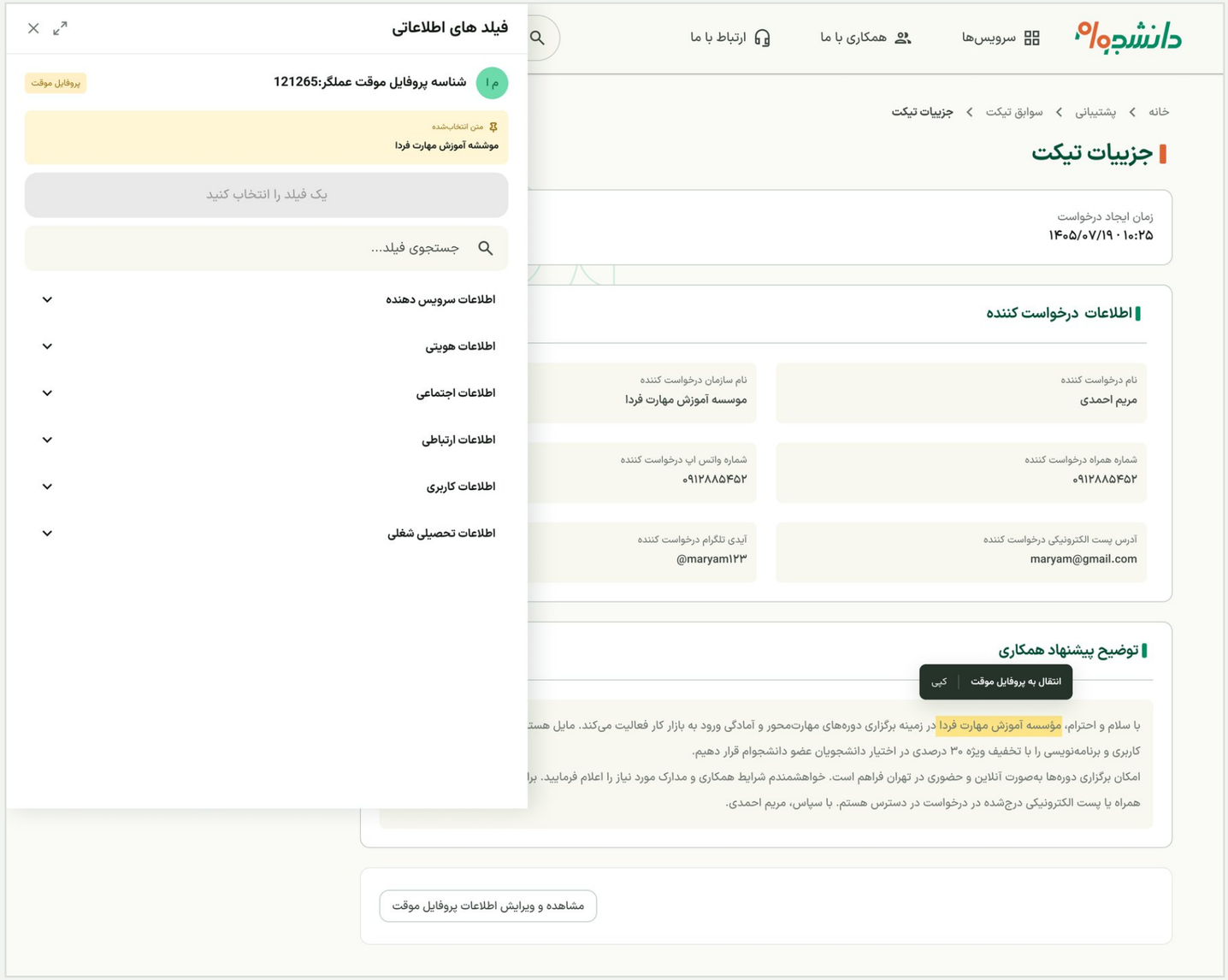
PART 4, CONTINUED

From a ticket to a profile, in one move

Admins often need to copy information from a ticket into a user's temporary profile. I designed two paths:

- **Quick:** select text in the ticket and send it to a field from a floating toolbar.
- **Full:** a side panel with every profile field, a search, and a single paste action.

Both keep the ticket visible, so the admin never loses context.



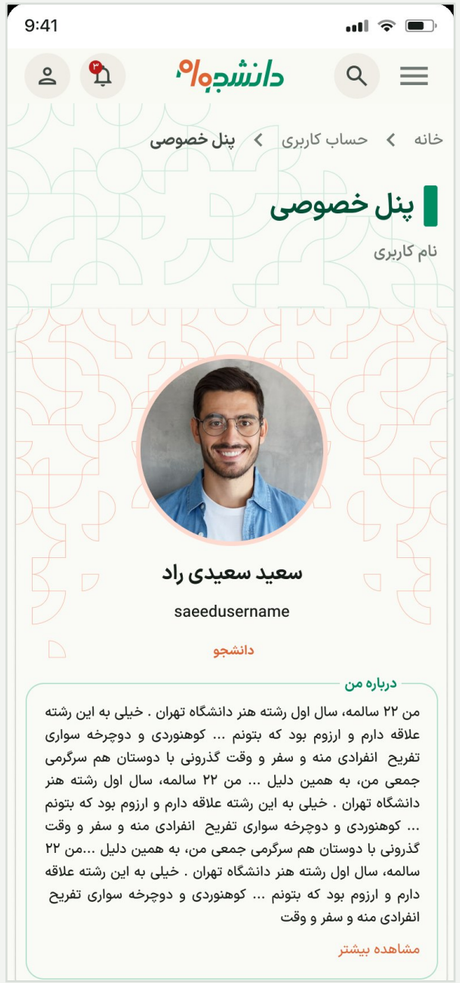
The side panel open next to the ticket.

PART 5

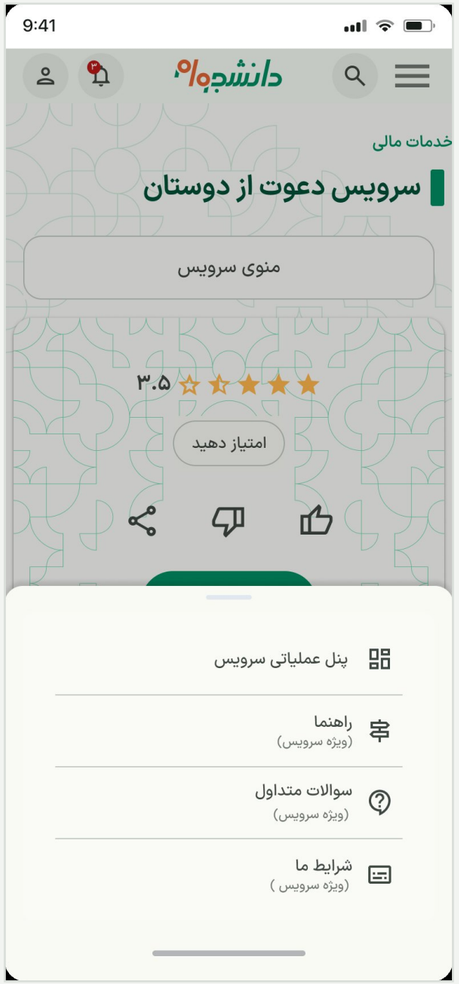
The same depth on a phone

Everything had to work on tablet and mobile without dropping features. The logic stays the same; only the pattern changes.

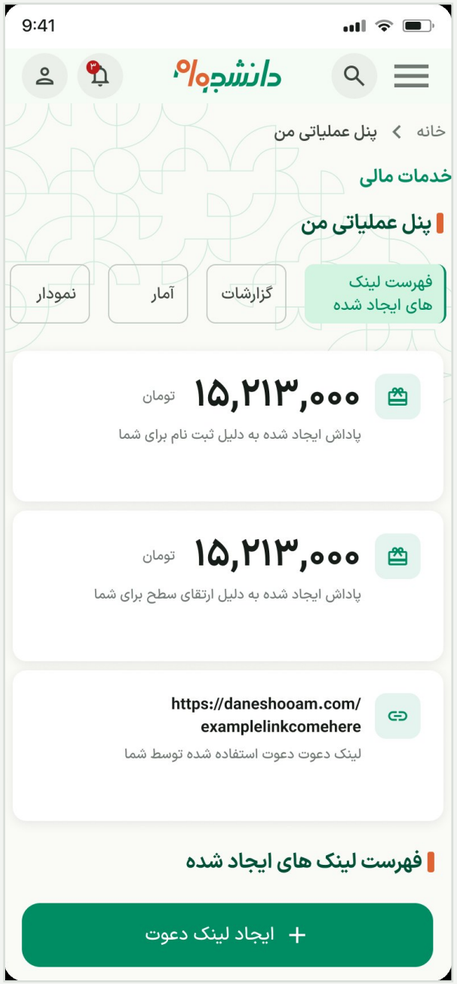
- **Navigation:** the global menu is always one hamburger. A service's own menu becomes a button that opens a **bottom sheet**, so there is never a second hamburger.
- **Panel sections:** the desktop sidebar becomes a scrollable row of chips.
- **Dense panels:** reports stay collapsible so the page doesn't become endless.



Private panel.



Service menu as a bottom sheet.



Operation panel: sections as chips.

PART 6

A colour library for light and dark

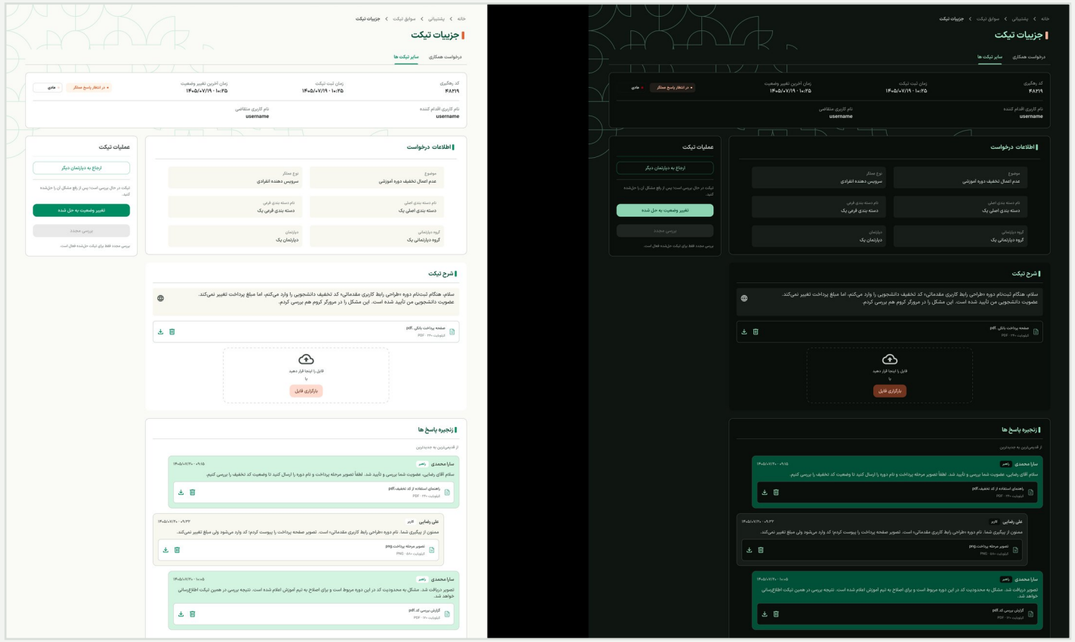
I built dark mode on top of **Material 3's colour system**, where every role has a light and a dark value as Figma variables.

- I replaced the defaults with Daneshjooam's brand palette and checked each change carried through to its dark equivalent.
- Where Material's roles weren't enough, I added custom colours, each defined for both themes.

Starting from a proven system let me spend effort where it mattered: making the brand work in both.



Tonal palettes and colour roles, light and dark.



Ticket details in light and dark mode.

WHAT I LEARNED

Detailed requirements are not the enemy of simplicity. The work is in deciding when to show each rule: before the click, at the moment of decision, or not at all.

In a product this large, a few clear rules (sidebar vs. tabs, heading vs. tabs, inline status vs. error) do more for consistency than any single screen.

CONTACT

Mahboub Ghafouri, UI/UX Designer

mahii.ghafouri@gmail.com · linkedin.com/in/mahboub-ghafouri-design · mahboub8061.github.io