معرفی و توضیح کامل سامانه – «LoopRent» پلتفرم هوشمند اجاره کالا

در سال‌های اخیر، با افزایش قیمت کالاها، رشد فرهنگ استفاده اشتراکی از منابع (Sharing Economy) و نیاز افراد به دسترسی کوتاه‌مدت به تجهیزات مختلف، ایجاد بستری دیجیتال برای اجاره ایمن، سریع و قابل اعتماد بیش از هر زمان دیگری ضروری شده است. سامانه LoopRent پاسخی هوشمندانه و مدرن به همین نیاز است؛ پلتفرمی که کاربران در آن می‌توانند هر نوع کالایی را برای مدت محدود اجاره کنند یا کالاهای بلااستفاده خود را برای کسب درآمد، در اختیار دیگران قرار دهند.

LoopRentیک سیستم تحت‌وب است که فرآیند اجاره را از حالت سنتی، پراکنده و پرریسک، به یک فرآیند کاملاً دیجیتال، قابل مدیریت و استاندارد تبدیل می‌کند. این سامانه به کاربران امکان می‌دهد بدون نیاز به تماس تلفنی، مراجعه حضوری یا واسطه، به‌صورت مستقیم با یکدیگر تعامل داشته باشند و کالاهای مختلف را در بازه زمانی دلخواه اجاره کنند.

در دنیایی که قیمت تجهیزات تخصصی (مثل دوربین، ابزارهای صنعتی، لوازم کمپینگ، تجهیزات برگزاری مراسم و...) روزبه‌روز بیشتر می‌شود، LoopRentبا ایجاد اکوسیستم اشتراک‌گذاری کالا به کاربران کمک می‌کند هزینه‌های خود را کاهش دهند، از خریدهای غیرضروری جلوگیری کنند و در مقابل، صاحبان کالا نیز یک منبع درآمد پایدار و کم‌دردسر داشته باشند.

این پلتفرم با طراحی ساده و کاربرپسند، یک ساختار منظم شامل دسته‌بندی کالاها، امکان جستجو، مشاهده جزئیات، ثبت‌نام و ورود کاربر، مدیریت پروفایل، افزودن کالا، مشاهده وضعیت اجاره‌ها و تعاملات کاربر را فراهم می‌کند. کاربران پس از ورود به سیستم می‌توانند کالاهای خود را ثبت کرده، تصاویر و توضیحات آن را بارگذاری کنند و برای آن قیمت روزانه تعیین کنند. سایر کاربران نیز با ورود به صفحه دسته‌بندی‌ها، جستجو یا مرور لیست‌ها، کالای مناسب را پیدا کرده و برای مدت مشخص اجاره کنند.

هدف اصلی LoopRent ایجاد دسترسی سریع، امن و مقرون‌به‌صرفه به کالاهاست. این سامانه علاوه بر کمک به اقتصاد خانواده‌ها، باعث افزایش بهره‌وری و گردش منابع در جامعه می‌شود. در نتیجه بسیاری از کالاهایی که زمان زیادی بدون استفاده می‌مانند، دوباره در چرخه مصرف قرار می‌گیرند و به همین دلیل نام این پلتفرم "LoopRent" به مفهوم «چرخه اجاره» انتخاب شده است.

از منظر فنی نیز پروژه با بهره‌گیری از ابزارهای مدرن توسعه وب طراحی شده و قابلیت توسعه‌پذیری، مدیریت‌پذیری و گسترش آینده را دارد. ساختار بخش‌های مختلف سامانه شامل ماژول‌های مدیریت کاربران، دسته‌بندی‌ها، کالاها، تعاملات اجاره، داشبورد ادمین و سیستم نمایش محتوا، به شکلی طراحی شده که بتوان در آینده قابلیت‌هایی مانند سیستم پرداخت آنلاین، احراز هویت دیجیتال، پشتیبانی آنلاین، پنل میزبان و... را به آن افزود.

در مجموع، LoopRentنه‌تنها یک سایت ساده برای اجاره کالا نیست، بلکه یک پلتفرم اقتصاد اشتراکی است که تلاش می‌کند تعاملات انسانی را آسان‌تر، امن‌تر و به‌صرفه‌تر کند. این سیستم با اتصال صاحبان کالا به متقاضیان اجاره، یک زنجیره ارزش اقتصادی جدید ایجاد می‌کند و می‌تواند در آینده به یکی از زیرساخت‌های مهم بازار اجاره آنلاین در ایران تبدیل شود.

✅فهرست سناریوها(Scenarios List)

- سناریوی مشاهده صفحه اصلی (Home / Landing)
- سناریوی مشاهده منوی کاربری و دسترسی به گزینه‌ها
- سناریوی مشاهده و انتخاب دسته‌بندی‌ها(Categories)
- سناریوی ورود (Login)
- سناریوی عضویت (Signup)
- سناریوی ثبت کالا (Add Item)
- سناریوی مشاهده لیست کالاهای من (My Items)
- سناریوی ویرایش پروفایل کاربر (Edit Profile)
- سناریوی مشاهده پروفایل کاربر(View Profile)

📘سناریوی1: مشاهده صفحه اصلی(Home Page)

Goal (🎯هدف)

کاربر وارد صفحه اصلی سایت می‌شود و خدمات، مزایا و منوی دسترسی سریع را مشاهده می‌کند.

Actor 👨💻

کاربر(User)

Precondition (📌پیش‌نیاز)

کاربر به اینترنت متصل است.

سایت looprent.irدر دسترس است.

Main Flow (روند اصلی)

1.کاربر وارد سایت می‌شود.

2.صفحه اصلی نمایش داده می‌شود.

3.سیستم سه بخش اصلی را نمایش می‌دهد:

آرامش خاطر

صرفه‌جویی در هزینه‌ها

عدم محدودیت زمانی

4.منوی بالای صفحه نمایش داده می‌شود:

کالاهای من

استعلام‌ها

اطلاعات حساب کاربری

5.کاربر می‌تواند روی هرکدام کلیک کند و وارد بخش‌های مربوطه شود.

Alternative Flow (جریان جایگزین)

اگر اینترنت کاربر کند باشد: بخش‌ها با تأخیر لود می‌شوند.

اگر سرور از دسترس خارج باشد: پیام خطا نمایش داده می‌شود.

Postcondition (پس‌شرط)

کاربر وارد یکی از بخش‌ها می‌شود یا در صفحه اصلی باقی می‌ماند.

📘سناریوی2: دسترسی به منوی کاربری(Header Menu)

🎯هدف

کاربر از طریق منوی بالا، یکی از بخش‌های «کالاهای من»، «استعلام‌ها» یا «اطلاعات حساب کاربری» را انتخاب می‌کند.

Actor

کاربر

Precondition

کاربر وارد صفحه سایت شده باشد.

منوی کاربری لود شده باشد.

Main Flow

1.کاربر روی آیکون مورد نظر (مثلاً آیکون آدمک = کالاهای من) کلیک می‌کند.

2.سیستم کاربر را به صفحه مربوطه هدایت می‌کند.

3.صفحه مقصد نمایش داده می‌شود.

Alternative Flow

اگر کاربر لاگین نباشد، سیستم کاربر را به صفحه ورود هدایت می‌کند.

Postcondition

کاربر وارد بخش درخواستی می‌شود.

📘سناریوی3: مشاهده و انتخاب دسته‌بندی‌ها(Categories)

Goal (🎯هدف)

کاربر دسته‌بندی‌های مختلف کالاها را مشاهده کرده و انتخاب می‌کند.

Actor 👨💻

کاربر

Precondition (📌پیش‌نیاز)

کاربر وارد سایت شده باشد.

دسته‌بندی‌های مختلف کالاها در سیستم موجود باشند.

Main Flow (📄روند اصلی)

1.کاربر وارد صفحه اصلی یا صفحه دسته‌بندی‌ها می‌شود.

2.سیستم فهرست دسته‌بندی‌های مختلف کالاها را نمایش می‌دهد (مثلاً ابزار باغبانی، لوازم آشپزخانه).

3.کاربر روی یکی از دسته‌بندی‌ها کلیک می‌کند.

.4سیستم لیست کالاهای موجود در آن دسته را نمایش می‌دهد.

.5کاربر می‌تواند هر کالا را مشاهده کرده و اطلاعات بیشتری درباره آن دریافت کند.

) Alternative Flow 🔄جریان جایگزین(

اگر اینترنت کاربر کند باشد، دسته‌بندی‌ها با تأخیر بارگذاری می‌شوند.

اگر دسته‌بندی خاصی کالای موجود نداشته باشد، سیستم پیام «هیچ کالایی برای نمایش وجود ندارد» نمایش می‌دهد.

) Postcondition 📌پس‌شرط(

کاربر به صفحه دسته‌بندی مورد نظر هدایت می‌شود و می‌تواند کالاهای آن را مشاهده کند.

📘سناریوی4: ورود(Login)

) Goal 🎯هدف(

کاربر با وارد کردن اطلاعات معتبر وارد حساب کاربری خود می‌شود.

Actor 👨‍💻

کاربر

) Precondition 📌پیش‌نیاز(

کاربر اطلاعات حساب خود (ایمیل و رمز عبور) را داشته باشد.

کاربر قبلاً ثبت‌نام کرده باشد.

Main Flow (روند اصلی)

1.کاربر وارد صفحه «ورود» می‌شود.

2.کاربر ایمیل و رمز عبور خود را وارد می‌کند.

3.کاربر دکمه «ورود» را می‌زند.

4.سیستم اطلاعات وارد شده را با پایگاه داده مقایسه می‌کند.

5.اگر ایمیل و رمز عبور صحیح باشند، کاربر وارد حساب خود می‌شود و به صفحه اصلی هدایت می‌شود.

6.اگر ورود موفقیت‌آمیز باشد، پیام «خوش آمدید» نمایش داده می‌شود.

Alternative Flow (جریان جایگزین)

اگر ایمیل یا رمز عبور اشتباه باشد، سیستم پیام خطا نمایش می‌دهد.

اگر کاربر به اینترنت متصل نباشد، سیستم پیام خطای «اتصال به اینترنت» نمایش می‌دهد.

Postcondition (پس‌شرط)

کاربر با موفقیت وارد حساب خود می‌شود.

📘سناریوی5 : عضویت(Signup)

Goal (🎯هدف)

کاربر یک حساب کاربری جدید ایجاد می‌کند.

Actor 💻👦

کاربر جدید

Precondition (📌پیش‌نیاز)

کاربر ایمیل و رمز عبور داشته باشد.

کاربر دسترسی به اینترنت داشته باشد.

Main Flow (📄روند اصلی)

1.کاربر وارد صفحه «عضویت» می‌شود.

2.کاربر اطلاعات زیر را وارد می‌کند:

نام و نام خانوادگی

ایمیل

شماره تماس

رمز عبور

تأیید رمز عبور

کشور، شهر، آدرس

3.کاربر دکمه «ثبت نام» را می‌زند.

4.سیستم اطلاعات وارد شده را بررسی و اعتبارسنجی می‌کند.

5.اگر همه اطلاعات معتبر باشند، کاربر به صفحه تایید ایمیل هدایت می‌شود.

6.سیستم پیامی به ایمیل کاربر ارسال می‌کند برای فعال‌سازی حساب.

7.کاربر پس از تأیید ایمیل وارد حساب خود می‌شود.

Alternative Flow (🔄جریان جایگزین)

اگر فیلدی خالی باشد، سیستم پیام خطای مربوطه را نمایش می‌دهد.

اگر ایمیل تکراری باشد، سیستم پیام خطای «ایمیل قبلاً ثبت شده» نمایش می‌دهد.

اگر رمز عبور و تأیید آن مطابقت نداشته باشد، سیستم پیام خطای «رمز عبورها تطابق ندارند» نمایش می‌دهد.

Postcondition (📌پس‌شرط)

حساب کاربری جدید با موفقیت ساخته شده و به صفحه تایید ایمیل هدایت می‌شود.

📘سناریوی6: ثبت کالا برای اجاره(Add Item)

Goal

کاربر یک کالا را در سایت ثبت می‌کند تا دیگران بتوانند آن را اجاره کنند.

Actor

کاربر ثبت‌نام‌شده(Owner)

Precondition

کاربر وارد حساب کاربری شده است.

صفحه «کالاهای من» یا فرم ثبت کالا باز است.

Main Flow

.1کاربر وارد بخش «ثبت کالا برای اجاره» می‌شود.

.2فرم ثبت کالا شامل فیلدهای زیر نمایش داده می‌شود:

نام کالا

دسته‌بندی

توضیحات

قیمت اجاره (تومان/روز)

آدرس تصویر(URL)

.3کاربر مقادیر فرم را وارد می‌کند.

.4کاربر دکمه «ثبت کالا» را می‌زند.

.5سیستم اطلاعات را بررسی و اعتبارسنجی می‌کند.

6.سیستم کالا را در پایگاه داده ذخیره می‌کند.

7.کالا در بخش «کالاهای ثبت‌شده من» نمایش داده می‌شود.

Alternative Flow

اگر یک فیلد خالی باشد ← پیام خطا نمایش داده می‌شود.

اگر URL تصویر نامعتبر باشد ← پیام خطای معتبر نبودن لینک نمایش داده می‌شود.

اگر سرور خطا دهد ← کاربر پیام "ثبت کالا انجام نشد" دریافت می‌کند.

Postcondition

کالا با موفقیت در پروفایل کاربر ثبت شده است.

📘سناریوی7: مشاهده لیست کالاهای من(My Items)

Goal

کاربر کالاهایی را که قبلاً ثبت کرده مشاهده می‌کند.

Actor

کاربر

Precondition

کاربر باید وارد حساب کاربری باشد.

حداقل یک کالا ثبت شده باشد.

Main Flow

.1کاربر وارد بخش «کالاهای من» می‌شود.

.2سیستم لیست کالاهای کاربر را از پایگاه داده دریافت می‌کند.

.3هر کالا شامل:

تصویر

نام

دسته‌بندی

توضیح کوتاه
نمایش داده می‌شود.

.4کاربر می‌تواند:

جزئیات کالا را ببیند

کالا را حذف کند

کالا را ویرایش کند

Alternative Flow

اگر کاربر هیچ کالایی ثبت نکرده باشد ← پیام «کالایی ثبت نشده است» نمایش داده می‌شود.

Postcondition

کاربر اطلاعات کالاهای خود را مشاهده می‌کند.

📘سناریوی8: مشاهده و ویرایش پروفایل کاربر(User Profile)

Goal

کاربر اطلاعات حساب خود را مشاهده و ویرایش می‌کند.

Actor

کاربر

Precondition

کاربر لاگین کرده باشد.

صفحه پروفایل در دسترس باشد.

Main Flow

1.کاربر روی گزینه «اطلاعات حساب کاربری (پروفایل من)» کلیک می‌کند.

2.سیستم اطلاعات کاربر را نمایش می‌دهد:

نام

ایمیل

شماره تماس

شهر

کشور

آدرس

.3کاربر روی «ویرایش» کلیک می‌کند.

.4فیلدهای قابل تغییر فعال می‌شوند.

.5کاربر اطلاعات جدید را وارد می‌کند.

.6کاربر دکمه «ذخیره» را می‌زند.

.7سیستم اطلاعات جدید را ذخیره کرده و نمایش می‌دهد.

Alternative Flow

اگر یک فیلد اشتباه وارد شود ← پیام خطای اعتبارسنجی نمایش داده می‌شود.

اگر اینترنت قطع باشد ← اطلاعات ذخیره نمی‌شود.

اگر سیستم دچار خطا شود ← پیام "ذخیره پروفایل انجام نشد" نمایش داده می‌شود.

Postcondition

اطلاعات کاربر با موفقیت ویرایش می‌شود.

فصل: ماژول‌های سیستم – (System Modules) سامانهLoopRent

سامانه «LoopRent» به‌عنوان یک پلتفرم هوشمند و جامع برای اجاره آنلاین کالا، از مجموعه‌ای از ماژول‌های مستقل اما مرتبط تشکیل شده است که هر یک بخش مهمی از عملکرد کلی سیستم را بر عهده دارند. طراحی ماژولار باعث می‌شود توسعه، نگهداری، به‌روزرسانی و توسعه آینده سیستم بسیار ساده‌تر و استانداردتر انجام شود. در این بخش، تمامی ماژول‌های اصلی و فرعی پروژه همراه با زیرسیستم‌ها، مسئولیت‌ها، داده‌های ورودی/خروجی، و نحوه تعامل آن‌ها به‌صورت کامل تشریح می‌شوند.

---

۱ .ماژول مدیریت کاربران(User Management Module)

ماژول مدیریت کاربران یکی از مهم‌ترین بخش‌های سامانه است که عملیات مرتبط با ثبت‌نام، ورود، خروج، مدیریت پروفایل و سطح دسترسی کاربران را بر عهده دارد.

وظایف کلیدی

ثبت‌نام کاربران جدید(Sign Up)

ورود کاربران(Login)

خروج از سیستم(Logout)

بازیابی رمز عبور

مدیریت اطلاعات پروفایل

تغییر رمز عبور

سطح‌بندی کاربران (کاربر معمولی، صاحب کالا، ادمین)

زیر ماژول‌ها

User Authentication

User Authorization

Profile Management

Security Layer

داده‌های ذخیره‌شده

نام، ایمیل، شماره تماس

رمز عبور هش‌شده

تاریخ ثبت‌نام

وضعیت فعال/غیرفعال

نقش کاربر

---

۲ .ماژول دسته‌بندی‌ها(Category Management Module)

این ماژول ساختار درختی دسته‌بندی کالاها را مدیریت می‌کند و یکی از پایه‌ای‌ترین بخش‌های سیستم اجاره کالاست.

وظایف

ایجاد دسته‌بندی جدید

ویرایش دسته‌بندی

حذف دسته

اختصاص دسته به کالا

نمایش دسته‌ها برای کاربران

ساختار والد/فرزند(Parent/Child)

ویژگی‌ها

پشتیبانی از چند سطح دسته‌بندی

امکان مدیریت توسط ادمین

استفاده در صفحه اصلی و فهرست کالا

---

۳. ماژول مدیریت کالا (Product / Item Management Module)

این ماژول بخش مرکزی سایت است که مدیریت کالاهای قابل اجاره توسط کاربران را انجام می‌دهد.

قابلیت‌ها

افزودن کالای جدید

ثبت تصاویر کالا

تعیین قیمت روزانه

وارد کردن توضیحات و مشخصات فنی

ویرایش کالا

حذف کالا

مشاهده موجودی و وضعیت کالا

ویژگی‌های تکنیکی

پشتیبانی از چند تصویر

ذخیره‌سازی وضعیت کالا (Available / Rented)

اتصال کالا به دسته‌بندی

داده‌های ذخیره‌شده

عنوان کالا

توضیحات

قیمت

مالک کالا

تصاویر

تاریخ ثبت

---

۴ .ماژول جستجو و فیلتر کالا(Search & Filtering Module)

این ماژول امکان جستجوی سریع و یافتن کالا را فراهم می‌کند.

امکانات

جستجوی متن کامل(Full Text Search)

فیلتر براساس دسته‌بندی

فیلتر براساس قیمت

فیلتر براساس شهر یا موقعیت

مرتب‌سازی براساس محبوبیت یا جدیدترین

فناوری‌ها

Django ORM Query Filters

Search Optimization

---

۵ .ماژول رزرو و اجاره(Rental & Booking Module)

مهم‌ترین ماژول سیستم که فرآیند اجاره را از ابتدا تا انتها مدیریت می‌کند.

کارکردها

ثبت درخواست اجاره کالا

انتخاب تاریخ شروع و پایان اجاره

محاسبه هزینه بر اساس تعداد روز

نمایش وضعیت اجاره

مدیریت سفارش‌های فعال

لغو سفارش

تایید یا رد توسط صاحب کالا

چرخه اجاره

1.کاربر کالای موردنظر را انتخاب می‌کند

2.تاریخ را مشخص می‌کند

3.سیستم هزینه را محاسبه می‌کند

4.درخواست به صاحب کالا ارسال می‌شود

5.تایید/رد توسط مالک

6.ثبت نهایی اجاره

وضعیت‌های رزرو

Pending

Accepted

Rejected

Completed

---

۶ .ماژول پرداخت(Payment Module)

)در آینده قابل اضافه‌کردن(

وظایف

اتصال به درگاه پرداخت

ثبت تراکنش

بازگشت وجه

نمایش تاریخچه پرداخت

ویژگی‌ها

امنیت بالا

گزارش‌گیری مالی

لاگ خطاها

---

۷. ماژول پیام‌رسان داخلی (Chat / Messaging Module)

(اختیاری، برای آینده)

قابلیت‌ها

چت بین صاحب کالا و اجاره‌کننده

ارسال پیام درباره جزئیات

ارسال اطلاع‌رسانی‌ها

---

۸. ماژول مدیریت تصاویر (Image Handling Module)

قابلیت‌ها

آپلود تصویر کالا

فشرده‌سازی تصویر

تغییر سایز خودکار

ذخیره امن در سرور

حذف و ویرایش تصویر

---

۹. ماژول مدیریت اعلان‌ها (Notification Module)

انواع نوتیفیکیشن

تایید/رد سفارش

تغییر وضعیت کالا

پیام جدید

ثبت کالای جدید

کانال‌ها

ایمیل

پیام درون سیستم

🟦 ۱۰ .ماژول داشبورد مدیریت(Admin Dashboard Module)

ماژول مخصوص ادمین برای مشاهده آمار، مدیریت کاربران و کنترل سیستم.

قابلیت‌ها

مدیریت کامل کاربران

مشاهده گزارش فعالیت‌ها

ایجاد و حذف دسته‌بندی

حذف یا ویرایش کالاهای مشکل‌دار

مشاهده سفارش‌ها

گزارش روزانه/هفتگی/ماهانه

🟦 ۱۱ .ماژول مدیریت نقش‌ها(Role & Permission Module)

نقش‌های موجود:

User

Owner

Admin

وظایف

کنترل دسترسی‌ها

جلوگیری از دسترسی غیرمجاز

تعریف سطوح مختلف مدیریت

---

۱۲. ماژول مدیریت گزارش‌ها(Reporting Module)

)برای نسخه حرفه‌ای(

گزارش‌هایی که سیستم تولید می‌کند:

تعداد کالاهای ثبت‌شده

تعداد اجاره‌های انجام‌شده

درآمد صاحبان کالا

وضعیت دسته‌بندی‌ها

رفتار کاربران

---

۱۳. ماژول مدیریت امنیت(Security Module)

وظایف

جلوگیری از حملاتBrute Force

مدیریتToken ها

محدودیت نرخ درخواست(Rate Limiting)

محافظت ازAPI

هش کردن رمز عبور

🟦 ۱۴ .ماژول مدیریت موقعیت جغرافیایی(Location Management Module)

قابلیت‌ها

ثبت شهر کاربر

جستجوی کالا براساس موقعیت

نمایش کالاهای نزدیک

🟦 ۱۵ .ماژول) Frontend UI/UX نمایش سمت کاربر(

وظایف

نمایش صفحات سایت

ریسپانسیو بودن

نمایش لیست کالاها

فرم‌ها و ورودی‌ها

صفحات دسته‌بندی

صفحه جزئیات کالا

فصل: لیست کامل فیچرهای سیستم LoopRent

---

۱ .فیچر ثبت‌نام کاربر جدید(User Registration)

این فیچر امکان ایجاد حساب کاربری جدید در سامانه را فراهم می‌کند.

هدف

ایجاد یک پروفایل یکتا برای هر کاربر برای استفاده از خدمات سایت.

قابلیت‌ها

فرم ثبت‌نام شامل: نام، ایمیل، رمز عبور

بررسی صحت ورودی‌ها(Validation)

جلوگیری از ثبت‌نام تکراری

ارسال کاربر به داشبورد پس از تکمیل ثبت‌نام

Flow

1.کاربر وارد صفحه ثبت نام می‌شود.

2.اطلاعات خود را وارد می‌کند.

3.سیستم صحت داده‌ها را بررسی می‌کند.

4.حساب ایجاد می‌شود.

5.کاربر وارد پنل خود می‌شود.

نکات فنی

هش کردن رمز عبور

جلوگیری ازSpam

---

۲ .فیچر ورود کاربران(User Login)

هدف

ورود امن کاربران ثبت‌نام‌شده.

قابلیت‌ها

ورود با ایمیل و رمز عبور

بررسی رمز اشتباه

ذخیره نشست(Session Handling)

Flow

1.کاربر ایمیل و رمز را وارد می‌کند.

.2سیستم مقایسه می‌کند.

.3در صورت صحت ← ورود به پنل.

---

۳ .فیچر خروج کاربر(Logout)

هدف

پایان‌دادن امن نشست کاربر.

قابلیت‌ها

حذفsession/token

انتقال به صفحه اصلی

---

۴ .فیچر بازیابی رمز عبور(Password Reset)

)در صورت نیاز می‌توان حذف کرد(

---

۵ .فیچر مشاهده صفحه اصلی(Landing Page)

هدف

نمایش کلیت سیستم، دسته‌بندی‌ها و کالاهای پیشنهادی.

قابلیت‌ها

نمایش دسته‌بندی‌ها

نمایش کالاهای جدید

دکمه ورود/ثبت‌نام

تصاویر بنر

Flow

کاربر وارد سایت می‌شود ← بخش‌های اصلی را می‌بیند.

🟦 ۶ .فیچر مشاهده دسته‌بندی‌ها(Category Listing)

هدف

دسته‌بندی کالاها برای دسترسی سریع.

قابلیت‌ها

نمایش لیست دسته‌ها

نمایش آیکون/تصویر هر دسته

کلیک ← ورود به کالاهای همان دسته

Flow

.1کاربر روی دسته "ابزار" می‌زند.

.2لیست کالاهای ابزار نمایش داده می‌شود.

---

۷ .فیچر نمایش لیست کالاها(Item Listing)

هدف

نمایش کالاهای قابل اجاره.

قابلیت‌ها

نمایش نام کالا

قیمت روزانه

تصویر

فیلتر

دسته‌بندی

جستجو

Flow

کلیک روی یک دسته ← نمایش کالاها

---

۸ .فیچر نمایش جزئیات کالا(Item Details Page)

هدف

نشان دادن اطلاعات کامل درباره کالا.

اطلاعات نمایش داده‌شده

تصویر بزرگ

قیمت

توضیحات

صاحب کالا

وضعیت موجودی

دکمه "درخواست اجاره"

Flow

کاربر یک کالا را انتخاب می‌کند ← صفحه جزئیات باز می‌شود.

---

۹. فیچر ثبت کالای جدید(Add New Item)

هدف

صاحبان کالا بتوانند کالاهای خود را برای اجاره قرار دهند.

قابلیت‌ها

افزودن عنوان

قیمت روزانه

توضیحات

دسته‌بندی

آپلود تصویر

انتخاب وضعیت

Flow

کاربر وارد ← کالاهای من ← افزودن کالا.

---

۱۰ .فیچر ویرایش اطلاعات کالا(Edit Item)

هدف

صاحب کالا بتواند مشخصات کالا را بروزرسانی کند.

قابلیت‌ها

تغییر قیمت

تغییر تصویر

تغییر توضیحات

تغییر وضعیت

---

۱۱ .فیچر حذف کالا(Delete Item)

هدف

حذف یک کالا توسط صاحب آن.

Flow

کالاهای من ← حذف ← تایید حذف.

---

۱۲ .فیچر داشبورد کاربر(User Dashboard)

هدف

مرکز مدیریت فعالیت‌های کاربر.

قابلیت‌ها

نمایش سفارش‌های فعال

نمایش کالاهای ثبت‌شده

نمایش پروفایل

دسترسی به تنظیمات

---

۱۳ .فیچر مدیریت کالاهای من(My Items)

هدف

نمایش کالاهایی که کاربر ثبت کرده.

قابلیت‌ها

لیست کالا

دکمه ویرایش

دکمه حذف

نمایش تعداد اجاره

---

۱۴ .فیچر مدیریت پروفایل کاربر(User Profile)

هدف

امکان ویرایش اطلاعات شخصی.

قابلیت‌ها

تغییر نام

تغییر تصویر پروفایل

تغییر شماره تماس

انتخاب شهر

تغییر رمز

---

۱۵ .فیچر جستجوی کالا(Search)

هدف

کاربر بتواند هر کالایی را جستجو کند.

قابلیت‌ها

جستجو بر اساس عنوان

جستجو بر اساس توضیحات

جستجو بر اساس دسته

---

۱۶ .فیچر فیلتر کالاها(Filtering)

هدف

کاربر نتیجه جستجو را دقیق‌تر کند.

فیلترها

قیمت

دسته‌بندی

شهر

وضعیت موجودی

---

۱۷ .فیچر مرتب‌سازی کالا(Sorting)

حالت‌ها

ارزان‌ترین

گران‌ترین

جدیدترین

پربازدیدترین

---

۱۸ .فیچر درخواست اجاره(Rental Request)

هدف

کاربر بتواند کالایی را برای تاریخ مشخص اجاره کند.

قابلیت‌ها

انتخاب تاریخ شروع

انتخاب تاریخ پایان

محاسبه قیمت

ارسال درخواست به صاحب کالا

نمایش وضعیت

🟦 ۱۹ .فیچر مدیریت وضعیت اجاره(Rental Status Management)

وضعیت‌ها

Pending

Accepted

Rejected

Rented

Returned

🟦 ۲۰ .فیچر تایید یا رد سفارش توسط صاحب کالا

هدف

صاحب کالا درخواست‌ها را مدیریت کند.

قابلیت‌ها

تایید

رد

توضیح علت رد

🟦 ۲۱ .فیچر نمایش تاریخچه سفارش‌ها(Order History)

هدف

کاربر بتواند تمام سفارش‌های قبلی را ببیند.

---

۲۲. فیچر نمایش اعلان‌ها(Notifications)

)در نسخه ساده می‌تواند فقط پیام باشد(

نوتیفیکیشن‌ها شامل:

تایید سفارش

رد سفارش

تغییر وضعیت کالا

پیام جدید

---

۲۳. فیچر آپلود تصویر(Image Upload)

قابلیت‌ها

انتخاب تصویر

ذخیره در سرور

تغییر اندازه

حذف تصویر قبلی

---

۲۴. فیچر امنیت(Security Features)

شامل:

جلوگیری از ورود با رمز اشتباه چندباره

Hashکردن رمز عبور

Session Timeout

جلوگیری ازXSS

جلوگیری ازSQL Injection

---

۲۵ .فیچر مدیریت نقش‌ها(Roles & Permissions)

نقش‌ها:

User

Owner

Admin

---

۲۶ .فیچر گزارش‌گیری(Reports)

گزارش‌ها:

تعداد کالا

تعداد اجاره

درآمد ناشی از اجاره

🟦 ۲۷ .فیچر اتصال به درگاه پرداخت(Payment Gateway)

)در نسخه نهایی(

🟦 ۲۸ .فیچر نمایش پیام‌های خطا(Error Handling)

قابلیت‌ها

نمایش خطای فرم

نمایش خطای اتصال

صفحه 404

🟦 ۲۹ .فیچر UI/UX حرفه‌ای(Responsive Design)

شامل:

نسخه موبایل

نسخه دسکتاپ

المان‌هایTailwind

🟦 ۳۰ .فیچر مدیریت سیستم توسط ادمین(Admin Panel Features)

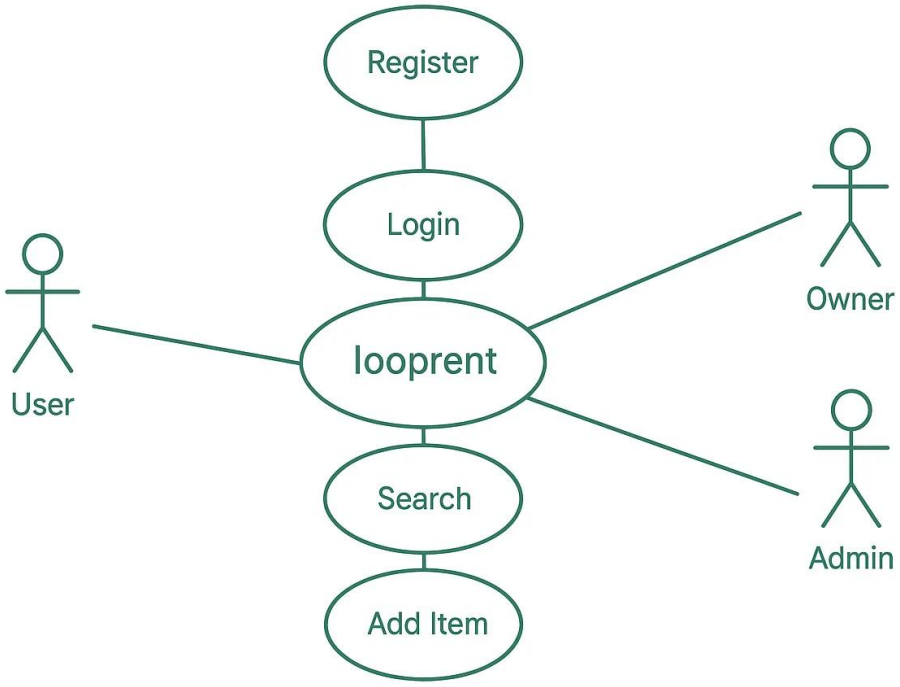
قابلیت‌ها:

مدیریت کاربران

مدیریت کالا

مدیریت دسته‌بندی‌ها

مدیریت سفارش‌ها



در ادامه به بررسی فایل های اصلی برنامه و کامپوننت ها میپردازیم :

فایل های اصلی :

App.jsx

```jsx
import React, { useEffect, useState } from "react";
import {
  BrowserRouter as Router,
  Routes,
  Route,
  useNavigate,
} from "react-router-dom";
import Header from "./components/Header";
import HomePage from "./components/HomePage";
import SignUpForm from "./components/SignUpForm";
import LoginForm from "./components/LoginForm";
import AddItemForm from "./components/AddItemForm";
import UserDashboard from "./components/UserDashboard";
import UserProfile from "./components/UserProfile";
import { supabase } from "./supabaseClient";

function HomeWithNavigation() {
  const navigate = useNavigate();
  return (
    <HomePage
      onLogin={() => navigate("/login")}
      onRegister={() => navigate("/signup")}
    />
  );
}

function App() {
  const [user, setUser] = useState(null);

  // 🧠 گرفتن کاربر فعلی از Supabase
  useEffect(() => {
    const getUser = async () => {
```

```
    const { data, error } = await supabase.auth.getUser();
    if (data?.user) {
      console.log("✅ کاربر پیدا شد:", data.user);
      setUser(data.user);
    } else {
      console.log("⚠️ هیچ کاربری پیدا نشد:", error);
    }
  };

  getUser();

  // گوش دادن به تغییر وضعیت لاگین / خروج
  const { data: subscription } = supabase.auth.onAuthStateChange(
    (_event, session) => {
      setUser(session?.user ?? null);
    }
  );

  return () => subscription.subscription.unsubscribe();
}, []);

// فقط برای تست:
console.log("🧠 User object in App:", user);

return (
  <Router>
    <div dir="rtl" className="font-sans bg-green-50 min-h-screen text-right">
      <Header />

      <Routes>
        <Route path="/" element={<HomeWithNavigation />} />
        <Route path="/signup" element={<SignUpForm />} />
        <Route path="/login" element={<LoginForm />} />
        <Route path="/dashboard" element={<UserDashboard />} />
        <Route path="/dashboard/profile" element={<UserProfile />} />

        {/* 🟢 پاس دادن user به AddItemForm */}
        <Route
          path="/dashboard/items"
          element={<AddItemForm user={user} />}
        />

        <Route
          path="/dashboard/requests"
          element={
```

```
              <p className="p-10 text-green-900 text-xl">درخواست‌های شما</p>
            }
          />

          <Route
            path="*"
            element={
              <div className="flex flex-col items-center justify-center h-[70vh]
text-green-900">
                <h2 className="text-3xl font-bold mb-3">صفحه پیدا نشد ☹</h2>
                <p className="text-lg">آدرس وارد شده اشتباه است.</p>
              </div>
            }
          />
        </Routes>
      </div>
    </Router>
  );
}

export default App;
```

🟢 خلاصه توضیح کد App.js

1) وارد کردن صفحات و کامپوننت‌ها

در ابتدای فایل، همه‌ی کامپوننت‌ها مثل Header ، LoginForm، SignupForm، Dashboard و... ایمپورت می‌شن تا داخل Router استفاده بشن.

---

2) تابع HomeWithNavigation

برای صفحه اول سایت (Home) ، از `useNavigate` استفاده شده تا وقتی کاربر روی "ورود" یا "ثبت‌نام" کلیک کرد، به مسیرهای مربوطه (`/login` و `/signup`) هدایت بشه.

---

3) مدیریت وضعیت کاربر (User State)

داخل `App` یک state به نام `user` تعریف شده:

```
const [user, setUser] = useState(null);
```

این state نگه می‌داره:

- آیا کاربر وارد شده؟
- اگر وارد شده، اطلاعاتش چیه؟

(4گرفتن کاربر فعلی ازSupabase

در :useEffect

- یک بار هنگام لود شدن صفحه، supabase.auth.getUser()اجرا می‌شود.
- اگر کاربر لاگین بوده ← مقدار user ست می‌شود.
- اگر نه user = null ← می‌ماند.

(5گوش دادن به تغییرات لاگین/خروج

این بخش اگر کاربر وارد شود یا خارج شود، stateاتوماتیک آپدیت می‌شود:

```
supabase.auth.onAuthStateChange((_event, session) => {
                                setUser(session?.user ?? null);
                                });
```

یعنی کل سیستم همیشه از وضعیت کاربر باخبر است.

(6ساختار کلیRouter

داخل <Router>همه مسیرها تعریف شده:

صفحه	مسیر
صفحه اصلی	/
فرم ثبت‌نام	/signup
فرم ورود	/login
داشبورد	/dashboard
پروفایل	/dashboard/profile
افزودن کالا	/dashboard/items

و اگر هیچ مسیری پیدا نشد ← صفحه **404**نمایش داده می‌شود.

(7پاس دادن user به AddItemForm

در این مسیر:

```
<Route
  path="/dashboard/items"
  element={<AddItemForm user={user} />}
/>
```

اطلاعات کاربر وارد شده *به فرم اضافه‌کردن کالا* فرستاده می‌شود
تا بتوانی بفهمی چه کسی کالا را ثبت کرده.

✨جمع‌بندی ساده

این فایل کارهای زیر را انجام می‌دهد:

✓مدیریت کامل مسیرهای سایت با React Router

✓چک کردن وضعیت ورود کاربر با Supabase

✓ذخیره اطلاعات کاربر در state و استفاده در صفحات

✓هدایت کاربران به صفحه‌های مختلف

✓نمایش صفحات اصلی، ورود، ثبت‌نام، داشبورد، پروفایل، افزودن کالا و صفحه 404

Main.jsx

```jsx
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
import App from './App.jsx'

createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>,
)
```

این فایل در واقع **نقطه شروع اجرای برنامه React** توئه. خیلی خلاصه و شفاف توضیح میدم:

---

1) ◈ وارد کردن StrictMode

```
import { StrictMode } from 'react'
```

`StrictMode` یک ابزار کمکی از خود React است که:

- خطاهای احتمالی
- استفاده از کدهای قدیمی
- و مشکلات lifecycle

را **در حالت توسعه (dev)** هشدار می‌دهد.

⚠ فقط در development فعاله و روی نسخه نهایی (production) هیچ اثری ندارد.

---

2) ◈ ساخت Root جدید برای React 18

```
import { createRoot } from 'react-dom/client'
```

از React 18 به بعد:

- به جای `ReactDOM.render`
- از `createRoot` استفاده می‌شود

که **عملکرد بهتر** و **concurrent rendering** را فراهم می‌کند.

---

3) ایمپورت استایل‌ها
```
import './index.css'
```

فایل CSS سراسری پروژه:

- فونت‌ها
- رنگ‌ها
- Tailwindیا استایل‌های عمومی

از اینجا وارد کل اپ می‌شوند.

---

4) ایمپورت کامپوننت اصلی
```
import App from './App.jsx'
```

Appریشه کل کامپونت‌های برنامه است
)همه صفحات و Router داخلش قرار دارند.(

---

5) Mount کردن React روی DOM
```
createRoot(document.getElementById('root')).render(
  <StrictMode>
    <App />
  </StrictMode>
)
```

این کد یعنی:

- عنصر در فایل index.html
- محل اتصال React به HTML است
- و کامپونت Appداخل آن رندر می‌شود.

خلاصه خیلی کوتاه

این فایل:

✔ React را به HTML وصل می‌کند
✔ اپلیکیشن را از Appاجرا می‌کند
✔ خطاهای توسعه را با StrictModeزودتر نشان می‌دهد
✔ نقطه شروع کل پروژه است

کامپوننت ها :

Header.jsx

```jsx
import React from "react";
import { useNavigate } from "react-router-dom";

const Header = () => {
  const navigate = useNavigate();

  return (
    <header className="bg-green-900 text-white flex justify-between items-center
px-6 py-4">
      {/* عنوان سایت (کلیک‌پذیر برای برگشت به صفحه اصلی) */}
      <h1
        className="text-xl font-bold cursor-pointer transition-colors duration-
200 hover:text-green-300"
        onClick={() => navigate("/")}
      >
        اجاره لوازم و کالا
      </h1>

      {/* منوی سه‌نقطه‌ای
      <div className="text-2xl cursor-pointer transition-colors duration-200
hover:text-green-300">*
        ⋮
      </div>*/}
    </header>
  );
```

```
};

export default Header;
```

این کامپوننت **هدر (Header)** سایت تو رو می‌سازه. خیلی خلاصه، تمیز و قابل استفاده برای داکیومنت توضیحش می‌دم:

توضیح کامپوننتHeader

1)وارد کردن وابستگی‌ها

```
import React from "react";
import { useNavigate } from "react-router-dom";
```

- `React`برای ساخت کامپوننت
- `useNavigate`برای **جابجایی بین صفحات بدون رفرش**

---

2)تعریف کامپوننتHeader

```
const Header = () => {
const navigate = useNavigate();
```

- `navigate`یک تابع است که مسیر صفحه را تغییر می‌دهد
- این کار باعث می‌شود سایت مثل SPA سریع و بدون reload باشد

---

3)ساختار ظاهری هدر

```
<header className="bg-green-900 text-white flex justify-between items-center
px-6 py-4">
```

این هدر:

- پس‌زمینه سبز تیره دارد

- متن سفید است
- آیتم‌ها را با Flex کنار هم می‌چیند
- فاصله مناسب در اطراف دارد

)همه با کلاس‌های(Tailwind

(4عنوان سایت (قابل کلیک(

```
<h1
onClick={() => navigate("/")}
>
  اجاره لوازم و کالا
</h1>
```

- عنوان سایت نمایش داده می‌شود
- وقتی روی آن کلیک شود ← کاربر به صفحه اصلی (/) برمی‌گردد
- افکت hover برای زیباتر شدن رابط کاربری اضافه شده

(5منوی سه‌نقطه‌ای (غیرفعال(

```
*/}منوی سه‌نقطه‌ای{/*
```

- فعلاً کامنت شده
- می‌توان در آینده برای:
  - منوی کاربر
  - تنظیمات
  - خروج از حساب

  استفاده شود

(6خروجی کامپوننت

```
export default Header;
```

تا در فایل‌هایی مثل App.jsx استفاده شود.

جمع‌بندی ساده

این کامپوننت:

✅ هدر ثابت سایت را نمایش می‌دهد
✅ امکان بازگشت سریع به صفحه اصلی را فراهم می‌کند
✅ از React Router برای ناوبری بدون رفرش استفاده می‌کند
✅ با Tailwind طراحی شده و ریسپانسیو است
✅ آماده توسعه برای منوی کاربری در آینده است

HeroSection.jsx

```jsx
import React from "react";
import { useNavigate } from "react-router-dom";
import bgImage from "../assets/bg-light-pattern.jpg";

const HeroSection = () => {
  const navigate = useNavigate();

  return (
    <section
      className="relative flex flex-col justify-center items-center text-center
py-24 sm:py-28 text-green-900"
      style={{
        backgroundImage: `url(${bgImage})`,
        backgroundSize: "cover",
        backgroundPosition: "center",
        backgroundRepeat: "no-repeat",
      }}
    >
      {/* لایه‌ی نیمه‌شفاف برای خواناتر شدن متن */}
      <div className="absolute inset-0 bg-white/70"></div>

      <div className="relative z-10 px-4">
        <h1 className="text-3xl sm:text-5xl font-extrabold mb-4">
          همه‌چیز برای اجاره در دسترس توست
        </h1>
        <p className="text-lg sm:text-2xl mb-8">
          هر وسیله‌ای، هر زمان، هر جا 🍃
        </p>

        <div className="flex justify-center gap-4">
          <button
```

```
            onClick={() => navigate("/signup")}
            className="bg-green-800 hover:bg-green-900 text-white px-8 py-3
rounded-full text-lg font-semibold transition"
          >
            عضویت
          </button>
          <button
            onClick={() => navigate("/login")}
            className="bg-white border border-green-800 text-green-800 hover:bg-
green-100 px-8 py-3 rounded-full text-lg font-semibold transition"
          >
            ورود
          </button>
        </div>
      </div>
    </section>
  );
};

export default HeroSection;
```

این کامپوننت در واقع **بخش Hero) بنر اصلی صفحه اول** (سایت اجاره توست. توضیحش رو ساده، مرحله‌به‌مرحله و مناسب داکیومنت برات می‌گم:

توضیح کامپوننت HeroSection

1) ایمپورت‌ها

```
import React from "react";
import { useNavigate } from "react-router-dom";
import bgImage from "../assets/bg-light-pattern.jpg";
```

- `React` برای ساخت کامپوننت
- `useNavigate` برای هدایت کاربر به صفحات دیگر بدون رفرش
- `bgImage` تصویر پس‌زمینه Hero Section است

---

2) تعریف کامپوننت

```
const HeroSection = () => {
const navigate = useNavigate();
```

- navigateمسیر کاربر را تغییر می‌دهد
- این کار باعث تجربه کاربری سریع (SPA) می‌شود

(3ساختار کلیHero Section

```
<section className="relative flex flex-col justify-center items-center text-
center ...">
```

این بخش:

- در مرکز صفحه قرار می‌گیرد
- متن‌ها وسطچین هستند
- فاصله عمودی مناسب دارد
- رنگ اصلی متن سبز است
- طراحی ریسپانسیو دارد(sm / desktop)

(4تنظیم تصویر پس‌زمینه

```
style={{
backgroundImage: `url(${bgImage})`,
backgroundSize: "cover",
backgroundPosition: "center",
}}
```

- تصویر کل بخش را پوشش می‌دهد
- بدون تکرار
- همیشه وسط تصویر نمایش داده می‌شود

(5لایه نیمه‌شفاف(Overlay)

```
<div className="absolute inset-0 bg-white/70"></div>
```

هدف:

- افزایش خوانایی متن
- جلوگیری از شلوغ شدن بک‌گراند
- ایجاد حس طراحی مدرن

(6محتوای اصلی (متن و دکمه‌ها

```
<div className="relative z-10">
```

- با z-10روی لایه سفید قرار می‌گیرد
- متن‌ها واضح دیده می‌شوند

تیتر اصلی:

```
همه‌چیز برای اجاره در دسترس توست
```

پیام اصلی برند را منتقل می‌کند.

متن توضیحی:

```
هر وسیله‌ای، هر زمان، هر جا
```

کوتاه، دوستانه و برندینگ محور.

(7دکمه‌هایCall To Action

دو دکمه مهم:

عضویت

```
onClick={() => navigate("/signup")}
```

کاربر را به صفحه ثبت‌نام هدایت می‌کند.

ورود

```
onClick={() => navigate("/login")}
```

برای کاربران قبلی جهت ورود.

هر دو دکمه:

- ریسپانسیو هستند
- افکت hover دارند
- تجربه کاربری خوبی ایجاد می‌کنند

جمع‌بندی

این کامپوننت:

- ✅ بخش اصلی معرفی سایت است
- ✅ پیام برند را واضح منتقل می‌کند
- ✅ کاربر را به اقدام (ثبت‌نام یا ورود) دعوت می‌کند
- ✅ از ناوبری React Router استفاده می‌کند
- ✅ طراحی مدرن و ریسپانسیو دارد

AddItemForm.jsx

```
import React, { useState, useEffect } from "react";
import { supabase } from "../supabaseClient";

const AddItemForm = ({ user }) => {
  const [formData, setFormData] = useState({
    title: "",
    category: "",
    description: "",
    price_per_day: "",
    image_url: "",
  });

  const [loading, setLoading] = useState(false);
  const [myItems, setMyItems] = useState([]); // ✅ آیتم‌های خود کاربر

  const handleChange = (e) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  // ✅ گرفتن کالاهای کاربر از دیتابیس
  const fetchMyItems = async () => {
    if (!user) return; // اگر هنوز لاگین نکرده
    console.log("🔍 در حال گرفتن کالاهای کاربر:", user.id);
    const { data, error } = await supabase
      .from("items")
      .select("*")
      .eq("owner_id", user.id)
      .order("created_at", { ascending: false });
```

```
  if (error) {
    console.error("❌ خطا در دریافت کالا:", error.message);
  } else {
    console.log("✅ کالاهای من:", data);
    setMyItems(data);
  }
};

useEffect(() => {
  fetchMyItems();
}, [user]); // هر وقت user تغییر کرد دوباره بخون

const handleSubmit = async (e) => {
  e.preventDefault();
  setLoading(true);
  console.log("🧠 User object:", user);

  try {
    const { error } = await supabase.from("items").insert([
      {
        title: formData.title,
        category: formData.category,
        description: formData.description,
        price_per_day: Number(formData.price_per_day),
        image_url: formData.image_url,
        available: true,
        owner_id: user?.id || null,
      },
    ]);

    if (error) throw error;

    alert("✅ کالا با موفقیت ثبت شد!");
    setFormData({
      title: "",
      category: "",
      description: "",
      price_per_day: "",
      image_url: "",
    });

    fetchMyItems(); // ✅ بعد از ثبت، لیست را بروز کن
  } catch (err) {
    console.error("❌ خطا در ثبت کالا:", err.message);
    alert("ثبت کالا ناموفق بود!");
```

```
  } finally {
    setLoading(false);
  }
};

return (
  <div className="max-w-3xl mx-auto mt-10 p-6 bg-white rounded shadow-md">
    <h2 className="text-2xl font-bold text-green-900 mb-4 text-center">
      ثبت کالا برای اجاره
    </h2>

    {/* ✅ فرم ثبت کالا */}
    <form className="flex flex-col gap-4" onSubmit={handleSubmit}>
      <input
        type="text"
        name="title"
        placeholder="نام کالا"
        className="border p-2 rounded text-right"
        value={formData.title}
        onChange={handleChange}
        required
      />

      <select
        name="category"
        className="border p-2 rounded text-right"
        value={formData.category}
        onChange={handleChange}
        required
      >
        <option value="">انتخاب دسته‌بندی</option>
        <option value="ابزار و تجهیزات">🛠 ابزار و تجهیزات</option>
        <option value="دوربین و عکاسی">📷 دوربین و عکاسی</option>
        <option value="کمپینگ و سفر">🏕 کمپینگ و سفر</option>
        <option value="لباس و لوازم مجلسی">👗 لباس و لوازم مجلسی</option>
        <option value="لوازم خانگی">🏠 لوازم خانگی</option>
        <option value="تجهیزات مجالس">🎉 تجهیزات مجالس</option>
        <option value="دیجیتال و اداری">💻 دیجیتال و اداری</option>
        <option value="وسایل نقلیه">🚗 وسایل نقلیه</option>
        <option value="سرگرمی و آموزشی">🎮 سرگرمی و آموزشی</option>
      </select>

      <textarea
        name="description"
        placeholder="توضیحات کالا"
```

```
          className="border p-2 rounded text-right"
          value={formData.description}
          onChange={handleChange}
          required
        />

        <input
          type="number"
          name="price_per_day"
          placeholder="قیمت اجاره (تومان در روز)"
          className="border p-2 rounded text-right"
          value={formData.price_per_day}
          onChange={handleChange}
          required
        />

        <input
          type="text"
          name="image_url"
          placeholder="آدرس تصویر کالا (URL)"
          className="border p-2 rounded text-right"
          value={formData.image_url}
          onChange={handleChange}
          required
        />

        <button
          type="submit"
          disabled={loading}
          className="bg-green-900 text-white py-2 rounded hover:bg-green-800
transition"
        >
          {loading ? "در حال ثبت..." : "ثبت کالا"}
        </button>
      </form>

      {/* ✅ لیست کالاهای ثبت‌شده */}
      <h3 className="text-xl font-bold text-green-900 mt-8 mb-3">
        کالاهای ثبت‌شده من
      </h3>
      {myItems.length === 0 ? (
        <p className="text-gray-600 text-center">هنوز کالایی ثبت نکرده‌اید.</p>
      ) : (
        <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4">
          {myItems.map((item) => (
```

```
            <div
              key={item.id}
              className="border rounded-lg p-3 shadow-sm hover:shadow-md
transition"
            >
              <img
                src={item.image_url}
                alt={item.title}
                className="w-full h-40 object-cover rounded"
              />
              <h4 className="font-semibold mt-2 text-green-900">
                {item.title}
              </h4>
              <p className="text-sm text-gray-700">{item.category}</p>
              <p className="text-sm text-gray-500 mt-1">
                {item.price_per_day} تومان / روز
              </p>
            </div>
          ))}
        </div>
      )}
    </div>
  );
};

export default AddItemForm;
```

این کامپوننت یکی از **مهم‌ترین بخش‌های پروژه اجاره کالا**ست. توضیحش رو طوری می‌گم که هم **کاملاً بفهمی**، هم بتونی **عیناً داخل داکیومنت دانشگاهی** استفاده کنی.

توضیح کامپوننت `AddItemForm`

1) ایمپورت‌ها

```
import React, { useState, useEffect } from "react";
import { supabase } from "../supabaseClient";
```

- `useState` برای مدیریت داده‌های فرم و لیست کالاها
- `useEffect` برای گرفتن داده‌ها هنگام تغییر کاربر

- supabaseبرای ارتباط مستقیم با دیتابیس(CRUD)

(2تعریف کامپوننت وState ها

```
const AddItemForm = ({ user }) => {
```

این کامپوننت اطلاعات **کاربر لاگین‌شده** را به‌صورت prop دریافت می‌کند.

Stateهای اصلی:

```
// formDataاطلاعات فرم ثبت کالا
// loadingوضعیت ارسال فرم
// myItemsلیست کالاهای ثبت‌شده توسط کاربر
```

(3مدیریت تغییرات فرم

```
const handleChange = (e) => {
setFormData({ ...formData, [e.target.name]: e.target.value });
};
```

هر بار که کاربر در input چیزی تایپ می‌کند:

- مقدار جدید در state ذخیره می‌شود
- فرم به‌صورت Controlled Component مدیریت می‌شود

(4گرفتن کالاهای کاربر از دیتابیس

```
const fetchMyItems = async () => {
```

این تابع:

- فقط اگر کاربر لاگین کرده باشد اجرا می‌شود
- از جدول itemsداده می‌گیرد
- فقط کالاهایی که متعلق به همان کاربر هستند(owner_id = user.id)
- جدیدترین کالاها را اول نمایش می‌دهد

```
.eq("owner_id", user.id)
.order("created_at", { ascending: false })
```

(5اجرای خودکار دریافت کالاها

```
useEffect(() => {
```

```
    fetchMyItems();
  }, [user]);
```

هر زمان:

- کاربر لاگین شد
- یا اطلاعات کاربر تغییر کرد

لیست کالاها دوباره از دیتابیس خوانده می‌شود.

(6ثبت کالا در دیتابیس

```
const handleSubmit = async (e) => {
```

در این تابع:

1. ارسال فرم متوقف می‌شود
2. حالت loading فعال می‌شود
3. یک رکورد جدید در جدول `items`درج می‌شود
4. شناسه کاربر به عنوان مالک ذخیره می‌شود

```
owner_id: user?.id || null
```

(7مدیریت خطا و موفقیت

- در صورت موفقیت ← پیام تأیید نمایش داده می‌شود
- فرم پاک می‌شود
- لیست کالاها دوباره لود می‌شود
- در صورت خطا ← پیام خطا نمایش داده می‌شود

(8بخش رابط کاربری(UI

فرم ثبت کالا :

شامل:

- نام کالا
- دسته‌بندی
- توضیحات
- قیمت اجاره روزانه
- لینک تصویر

همه فیلدها:

- Requiredهستند
- راست‌چین
- ریسپانسیو
- طراحی‌شده باTailwind

لیست کالاهای ثبت‌شده:

```
myItems.map(...)
```

- کارت‌بندی شده
- تصویر، نام، دسته و قیمت نمایش داده می‌شود
- فقط کالاهای همان کاربر نشان داده می‌شود

اگر کاربر هیچ کالایی نداشته باشد:

```
هنوز کالایی ثبت نکرده اید.
```

جمع‌بندی کاربردی

این کامپوننت:

✔ امکان ثبت کالا برای اجاره را فراهم می‌کند
✔ اطلاعات را در Supabase ذخیره می‌کند
✔ فقط کالاهای مربوط به کاربر را نمایش می‌دهد
✔ بعد از ثبت، لیست را به‌روزرسانی می‌کند
✔ از الگوی CRUD به‌صورت کامل استفاده می‌کند
✔ تجربه کاربری ساده و حرفه‌ای دارد

ثبت کالا برای اجاره

نام کالا

انتخاب دسته‌بندی

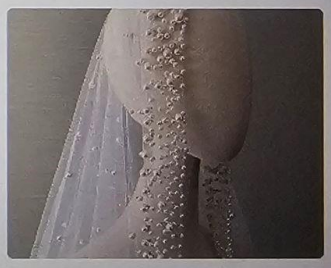
توضیحات کالا

قیمت اجاره (تومان در روز)

آدرس تصویر کالا (URL)

ثبت کالا

کالاهای ثبت‌شده من



تور عروس
لباس و لوازم مجلسی
350000 تومان / روز

دستگاه پرس آجر
ابزار و تجهیزات
2500000 تومان / روز

تور ماهیگیری
کمپینگ و سفر
50000 تومان / روز

LoginForm.jsx

```
// src/components/LoginForm.jsx
import React, { useState } from "react";
import { supabase } from "../supabaseClient"; // مسیرت را درست تنظیم کن
import { useNavigate } from "react-router-dom";

const LoginForm = () => {
  const navigate = useNavigate();
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [loading, setLoading] = useState(false);

  const handleLogin = async (e) => {
    e.preventDefault();
    console.log("🔑 دکمه ورود کلیک شد");

    setLoading(true);
    try {
      const { data, error } = await supabase.auth.signInWithPassword({
        email,
        password,
      });

      if (error) throw error;

      alert("✅ ورود موفق بود!");
      navigate("/dashboard");
    } catch (error) {
      console.error("❌ خطا در ورود:", error);
      alert("ایمیل یا رمز عبور اشتباه است یا کاربر وجود ندارد!");
    } finally {
      setLoading(false);
    }
  };

  return (
    <div className="max-w-md mx-auto mt-10 p-6 bg-white rounded shadow-md">
      <h2 className="text-2xl font-bold text-green-900 mb-4 text-
center">ورود</h2>
      <form onSubmit={handleLogin} className="flex flex-col gap-4">
        <input
```

```
          type="email"
          value={email}
          onChange={(e) => setEmail(e.target.value)}
          placeholder="ایمیل"
          className="border p-2 rounded text-right"
          required
        />
        <input
          type="password"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
          placeholder="رمز عبور"
          className="border p-2 rounded text-right"
          required
        />
        <button
          type="submit"
          disabled={loading}
          className="bg-green-900 text-white py-2 rounded hover:bg-green-800"
        >
          {loading ? "در حال ورود..." : "ورود"}
        </button>
      </form>
    </div>
  );
};

export default LoginForm;
```

این کامپوننت **فرم ورود کاربران** با استفاده از **Supabase Auth** است. توضیحش رو مرحله‌به‌مرحله و مناسب داکیومنت می‌نویسم:

توضیح کامپوننت `LoginForm`

1) ایمپورت‌ها

```
import React, { useState } from "react";
import { supabase } from "../supabaseClient";
import { useNavigate } from "react-router-dom";
```

- `useState`برای نگهداری ایمیل، رمز عبور و وضعیت لودینگ
- `supabase`برای احراز هویت کاربر
- `useNavigate`برای انتقال کاربر بعد از ورود موفق

State (2های کامپوننت

```
const [email, setEmail] = useState("");
const [password, setPassword] = useState("");
const [loading, setLoading] = useState(false);
```

- `email` ← ایمیل واردشده توسط کاربر
- `password` ← رمز عبور
- `loading` ← جلوگیری از چندبار کلیک روی دکمه ورود

(3تابع ورود کاربر

```
const handleLogin = async (e) => {
```

مراحل انجام:

1. جلوگیری از رفرش صفحه
2. فعال کردن حالت لودینگ
3. ارسال ایمیل و رمز عبور بهSupabase
4. بررسی خطا یا موفقیت

```
supabase.auth.signInWithPassword({ email, password });
```

(4مدیریت موفقیت و خطا

- اگر ورود موفق باشد:
  - پیام تأیید نمایش داده می‌شود
  - کاربر به `dashboard/`منتقل می‌شود
- اگر خطا رخ دهد:
  - پیام خطا نمایش داده می‌شود
  - اطلاعات در کنسول لاگ می‌شود (برای دیباگ)

(5رابط کاربری فرم

فرم شامل:

- فیلد ایمیل
- فیلد رمز عبور
- دکمه ورود با متن پویا

```
" ? loading}در حال ورود" : "...ورود}"
```

که تجربه کاربری را بهتر می‌کند.

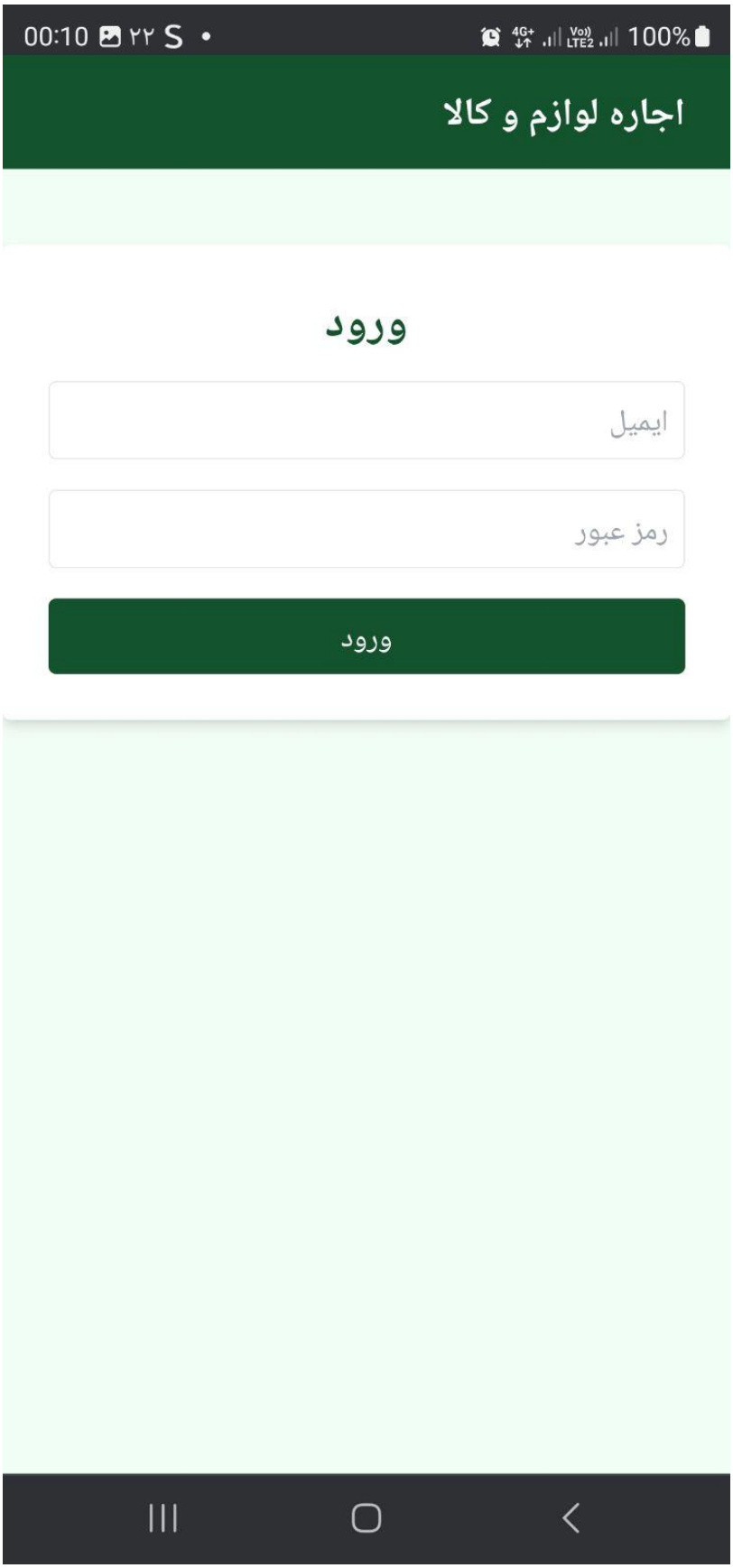
(6طراحی رابط کاربری

- طراحی شده باTailwind CSS
- راست‌چین برای زبان فارسی
- ساده، تمیز و ریسپانسیو

جمع‌بندی

این کامپوننت:

✔احراز هویت کاربر را انجام می‌دهد
✔از Supabase Auth استفاده می‌کند
✔خطاهای ورود را مدیریت می‌کند
✔بعد از ورود موفق، کاربر را به داشبورد هدایت می‌کند
✔تجربه کاربری ساده و امن فراهم می‌کند


00:10 ۲۲ S •
100%
اجاره لوازم و کالا
ورود
ایمیل
رمز عبور
ورود

SignUpForm.jsx

```jsx
// src/components/SignUpForm.jsx
import React, { useState } from "react";
import { supabase } from "../supabaseClient"; // مسیر را بر اساس پروژه‌ات تنظیم کن
import { useNavigate } from "react-router-dom";

const SignUpForm = () => {
  const navigate = useNavigate();
  const [formData, setFormData] = useState({
    name: "",
    phone: "",
    email: "",
    password: "",
    confirmPassword: "",
    country: "",
    city: "",
    address: "",
  });
  const [loading, setLoading] = useState(false);

  const handleChange = (e) => {
    setFormData((p) => ({ ...p, [e.target.name]: e.target.value }));
  };

  const handleSubmit = async (e) => {
    e.preventDefault();

    if (formData.password !== formData.confirmPassword) {
      alert("رمز عبور و تکرار آن یکسان نیست!");
      return;
    }

    setLoading(true);
    try {
      // 1️⃣ ساخت حساب کاربری در Supabase Auth
      const { data, error } = await supabase.auth.signUp({
        email: formData.email,
        password: formData.password,
      });
```

```
    if (error) throw error;
    const user = data.user;

    // 2️⃣ ذخیره اطلاعات اضافی کاربر در جدول users
    if (user) {
      const { error: insertError } = await supabase.from("users").insert([
        {
          id: user.id,
          name: formData.name,
          phone: formData.phone,
          email: formData.email,
          country: formData.country,
          city: formData.city,
          address: formData.address,
          created_at: new Date(),
        },
      ]);

      if (insertError) throw insertError;
    }

    alert("ثبت‌نام با موفقیت انجام شد ✅");
    navigate("/dashboard");
    setFormData({
      name: "",
      phone: "",
      email: "",
      password: "",
      confirmPassword: "",
      country: "",
      city: "",
      address: "",
    });
  } catch (error) {
    console.error("❌ خطا در ثبت‌نام:", error);
    alert("خطا در ثبت‌نام: " + error.message);
  } finally {
    setLoading(false);
  }
};

return (
  <div className="max-w-md mx-auto mt-10 p-6 bg-white rounded shadow-md">
```

```
      <h2 className="text-2xl font-bold text-green-900 mb-4 text-
center">عضویت</h2>
      <form onSubmit={handleSubmit} className="flex flex-col gap-4">
        <input name="name" value={formData.name} onChange={handleChange}
placeholder="نام و نام خانوادگی" className="border p-2 rounded" required />
        <input name="phone" value={formData.phone} onChange={handleChange}
placeholder="شماره تماس" className="border p-2 rounded" required />
        <input name="email" type="email" value={formData.email}
onChange={handleChange} placeholder="ایمیل" className="border p-2 rounded" required
/>
        <input name="password" type="password" value={formData.password}
onChange={handleChange} placeholder="رمز عبور" className="border p-2 rounded"
required />
        <input name="confirmPassword" type="password"
value={formData.confirmPassword} onChange={handleChange} placeholder="تأیید رمز عبور"
className="border p-2 rounded" required />
        <input name="country" value={formData.country} onChange={handleChange}
placeholder="کشور" className="border p-2 rounded" />
        <input name="city" value={formData.city} onChange={handleChange}
placeholder="شهر" className="border p-2 rounded" />
        <textarea name="address" value={formData.address} onChange={handleChange}
placeholder="آدرس کامل" className="border p-2 rounded" rows="3" />
        <button type="submit" disabled={loading} className="bg-green-900 text-
white py-2 rounded hover:bg-green-800">
          {loading ? "در حال ثبت..." : "ثبت نام"}
        </button>
      </form>
    </div>
  );
};

export default SignUpForm;
```

اینجا توضیح **کامل، مرحله‌به‌مرحله و مناسب داکیومنت دانشگاهی / پروژه MVP** برای کامپوننت `SignUpForm` آوردم:

توضیح کامپوننت SignUpForm

این کامپوننت وظیفه **ثبت‌نام کاربر جدید** در سیستم را بر عهده دارد و از **Supabase Authentication**به‌همراه **ذخیره اطلاعات تکمیلی کاربر در دیتابیس** استفاده می‌کند.

---

1 ایمپورت‌ها

```
import React, { useState } from "react";
import { supabase } from "../supabaseClient";
import { useNavigate } from "react-router-dom";
```

- useState → مدیریت داده‌های فرم
- supabase → ارتباط با سرویس احراز هویت و دیتابیس
- useNavigate → هدایت کاربر پس از ثبت‌نام موفق

---

2 مدیریت State فرم

```
const [formData, setFormData] = useState({ ... });
```

تمام اطلاعات کاربر در یک آبجکت ذخیره می‌شود:

- اطلاعات هویتی (نام، تلفن، ایمیل)
- اطلاعات مکانی (کشور، شهر، آدرس)
- اطلاعات امنیتی (رمز عبور)

این ساختار باعث **مدیریت ساده‌تر فرم‌های بزرگ** می‌شود.

---

3 تابع handleChange

```
const handleChange = (e) => {
setFormData((p) => ({ ...p, [e.target.name]: e.target.value }));
};
```

- هر ورودی بر اساس name مقدار مربوطه را آپدیت می‌کند
- از الگوی **Controlled Components** در React استفاده شده
- از تکرار کد جلوگیری می‌شود

---

4 اعتبارسنجی رمز عبور

```
if (formData.password !== formData.confirmPassword)
```

- بررسی تطابق رمز عبور و تکرار آن

- جلوگیری از ارسال اطلاعات اشتباه به سرور
- افزایش تجربه کاربری و امنیت

5⃣ ثبت‌نام در Supabase Auth

```
supabase.auth.signUp({
email: formData.email,
password: formData.password,
});
```

- ایجاد حساب کاربری در سیستم احراز هویتSupabase
- مدیریت امن رمز عبور بدون ذخیره آن در دیتابیس پروژه

6⃣ ذخیره اطلاعات تکمیلی کاربر

```
supabase.from("users").insert([...]);
```

در این مرحله:

- اطلاعات اضافی کاربر در جدول `users` ذخیره می‌شود
- `id` کاربر با `user.id` هماهنگ است) ارتباط مستقیم با(Auth
- امکان توسعه پروفایل کاربر در آینده فراهم می‌شود

📌 **تفکیک احراز هویت از اطلاعات پروفایل = معماری استاندارد**

7⃣ مدیریت موفقیت عملیات

```
alert("ثبت‌نام با موفقیت انجام شد ✅");
navigate("/dashboard");
```

- نمایش پیام موفقیت
- انتقال مستقیم کاربر به داشبورد
- ریست کردن فرم برای استفاده مجدد

8⃣ مدیریت خطا

```
catch (error) {
console.error("❌ خطا در ثبت‌نام:", error);
}
```

- ثبت خطا در کنسول برای دیباگ
- نمایش پیام مناسب به کاربر

9️⃣ رابط کاربری فرم

فرم شامل:

- فیلدهای متنی
- فیلد رمز عبور و تأیید آن
- دکمه با وضعیت لودینگ

```
" ? loading}در حال ثبت" : "...ثبت نام{"
```

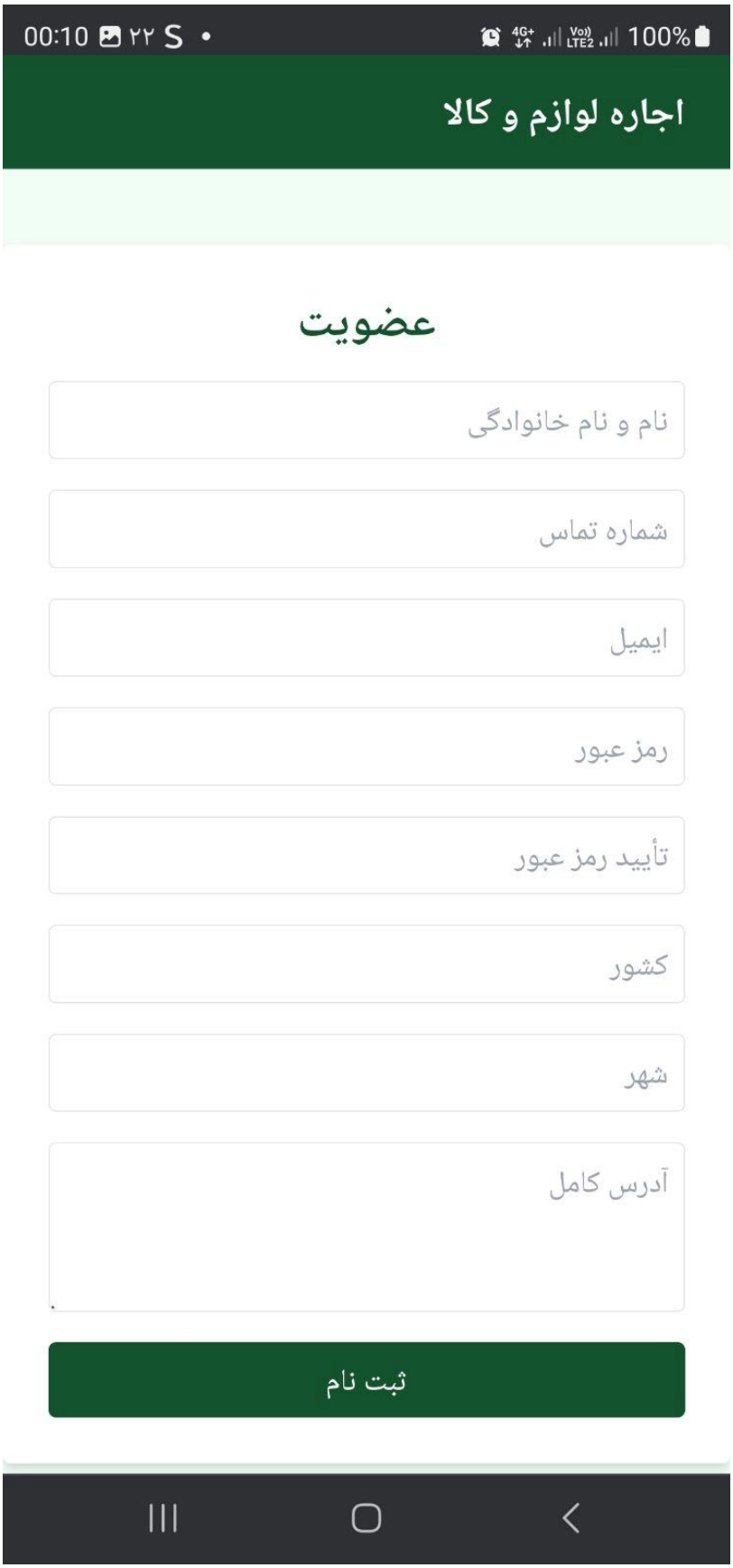
1️⃣0️⃣ طراحی و تجربه کاربری

- پیاده‌سازی با **Tailwind CSS**
- طراحی ساده، فارسی و راست‌چین
- مناسب موبایل و دسکتاپ
- هماهنگ با رنگ‌بندی سبز تیره پروژه (LoopRent)

✨ جمع‌بندی نهایی

این کامپوننت:

✅ ثبت‌نام امن کاربر
✅ استفاده از Supabase Auth
✅ ذخیره اطلاعات تکمیلی در دیتابیس
✅ اعتبارسنجی فرم
✅ تجربه کاربری مناسب
✅ قابلیت توسعه در آینده


00:10 ۲۲ S •
100%
اجاره لوازم و کالا
عضویت
نام و نام خانوادگی
شماره تماس
ایمیل
رمز عبور
تأیید رمز عبور
کشور
شهر
آدرس کامل
ثبت نام

LogoutButton.jsx

```jsx
import React from "react";
import { supabase } from "../supabaseClient";
import { useNavigate } from "react-router-dom";

const LogoutButton = () => {
  const navigate = useNavigate();

  const handleLogout = async () => {
    try {
      await supabase.auth.signOut();
      alert("👌 با موفقیت خارج شدی!");
      navigate("/login"); // یا مسیر صفحه ورودت
    } catch (error) {
      console.error("❌ خطا در خروج:", error);
      alert("مشکلی در خروج پیش آمد!");
    }
  };

  return (
    <button
      onClick={handleLogout}
      className="bg-red-600 text-white py-2 px-4 rounded hover:bg-red-700"
    >
      خروج از حساب
    </button>
  );
};

export default LogoutButton;
```

این کامپوننت خیلی ساده اما مهمه؛ **وظیفه خروج کاربر از سیستم** رو داره. توضیح مرحله به مرحله میدم:

توضیح کامپوننت LogoutButton

1️⃣ ایمپورت‌ها

```
import React from "react";
import { supabase } from "../supabaseClient";
import { useNavigate } from "react-router-dom";
```

- `React` برای ساخت کامپوننت
- `supabase` برای مدیریت احراز هویت(Auth)
- `useNavigate` برای انتقال کاربر به صفحه ورود بعد از خروج

2️⃣ تعریف کامپوننت

```
const LogoutButton = () => {
const navigate = useNavigate();
```

- `navigate` تابعی است که مسیر صفحه را تغییر می‌دهد بدون رفرش صفحه(SPA)

3️⃣ تابع خروج کاربر

```
const handleLogout = async () => {
try {
await supabase.auth.signOut();
alert("👋 با موفقیت خارج شدی!");
navigate("/login");
} catch (error) {
console.error("❌ خطا در خروج:", error);
alert("مشکلی در خروج پیش آمد!");
}
};
```

- `supabase.auth.signOut()` → کاربر را از سیستم خارج می‌کند
- در صورت موفقیت → پیام موفقیت نشان داده شده و کاربر به صفحه ورود هدایت می‌شود
- در صورت خطا → پیام خطا به کاربر و لاگ خطا در کنسول

4️⃣ دکمه رابط کاربری

```
<button onClick={handleLogout} className="bg-red-600 text-white ...">
```

```
خروج از حساب
</button>
```

- طراحی ساده و واضح
- رنگ قرمز برای نشان دادن عمل خروج
- افکت hover برای تجربه کاربری بهتر

جمع‌بندی

این کامپوننت:

✅ امکان **خروج امن کاربر** را فراهم می‌کند
✅ استفاده از **Supabase Auth** برای مدیریت نشست‌ها
✅ انتقال سریع به صفحه ورود بعد از خروج
✅ تجربه کاربری ساده، واضح و ریسپانسیو دارد

UserDashboard.jsx

```jsx
// src/components/UserDashboard.jsx
import React, { useEffect, useState } from "react";
import { supabase } from "../supabaseClient";
import { useNavigate } from "react-router-dom";
import {
  Cloud,
  CheckCircle,
  RotateCcw,
  User,
  Heart,
  Info,
  Clock,
  Wallet,
  Smile,
  ShieldCheck,
  FileCheck,
  Lock,
```

```
} from "lucide-react";

const UserDashboard = () => {
  const [userInfo, setUserInfo] = useState(null);
  const [userItems, setUserItems] = useState([]);
  const [loading, setLoading] = useState(true);
  const navigate = useNavigate();

  // ▨ گرفتن اطلاعات کاربر و آیتم‌ها از Supabase
  useEffect(() => {
    const fetchUserData = async () => {
      const {
        data: { user },
        error: userError,
      } = await supabase.auth.getUser();

      if (userError || !user) {
        console.warn("کاربر وارد نشده است.");
        navigate("/login");
        return;
      }

      try {
        // اطلاعات پروفایل کاربر
        const { data, error } = await supabase
          .from("users")
          .select("*")
          .eq("id", user.id)
          .single();

        if (error) throw error;
        setUserInfo(data);
      } catch (err) {
        console.error("❌ خطا در دریافت اطلاعات کاربر:", err);
      }
    };

    const fetchItems = async () => {
      try {
        const { data, error } = await supabase.from("items").select("*");
        if (error) throw error;
        setUserItems(data);
      } catch (err) {
        console.error("❌ خطا در دریافت کالاها:", err);
      } finally {
```

```
        setLoading(false);
      }
    };

    fetchUserData();
    fetchItems();
  }, [navigate]);

  if (loading)
    return <p className="text-center mt-10 text-green-800">در حال بارگذاری...</p>;

  return (
    <div className="min-h-screen bg-green-50" dir="rtl">
      {/* ✅ هدر */}
      <div className="w-full bg-green-100 text-green-900 py-8 px-8 text-right
shadow-sm border-b border-green-200">
        <h1 className="text-4xl font-extrabold tracking-wide">حساب کاربری من</h1>
        <p className="text-lg mt-2 opacity-80">
          {userInfo ? `${userInfo.name || ""} 🌿` : "به حساب خود خوش آمدی 🌿"}
        </p>
      </div>

      <div className="max-w-5xl mx-auto py-10 px-6 text-right">
        {/* ✅ کارت‌های بالای داشبورد */}
        <div className="grid grid-cols-1 sm:grid-cols-3 gap-6 mb-10">
          <DashboardCard icon={<Cloud size={52} />} title="جاری" dark />
          <DashboardCard icon={<CheckCircle size={52} />} title="مرجوع شده" />
          <DashboardCard
            icon={<RotateCcw size={52} />}
            title="در انتظار مرجوعی"
            yellow
          />
        </div>

        {/* ✅ منوی اطلاعات */}
        <div
          className="flex flex-col gap-3 mb-10 text-right text-lg font-semibold"
          dir="rtl"
        >
          <MenuButton
            onClick={() => navigate("/dashboard/items")}
            icon={<User size={28} />}
            label="کالاهای من"
          />
          <MenuButton
```

```
            onClick={() => navigate("/dashboard/requests")}
            icon={<Heart size={28} />}
            label="استعلام‌ها"
          />
          <MenuButton
            onClick={() => navigate("/dashboard/profile")}
            icon={<Info size={28} />}
            label="اطلاعات حساب کاربری (پروفایل من)"
          />
        </div>

        {/* ✅ مزایا */}
        <div className="grid grid-cols-1 sm:grid-cols-3 gap-6 text-center mt-16">
          <Feature icon={<Smile size={60} />} text="آسودگی خاطر" />
          <Feature icon={<Wallet size={60} />} text="صرفه‌جویی در هزینه‌ها" />
          <Feature icon={<Clock size={60} />} text="عدم وجود محدودیت زمانی" />
          <Feature icon={<ShieldCheck size={60} />} text="سیستم احراز هویت" />
          <Feature icon={<FileCheck size={60} />} text="مستندسازی سلامت کالا" />
          <Feature icon={<Lock size={60} />} text="اجاره امن با تضمین دیجیتال" />
        </div>
      </div>
    </div>
  );
};

// 🟢 کامپوننت‌های کمکی برای تمیزی کد
const DashboardCard = ({ icon, title, dark, yellow }) => (
  <div
    className={`${
      dark
        ? "bg-green-900 text-white"
        : yellow
        ? "bg-white border border-yellow-400 text-yellow-700"
        : "bg-white border border-green-300 text-green-800"
    } rounded-2xl flex flex-col items-center justify-center shadow-lg transition-transform hover:scale-105`}
    style={{ width: "280px", height: "280px", margin: "0 auto" }}
  >
    {icon}
    <p className="mt-3 font-extrabold text-2xl tracking-wide">{title}</p>
  </div>
);

const MenuButton = ({ onClick, icon, label }) => (
  <button
```

```
    onClick={onClick}
    className="flex items-center justify-start gap-3 text-green-800
hover:underline"
  >
    {icon}
    {label}
  </button>
);

const Feature = ({ icon, text }) => (
  <div className="flex flex-col items-center justify-center text-green-900">
    <div className="bg-green-800 text-white p-3 rounded-full mb-3">{icon}</div>
    <p className="font-semibold text-lg">{text}</p>
  </div>
);

export default UserDashboard;
```

کامپوننت `UserDashboard` یکی از بخش‌های اصلی پروژه شماست. این داشبورد **اطلاعات کاربر، آیتم‌ها و ویژگی‌های پلتفرم** را نمایش می‌دهد. توضیح کامل و مرحله‌به‌مرحله:

توضیح کامپوننت `UserDashboard`

✅ ایمپورت‌ها

```
import React, { useEffect, useState } from "react";
import { supabase } from "../supabaseClient";
import { useNavigate } from "react-router-dom";
import { Cloud, CheckCircle, RotateCcw, User, Heart, Info, Clock, Wallet,
Smile, ShieldCheck, FileCheck, Lock } from "lucide-react";
```

- `useState` و `useEffect` برای مدیریت داده‌ها و اجرای فانکشن‌ها هنگام بارگذاری
- `supabase` برای گرفتن اطلاعات کاربر و آیتم‌ها
- `useNavigate` برای هدایت کاربر بین صفحات
- آیکون‌ها از **lucide-react** برای زیبایی داشبورد

✅ State های اصلی

```
const [userInfo, setUserInfo] = useState(null);
const [userItems, setUserItems] = useState([]);
  const [loading, setLoading] = useState(true);
```

- `userInfo` → اطلاعات پروفایل کاربر
- `userItems` → لیست کالاهای موجود یا ثبت‌شده
- `loading` → نمایش حالت بارگذاری

3️⃣ گرفتن اطلاعات کاربر و آیتم‌ها

```
useEffect(() => {
const fetchUserData = async () => { ... }
   const fetchItems = async () => { ... }
fetchUserData();
   fetchItems();
 }, [navigate]);
```

- `fetchUserData()`
  - گرفتن اطلاعات کاربر از Supabase Auth
  - سپس گرفتن جزئیات پروفایل از جدول `users`
  - اگر کاربر وارد نشده باشد → هدایت به صفحه ورود
- `fetchItems()`
  - گرفتن همه آیتم‌ها از جدول `items`
  - بعد از دریافت → خاموش شدن حالت `loading`

4️⃣ نمایش حالت بارگذاری

```
if (loading)
return <p className="text-center mt-10 text-green-800">در حال بارگذاری...</p>;
```

- تا زمان دریافت اطلاعات، پیام "در حال بارگذاری..." نمایش داده می‌شود.

5️⃣ هدر داشبورد

```
<h1 className="text-4xl font-extrabold tracking-wide">حساب کاربری من</h1>
<p>{userInfo ? `به حساب خود خوش آمدی ${userInfo.name || ""}` : "🌿"}</p>
```

- نام کاربر نمایش داده می‌شود
- اگر اطلاعات کاربر موجود نباشد → پیام خوش‌آمدگویی عمومی نمایش داده می‌شود

6️⃣کارت‌های بالای داشبورد

```
" dark /><DashboardCard icon={<Cloud /> title="جاری
" /><DashboardCard icon={<CheckCircle /> title="مرجوع شده
" yellow /><DashboardCard icon={<RotateCcw /> title="در انتظار مرجوعی
```

- نمایانگر وضعیت سفارش‌ها و آیتم‌ها
- از کامپوننت کمکی `DashboardCard` استفاده شده
- کارت‌ها قابلیت hover و بزرگ‌نمایی دارند

7️⃣منوی اطلاعات

```
<MenuButton onClick={() => navigate("/dashboard/items")} icon={<User />}
label="کالاهای من/> "
<MenuButton onClick={() => navigate("/dashboard/requests")} icon={<Heart />}
label="استعلام ها/> "
<MenuButton onClick={() => navigate("/dashboard/profile")} icon={<Info />}
label="اطلاعات حساب کاربری/> "
```

- دکمه‌های ناوبری به صفحات مختلف داشبورد
- آیکون و متن برای خوانایی بهتر

8️⃣بخش مزایا و ویژگی‌ها

```
" text={<Smile /> icon={<Feature>آسودگی خاطر/> "
" text={<Wallet /> icon={<Feature>صرفه‌جویی در هزینه ها/> "
...
```

- معرفی ویژگی‌های پلتفرم به کاربر
- آیکون‌ها + متن برای تجربه کاربری جذاب
- از کامپوننت `Feature` استفاده شده

9️⃣کامپوننت‌های کمکی

DashboardCard

- کارت وضعیت آیتم‌ها
- استایل با Tailwind
- رنگ‌بندی متنوع `dark` :، `yellow`، حالت پیش‌فرض

MenuButton

- دکمه ناوبری در داشبورد
- آیکون + متن

Feature

- نمایش مزایای پلتفرم
- آیکون در یک دایره رنگی + متن

---

10 طراحی و تجربه کاربری

- کاملاً راست‌چین و مناسب فارسی
- ریسپانسیو(desktop + mobile)
- استفاده از Tailwind CSS و افکت‌هایhover
- رنگ‌بندی هماهنگ با برند سبز پروژه

جمع‌بندی کاربردی

کامپوننت :`UserDashboard`

✅ **نمایش اطلاعات پروفایل کاربر**
✅ **نمایش وضعیت سفارش‌ها و آیتم‌ها**
✅ **منوی ناوبری داخلی داشبورد**
✅ **معرفی مزایا و ویژگی‌های پلتفرم**
✅ تجربه کاربری حرفه‌ای و ریسپانسیو

UserProfile.jsx

```jsx
// ✅ src/components/UserProfile.jsx
import React, { useState, useEffect } from "react";
import { supabase } from "../supabaseClient";
import { Pencil, Loader2 } from "lucide-react";

export default function UserProfile() {
  const [user, setUser] = useState(null);
  const [editing, setEditing] = useState(false);
  const [newAddress, setNewAddress] = useState("");
  const [loading, setLoading] = useState(true);
  const [uploading, setUploading] = useState(false);
  const [message, setMessage] = useState("");

  // 🟢 دریافت اطلاعات کاربر
  useEffect(() => {
```

```
  const fetchUser = async () => {
    setLoading(true);
    const { data: { user } } = await supabase.auth.getUser();
    if (!user) {
      setMessage("لطفاً وارد حساب خود شوید ❌");
      setLoading(false);
      return;
    }

    const { data, error } = await supabase
      .from("users")
      .select("*")
      .eq("id", user.id)
      .single();

    if (error) {
      console.error(error);
      setMessage("خطا در دریافت اطلاعات کاربر ❌");
    } else {
      setUser(data);
    }
    setLoading(false);
  };

  fetchUser();
}, []);

// 🟢 ذخیره تغییرات
const handleSave = async () => {
  if (!user) return;

  const { error } = await supabase.from("users").update({
    name: user.name,
    phone: user.phone,
    country: user.country,
    city: user.city,
    address: user.address,
  }).eq("id", user.id);

  if (error) {
    setMessage("خطا در ذخیره تغییرات ❌");
    console.error(error);
  } else {
    setEditing(false);
    setMessage("✅ تغییرات ذخیره شد!");
```

```
  }

  setTimeout(() => setMessage(""), 3000);
};

// 🟢 آپلود عکس پروفایل
const handleImageUpload = async (e) => {
  const file = e.target.files[0];
  if (!file || !user) return;

  try {
    setUploading(true);

    const fileName = `${user.id}-${Date.now()}.jpg;`
    const { error: uploadError } = await supabase.storage
      .from("avatars")
      .upload(fileName, file, { upsert: true });

    if (uploadError) throw uploadError;

    const { data: publicUrlData } = supabase.storage
      .from("avatars")
      .getPublicUrl(fileName);

    const publicUrl = publicUrlData.publicUrl;

    const { error: updateError } = await supabase
      .from("users")
      .update({ avatar_url: publicUrl })
      .eq("id", user.id);

    if (updateError) throw updateError;

    setUser({ ...user, avatar_url: publicUrl });
    setMessage("✅ تصویر پروفایل به‌روزرسانی شد!");
  } catch (err) {
    console.error(err);
    setMessage("❌ خطا در آپلود تصویر");
  } finally {
    setUploading(false);
  }
};

if (loading)
  return (
```

```
      <p className="text-center mt-20 text-gray-500 animate-pulse">
        در حال بارگذاری اطلاعات...
      </p>
    );

  if (!user)
    return (
      <p className="text-center mt-20 text-red-600">
        کاربر یافت نشد ❌
      </p>
    );

  return (
    <div
      className="flex justify-center py-12 px-4 bg-[#f9f3ec] min-h-screen"
      dir="rtl"
    >
      <div className="bg-[#fffaf4] shadow-md rounded-3xl p-8 w-full max-w-3xl
text-right relative">
        {message && (
          <div className="absolute top-3 right-3 bg-green-100 border border-
green-700 text-green-900 px-4 py-2 rounded-lg text-sm shadow">
            {message}
          </div>
        )}

        {/* عکس پروفایل */}
        <div className="flex items-center justify-between mb-8">
          <div className="flex items-center gap-4">
            <label htmlFor="profile-pic" className="cursor-pointer relative">
              <img
                src={
                  user.avatar_url ||
"https://via.placeholder.com/100?text=Avatar"
                }
                alt="Profile"
                className="w-24 h-24 rounded-full border-4 border-[#3d3a2f]
object-cover"
              />
              {uploading && (
                <span className="absolute inset-0 flex items-center justify-
center bg-white bg-opacity-70 rounded-full">
                  <Loader2 className="animate-spin text-[#3d3a2f]" size={28} />
                </span>
              )}
```

```
              <input
                id="profile-pic"
                type="file"
                accept="image/*"
                className="hidden"
                onChange={handleImageUpload}
              />
            </label>
            <div>
              <h2 className="text-2xl font-bold text-[#3d3a2f]">
                {user.name || "بدون نام"}
              </h2>
              <p className="text-gray-600">{user.email}</p>
            </div>
          </div>

          <button
            onClick={() => setEditing(!editing)}
            className="bg-[#2f5d3b] hover:bg-[#264e31] text-white px-6 py-2
rounded-xl transition flex items-center gap-2"
          >
            <Pencil size={18} />
            {editing ? "انصراف" : "ویرایش"}
          </button>
        </div>

        {/* فرم اطلاعات کاربری */}
        <div className="grid grid-cols-1 sm:grid-cols-2 gap-5 mb-8">
          <Field label="نام" name="name" value={user.name} editable={editing}
onChange={setUser} />
          <Field label="شماره تماس" name="phone" value={user.phone}
editable={editing} onChange={setUser} />
          <Field label="کشور" name="country" value={user.country}
editable={editing} onChange={setUser} />
          <Field label="شهر" name="city" value={user.city} editable={editing}
onChange={setUser} />
          <Field label="آدرس" name="address" value={user.address}
editable={editing} onChange={setUser} full />
        </div>

        {editing && (
          <div className="flex justify-end">
            <button
              onClick={handleSave}
```

```
              className="bg-[#2f5d3b] hover:bg-[#264e31] text-white px-6 py-2
rounded-xl transition"
            >
              ذخیره تغییرات
            </button>
          </div>
        )}
      </div>
    </div>
  );
}

// 🟢 کامپوننت فیلد ورودی
function Field({ label, name, value, editable, onChange, full }) {
  return (
    <div className={full ? "col-span-2" : ""}>
      <p className="text-sm text-[#7c6d58] mb-1">{label}</p>
      {editable ? (
        <input
          className="border border-[#e0cdb2] bg-[#fffdf9] rounded-xl py-2 px-4
text-[#3d3a2f] focus:ring-2 focus:ring-[#2f5d3b] w-full"
          value={value || ""}
          onChange={(e) => onChange((prev) => ({ ...prev, [name]: e.target.value
}))}
        />
      ) : (
        <div className="border border-[#e0cdb2] bg-[#fffdf9] rounded-xl py-2 px-4
text-[#3d3a2f] w-full">
          {value || "—"}
        </div>
      )}
    </div>
  );
}
```

این کامپوننت `UserProfile` **صفحه پروفایل کاربر** را مدیریت می‌کند و قابلیت‌های زیر را ارائه می‌دهد: مشاهده اطلاعات، ویرایش، ذخیره تغییرات و آپلود عکس پروفایل. توضیح مرحله‌به‌مرحله:

🟢 توضیح کامپوننت `UserProfile`

◻ ایمپورت‌ها

```
import React, { useState, useEffect } from "react";
import { supabase } from "../supabaseClient";
import { Pencil, Loader2 } from "lucide-react";
```

- useStateو useEffectبرای مدیریت state و اجرای فانکشن هنگام بارگذاری
- supabaseبرای احراز هویت و دسترسی به جدول usersو استوریج
- Pencilو Loader2برای آیکون و نمایش حالت لودینگ

2⃣ State های اصلی

```
const [user, setUser] = useState(null);      // اطلاعات کاربر
const [editing, setEditing] = useState(false); // حالت ویرایش فعال/غیرفعال
const [loading, setLoading] = useState(true); // لودینگ اولیه
const [uploading, setUploading] = useState(false); // وضعیت آپلود تصویر
const [message, setMessage] = useState(""); // پیام موفقیت یا خطا
```

- مدیریت نمایش و تغییر داده‌ها
- editingبرای کنترل فعال شدنinput ها

3⃣ دریافت اطلاعات کاربر

```
useEffect(() => {
  const fetchUser = async () => {
    setLoading(true);
    const { data: { user } } = await supabase.auth.getUser();
    if (!user) { setMessage("لطفاً وارد حساب خود شوید ❌"); setLoading(false); return; }

    const { data, error } = await supabase.from("users").select("*").eq("id", user.id).single();
    if (error) { setMessage("خطا در دریافت اطلاعات کاربر ❌"); console.error(error); }
    else { setUser(data); }
    setLoading(false);
  };
  fetchUser();
}, []);
```

- بررسی ورود کاربر
- گرفتن اطلاعات کاربر از جدول users
- اگر کاربر وجود نداشته باشد ⟶ پیام خطا نمایش داده می‌شود

4⃣ ذخیره تغییرات اطلاعات

```
const handleSave = async () => {
  if (!user) return;

  const { error } = await supabase.from("users").update({
    name: user.name,
    phone: user.phone,
    country: user.country,
    city: user.city,
    address: user.address,
  }).eq("id", user.id);

  if (error) { setMessage("❌ خطا در ذخیره تغییرات"); }
  else { setEditing(false); setMessage("✅ تغییرات ذخیره شد!"); }

  setTimeout(() => setMessage(""), 3000);
};
```

- بروزرسانی اطلاعات کاربر در جدول `users`
- پیام موفقیت یا خطا به کاربر نمایش داده می‌شود

5️⃣ آپلود عکس پروفایل

```
const handleImageUpload = async (e) => {
  const file = e.target.files[0];
  if (!file || !user) return;

  setUploading(true);
  const fileName = `${user.id}-${Date.now()}.jpg`;
  await supabase.storage.from("avatars").upload(fileName, file, { upsert: true });
  const { data: publicUrlData } = supabase.storage.from("avatars").getPublicUrl(fileName);
  await supabase.from("users").update({ avatar_url: publicUrlData.publicUrl }).eq("id", user.id);

  setUser({ ...user, avatar_url: publicUrlData.publicUrl });
  setMessage("✅ تصویر پروفایل به‌روزرسانی شد!");
  setUploading(false);
};
```

- آپلود تصویر در استوریج Supabase
- گرفتن URL عمومی و ذخیره در جدول `users`
- نمایش پیام موفقیت و وضعیت لودینگ حین آپلود

6️⃣ نمایش حالت‌ها

```
if (loading) return <p className="text-center mt-20 animate-pulse">در حال بارگذاری...</p>;
if (!user) return <p className="text-center mt-20 text-red-600">کاربر یافت نشد ❌</p>;
```

- نمایش پیام لودینگ یا خطا در صورت عدم وجود کاربر

7️⃣ رابط کاربری پروفایل

- **عکس پروفایل** با قابلیت کلیک و انتخاب فایل برای آپلود
- **نام، ایمیل و اطلاعات کاربری** نمایش داده می‌شود
- دکمه **ویرایش/انصراف** برای فعال کردن input ها
- فرم اطلاعات با کامپوننت کمکی `Field`:

```
<Field label="نام" name="name" value={user.name} editable={editing} onChange={setUser} />
```

- input ها فقط در حالت `editing` قابل تغییر هستند
- دکمه ذخیره تغییرات فعال می‌شود در صورت فعال بودن ویرایش

8️⃣ کامپوننت کمکی `Field`

```
function Field({ label, name, value, editable, onChange, full }) { ... }
```

- نمایش **label + input**
- اگر `editable` باشد input → فعال است
- اگر `editable` نباشد → فقط نمایش داده می‌شود
- `full` برای span دو ستون در grid

9️⃣ طراحی و UX

- راست‌چین و مناسب زبان فارسی
- استفاده از Tailwind CSS برای استایل، hover و focus
- نمایش پیام موفقیت/خطا به صورت toast کوچک
- نمایش Loader حین آپلود تصویر

✨ جمع‌بندی

کامپوننت :`UserProfile`

- ✅نمایش اطلاعات کاربر
- ✅ویرایش اطلاعات و ذخیره به دیتابیس
- ✅آپلود و به‌روزرسانی تصویر پروفایل
- ✅پیام‌های موفقیت و خطا
- ✅طراحی ریسپانسیو و حرفه‌ای

HomePage.jsx

```jsx
// src/components/HomePage.jsx
import React, { useState, useEffect } from "react";
import { createClient } from "@supabase/supabase-js";
import {
  MoreVertical,
  ChevronDown,
  Clock,
  ShieldCheck,
  Wallet,
  Heart,
  CheckCircle,
  Lock,
} from "lucide-react";
import { motion, AnimatePresence } from "framer-motion";

import pan from "../assets/pan.jpg";
import tv from "../assets/tv.jpg";
import garden from "../assets/garden.jpg";
import laptop from "../assets/laptop.jpg";
import HeroSection from "./HeroSection";

// === تنظیمات Supabase — اگر کلاینت رو جای دیگری می‌سازی فقط اینجا رو عوض کن ===
const supabaseUrl = "https://atggpdceswwcfujmjxag.supabase.co";
const supabaseAnonKey =
  "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImF0Z2dwZ
GNlc3d3Y2Z1am1qeGFnIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NjI1MDYxNDUsImV4cCI6MjA3ODA4Mj
E0NX0.FXQvRhyvKVFobFTvqGmM1iL3tduk8_KUhGC7z0n_yv0";
const supabase = createClient(supabaseUrl, supabaseAnonKey);
// ================================================================

export default function HomePage({ onLogin, onRegister }) {
  const [menuOpen, setMenuOpen] = useState(false);
  const [activeCategory, setActiveCategory] = useState(null);
  const [selectedSubcategory, setSelectedSubcategory] = useState(null);
  const [items, setItems] = useState([]); // آیتم‌هایی که فعلاً نمایش داده می‌شوند
  const [loading, setLoading] = useState(false);

  // گرفتن جدیدترین آیتم‌ها برای صفحه اصلی (بدون فیلتر)
  const fetchLatestItems = async () => {
    try {
```

```
    setLoading(true);
    const { data, error } = await supabase
      .from("items")
      .select("*")
      .order("created_at", { ascending: false })
      .limit(12);
    if (error) throw error;
    setItems(data || []);
  } catch (err) {
    console.error("خطا در دریافت جدیدترین کالاها:", err.message || err);
    setItems([]);
  } finally {
    setLoading(false);
  }
};

// گرفتن کالاها بر اساس زیردسته (فیلتر)
const fetchItems = async (subcategory) => {
  try {
    setLoading(true);
    setSelectedSubcategory(subcategory);
    // توجه: قالب درست برای ilike با بک‌تیک یا رشته ساخته شده
    const { data, error } = await supabase
      .from("items")
      .select("*")
      .ilike("category", `%${subcategory}%`)
      .order("created_at", { ascending: false });
    if (error) throw error;
    setItems(data || []);
  } catch (err) {
    console.error("خطا در دریافت کالا:", err.message || err);
    setItems([]);
  } finally {
    setLoading(false);
  }
};

// موقع بارگذاری کامپوننت، جدیدترین‌ها رو بارگذاری کن
useEffect(() => {
  fetchLatestItems();
}, []);

// ساختار دسته‌ها و زیردسته‌ها (همون چیزی که قبلاً داشتی)
const categories = [
  {
```

```
    title: "🛠 ابزار و تجهیزات",
    items: [
      "ابزار برقی (دریل، فرز، اره)",
      "ابزار دستی (چکش، پیچ‌گوشتی، آچار)",
      "تجهیزات ساختمانی (نردبان، داربست، بالابر)",
      "ابزار اندازه‌گیری (متر لیزری، تراز، کولیس)",
      "تجهیزات ایمنی (کلاه، دستکش، عینک)",
      "دستگاه‌های صنعتی (کمپرسور، جوشکاری، برش CNC)",
    ],
  },
  {
    title: "☎ دوربین و عکاسی",
    items: [
      "دوربین DSLR و بدون آینه",
      "لنزهای تخصصی (وایدتله، ماکرو)",
      "سه‌پایه و استابلایزر",
      "فلاش و نورپردازی",
      "تجهیزات بک‌گراند و پرده",
      "کارت حافظه و باتری یدکی",
    ],
  },
  {
    title: "🏕 کمپینگ و سفر",
    items: [
      "چادر و کیسه خواب",
      "کوله‌پشتی و تجهیزات کوهنوردی",
      "اجاق گاز سفری و ظروف کمپینگ",
      "چراغ قوه و پاوربانک",
      "صندلی و میز تاشو",
      "تجهیزات ماهیگیری و طبیعت‌گردی",
    ],
  },
  {
    title: "👗 لباس و لوازم مجلسی",
    items: [
      "لباس شب و مجلسی زنانه",
      "کت و شلوار و لباس رسمی مردانه",
      "لباس کودک برای مراسم",
      "کیف و کفش مجلسی",
      "اکسسوری (جواهرات، کراوات، شال)",
      "تاج و تور عروس",
    ],
  },
  {
    title: "🏠 لوازم خانگی",
```

```
    items: [
      "یخچال، فریزر، ماشین لباسشویی",
      "اجاق گاز، مایکروویو، پلوپز",
      "تلویزیون، کولر، بخاری",
      "جاروبرقی، اتو، پنکه",
      "مبلمان، تخت خواب، فرش",
      "ظروف آشپزخانه و سرویس غذاخوری",
    ],
  },
  {
    title: "🎉 تجهیزات مجالس",
    items: [
      "میز و صندلی مهمانی",
      "سیستم صوتی و نورپردازی",
      "سفره عقد و جایگاه عروس",
      "چادر، سایه‌بان، کولر و بخاری",
      "ظروف پذیرایی و یخچال نوشیدنی",
      "استند، بنر، میز پذیرش",
    ],
  },
  {
    title: "💻 دیجیتال و اداری",
    items: [
      "لپ‌تاپ و کامپیوتر رومیزی",
      "پرینتر، اسکنر، دستگاه کپی",
      "پروژکتور و پرده نمایش",
      "مانیتور، تبلت، موبایل",
      "تجهیزات شبکه و مودم",
      "دوربین وب‌کم و میکروفون",
    ],
  },
  {
    title: "🚗 وسایل نقلیه",
    items: [
      "خودرو سواری (اقتصادی، لوکس)",
      "موتور سیکلت و دوچرخه",
      "وانت و کامیونت",
      "ون و مینی‌بوس",
      "تجهیزات باربری (تریلر، چرخ‌دستی)",
      "اسکوتر برقی و وسایل نقلیه سبک",
    ],
  },
  {
    title: "🎮 سرگرمی و آموزشی",
    items: [
```

```
        "بازی‌های فکری و گروهی",
        "ساز های موسیقی (گیتار، کیبورد، سنتور)",
        "تجهیزات ورزشی (تردمیل، دمبل، توپ)",
        "کتاب و جزوه آموزشی",
        "وسایل نقاشی و طراحی",
        "وسایل آموزشی کودک و نوجوان",
      ],
    },
  ];

  const advantages = [
    { icon: <Clock className="w-8 h-8 text-green-800" />, title: "صرفه‌جویی در زمان"
},
    { icon: <Wallet className="w-8 h-8 text-green-800" />, title: "کاهش هزینه‌ها" },
    { icon: <Heart className="w-8 h-8 text-green-800" />, title: "آسودگی خاطر" },
    { icon: <ShieldCheck className="w-8 h-8 text-green-800" />, title: " سیستم احراز
هویت" },
    { icon: <CheckCircle className="w-8 h-8 text-green-800" />, title: " اطمینان از
سلامت کالا" },
    { icon: <Lock className="w-8 h-8 text-green-800" />, title: " اجاره امن با تضمین
دیجیتال" },
  ];

  return (
    <div className="min-h-screen bg-[#f9faf9] text-right p-4">
      <HeroSection />

      {/* منوی دسته‌ها */}
      <div className="relative flex justify-end mt-4">
        <button
          onClick={() => setMenuOpen(!menuOpen)}
          className="p-2 bg-green-100 rounded-full hover:bg-green-200"
        >
          <MoreVertical className="text-green-800" />
        </button>

        <AnimatePresence>
          {menuOpen && (
            <motion.div
              initial={{ opacity: 0, y: -10 }}
              animate={{ opacity: 1, y: 0 }}
              exit={{ opacity: 0, y: -10 }}
              className="absolute top-12 left-0 bg-white border border-green-200
rounded-xl shadow-lg w-72 max-h-[80vh] overflow-y-auto z-50"
            >
```

```
            {categories.map((cat, index) => (
              <div key={index} className="border-b border-gray-100">
                <button
                  onClick={() => setActiveCategory(activeCategory === index ?
null : index)}
                  className="flex justify-between items-center w-full px-4 py-3
text-green-900 hover:bg-green-50"
                >
                  <span>{cat.title}</span>
                  <ChevronDown
                    className={`transition-transform duration-300
${activeCategory === index ? "rotate-180" : ""}`}
                  />
                </button>

                <AnimatePresence>
                  {activeCategory === index && (
                    <motion.ul
                      initial={{ opacity: 0, height: 0 }}
                      animate={{ opacity: 1, height: "auto" }}
                      exit={{ opacity: 0, height: 0 }}
                      className="bg-green-50 text-sm px-6 pb-3 space-y-2"
                    >
                      {cat.items.map((item, i) => (
                        <li
                          key={i}
                          className="text-green-800 cursor-pointer hover:text-
green-600"

                          onClick={() => fetchItems(item)}
                        >
                          • {item}
                        </li>
                      ))}
                    </motion.ul>
                  )}
                </AnimatePresence>
              </div>
            ))}
          </motion.div>
        )}
      </AnimatePresence>
    </div>

    {/* اگر کاربری زیردسته انتخاب کرده، فهرست آن زیردسته را نمایش بده */}
    {selectedSubcategory && (
```

```
        <section className="mt-10">
          <h2 className="text-xl font-bold text-green-900 mb-4 text-center">
            کالاهای دسته: {selectedSubcategory}
          </h2>

          {loading ? (
            <p className="text-center text-green-800">در حال بارگذاری...</p>
          ) : items.length === 0 ? (
            <p className="text-center text-gray-500">کالایی برای این دسته پیدا نشد.</p>
          ) : (
            <div className="grid grid-cols-2 sm:grid-cols-3 lg:grid-cols-4 gap-
6">
              {items.map((item) => (
                <div key={item.id} className="bg-white rounded-2xl shadow p-4
text-center hover:shadow-md transition">
                  <img src={item.image_url} alt={item.title} className="w-32 h-32
object-cover mx-auto mb-3 rounded-xl" />
                  <h3 className="text-green-900 font-bold text-
lg">{item.title}</h3>
                  <p className="text-gray-500 text-sm line-clamp-
2">{item.description}</p>
                  <p className="text-green-700 font-semibold mt-
2">{item.price_per_day} تومان / روز</p>
                </div>
              ))}
            </div>
          )}
        </section>
      )}

      {/* چهار تصویر دسته + پایینشان جدیدترین کالاها */}
      <section className="grid grid-cols-2 sm:grid-cols-4 gap-6 my-10">
        {[{ img: pan, label: "لوازم آشپزخانه" }, { img:
garden, label: "ابزار باغبانی" }, { img: tv, label: "لوازم برقی منزل" }, { img: laptop,
label: "تجهیزات دیجیتال" }].map((box, i) => (
          <div key={i} className="bg-white rounded-2xl shadow p-4 text-center
hover:shadow-md transition">
            <img src={box.img} alt={box.label} className="w-24 h-24 mx-auto mb-3
rounded-xl" />
            <p className="text-green-900 font-medium">{box.label}</p>
          </div>
        ))}
      </section>

      {/* اینجا جدیدترین کالاها (صفحه اصلی) */}
```

```
      <section className="mt-12">
        <h2 className="text-2xl font-bold text-green-900 mb-6 text-center">
          {selectedSubcategory ?` کالاهای دسته: ${selectedSubcategory} `: "جدیدترین کالاهای اجاره‌ای"}
        </h2>

        {loading ? (
          <p className="text-center text-green-800">در حال بارگذاری...</p>
        ) : (
          <div className="grid grid-cols-2 sm:grid-cols-3 lg:grid-cols-4 gap-6">
            {items.length > 0 ? (
              items.map((item) => (
                <div key={item.id} className="bg-white rounded-2xl shadow p-4 text-center hover:shadow-md transition">
                  <img src={item.image_url} alt={item.title} className="w-32 h-32 object-cover mx-auto mb-3 rounded-xl" />
                  <h3 className="text-green-900 font-bold text-lg">{item.title}</h3>
                  <p className="text-gray-500 text-sm line-clamp-2">{item.description}</p>
                  <p className="text-green-700 font-semibold mt-2">{item.price_per_day} تومان / روز</p>
                </div>
              ))
            ) : (
              <p className="text-center text-gray-500 col-span-full">هنوز کالایی ثبت نشده است.</p>
            )}
          </div>
        )}
      </section>

      {/* مزایا */}
      <section className="mt-10">
        <h2 className="text-2xl font-bold text-green-900 mb-6 text-center">مزایای استفاده</h2>
        <div className="grid grid-cols-2 sm:grid-cols-3 gap-6 text-center">
          {advantages.map((adv, i) => (
            <div key={i} className="flex flex-col items-center justify-center bg-green-50 rounded-2xl p-4 shadow-sm hover:shadow-md transition">
              <div className="mb-3 bg-green-100 p-3 rounded-full">{adv.icon}</div>
              <p className="text-green-900 font-medium">{adv.title}</p>
            </div>
          ))}
```

```
        </div>
      </section>
    </div>
  );
}
```

کامپوننت HomePage **صفحه اصلی سایت اجاره کالا** است که وظایف اصلی آن نمایش جدیدترین کالاها، فیلتر کردن بر اساس دسته و زیردسته، نمایش تصاویر دسته‌بندی و مزایا است. توضیح مرحله‌به‌مرحله:

توضیح کامپوننت HomePage

ایمپورت‌ها

```
import React, { useState, useEffect } from "react";
import { createClient } from "@supabase/supabase-js";
import { motion, AnimatePresence } from "framer-motion";
import { MoreVertical, ChevronDown, Clock, Wallet, Heart, ShieldCheck,
CheckCircle, Lock } from "lucide-react";
import pan from "../assets/pan.jpg";
import tv from "../assets/tv.jpg";
import garden from "../assets/garden.jpg";
import laptop from "../assets/laptop.jpg";
import HeroSection from "./HeroSection";
```

- Supabase برای گرفتن اطلاعات کالاها
- Framer Motion برای انیمیشن منوها و باز/بسته شدن دسته‌ها
- Lucide-React برای آیکون‌ها
- تصاویر دسته‌ها و HeroSection برای بخش بالای صفحه

اتصال به Supabase

```
const supabase = createClient(supabaseUrl, supabaseAnonKey);
```

- اتصال به پایگاه داده برای خواندن اطلاعات کالاها
- استفاده از anonKey برای دسترسی عمومی (خواندن اطلاعات)

3️⃣ State‌های اصلی

```
const [menuOpen, setMenuOpen] = useState(false); // باز/بسته بودن منوی دسته‌ها
const [activeCategory, setActiveCategory] = useState(null); // دسته فعال در منو
const [selectedSubcategory, setSelectedSubcategory] = useState(null); // زیردسته انتخاب‌شده
const [items, setItems] = useState([]); // لیست کالاها برای نمایش
const [loading, setLoading] = useState(false); // وضعیت بارگذاری
```

- مدیریت منو، فیلتر دسته‌بندی و لودینگ کالاها

4️⃣ گرفتن کالاها از Supabase

جدیدترین کالاها:

```
const fetchLatestItems = async () => {
  setLoading(true);
  const { data, error } = await supabase.from("items").select("*").order("created_at", { ascending: false }).limit(12);
  setItems(data || []);
  setLoading(false);
};
```

- محدود به ۱۲ کالای جدید
- مرتب بر اساس زمان ایجاد (جدیدترین‌ها ابتدا)

فیلتر بر اساس زیردسته:

```
const fetchItems = async (subcategory) => {
  setLoading(true);
  setSelectedSubcategory(subcategory);
  const { data, error } = await supabase.from("items").select("*").ilike("category", `%${subcategory}%`).order("created_at", { ascending: false });
  setItems(data || []);
  setLoading(false);
};
```

- `ilike` برای جستجوی غیر حساس به حروف
- تغییر `selectedSubcategory` برای نمایش عنوان زیردسته

5️⃣ useEffect اولیه

```
useEffect(() => {
  fetchLatestItems();
}, []);
```

- هنگام بارگذاری صفحه ← نمایش جدیدترین کالاها

6️⃣تعریف دسته‌ها و زیردسته‌ها

```
const categories = [
  { title: "🛠ابزار و تجهیزات", items: ["ابزار برقی", "ابزار دستی", ... ] },
  { title: "📷دوربین و عکاسی", items: ["دوربین", "لنزهای تخصصی DSLR", ... ] },
  ...
];
```

- ساختار دسته و زیردسته‌ها برای منوی فیلتر

7️⃣تعریف مزایا

```
const advantages = [
  { icon: <Clock />, title: "صرفه‌جویی در زمان" },
  { icon: <Wallet />, title: "کاهش هزینه‌ها" },
  ...
];
```

- بخش پایین صفحه برای نمایش مزایای استفاده از پلتفرم

8️⃣رابط کاربری

منوی دسته‌ها:

- دکمه `MoreVertical`برای باز/بسته کردن منو
- `AnimatePresence + motion.div`برای انیمیشن باز و بسته شدن
- کلیک روی دسته ← باز شدن زیردسته‌ها
- کلیک روی زیردسته ← فیلتر کالاها

نمایش کالاها:

- اگر `selectedSubcategory`انتخاب شده ← نمایش کالاهای آن زیردسته
- اگر نه ← نمایش جدیدترین کالاها
- هر کارت کالا شامل:
  - تصویر(`image_url`)
  - عنوان(`title`)
  - توضیح کوتاه(`description`)

- قیمت روزانه(price_per_day)

نمایش چهار تصویر دسته‌بندی:

```
" }, ...].map(...)لـوازم آشپزخانـه[{ img: pan, label: "
```

- به‌صورت کارت با تصویر و نام دسته
- بدون فیلتر، صرفاً زیبایی و دسته‌بندی نمایشی

نمایش مزایا:

- gridبا آیکون و عنوان
- hover effectو shadow برای جلوه بهتر

9️⃣وضعیت لودینگ و خطا

- حین بارگذاری ← پیام "در حال بارگذاری"...
- اگر کالایی نباشد ← پیام "کالایی برای این دسته پیدا نشد"

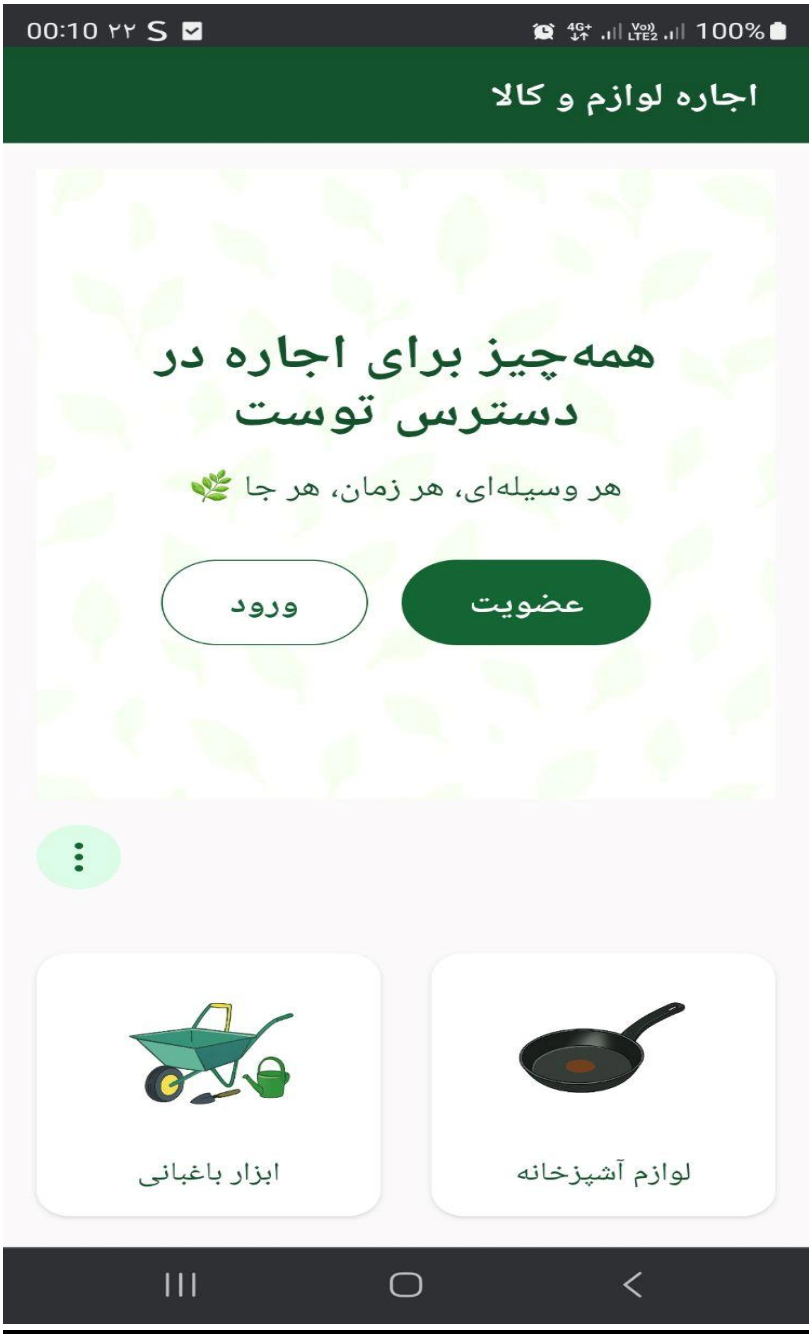
1️⃣0️⃣طراحی وUX

- راست‌چین برای زبان فارسی
- استفاده از Tailwind CSS برایgrid ، shadow، hover، rounded
- انیمیشن باز/بسته شدن منو و زیردسته‌ها با Framer Motion
- ریسپانسیو(grid-cols-2 sm:grid-cols-3 lg:grid-cols-4)

✨جمع‌بندی

کامپوننت :HomePage

✅نمایش جدیدترین کالاها و کالاهای فیلتر شده
✅فیلتر براساس دسته و زیردسته
✅نمایش تصاویر شاخص دسته‌ها
✅نمایش مزایای استفاده از پلتفرم
✅انیمیشن زیبا و UX مناسب برای کاربران


00:10 ۲۲ S
100%
اجاره لوازم و کالا
همه‌چیز برای اجاره در دسترس توست
هر وسیله‌ای، هر زمان، هر جا
عضویت
ورود
لوازم آشپزخانه
ابزار باغبانی


ابزار و تجهیزات
• ابزار برقی (دریل، فرز، اره)
• ابزار دستی (چکش، پیچ‌گوشتی، آچار)
• تجهیزات ساختمانی (نردبان، داربست، بالابر)
• ابزار اندازه‌گیری (متر لیزری، تراز، کولیس)
• تجهیزات ایمنی (کلاه، دستکش، عینک)
• دستگاه‌های صنعتی (کمپرسور، جوشکاری، برش CNC)
دوربین و عکاسی
کمپینگ و سفر
لباس و لوازم مجلسی
لوازم خانگی


جدیدترین کالاهای اجاره‌ای
تور عروس
این تور مناسب عروس میباشد
350000 تومان / روز
دستگاه پرس آجر
دستگاه صنعتی مناسب پرس آجر 10 گل
2500000 تومان / روز
تور ماهیگیری
مناسب برکه و استخر
50000 تومان / روز
مزایای استفاده
صرفه‌جویی در زمان
کاهش هزینه‌ها
آسودگی خاطر
سیستم احراز هویت
اطمینان از سلامت کالا
اجاره امن با تضمین دیجیتال