Root.dart

این کلاس در ابتدا با متغیر boolean به نام connected اتصال اینترنت را بررسی می‌کند. متغیر connected در ابتدا مقدار false دارد و فقط در تابع defaultWidget و در صورت متصل بودن به اینترنت، این متغیر true می‌شود. این تابع با باز کردن یک socket به آدرس google و بررسی پاسخ، متغیر connected را تغییر می‌دهد.

تابع defaultWidget در تابع initState صدا زده می‌شود.

در ادامه اگر متغیر connected، false باشد صفحه‌ای نمایش داده می‌شود که اعلام می‌کند «اینترنت وصل نیست» ولی اگر این متغیر true باشد به سراغ بررسی متغیر isLoggedIn می‌رود. اگر این متغیر true باشد به ویجت Home وگرنه به ویجت UserLoginPage منتقل می‌شود.

تابع isLogged بررسی می‌کند که اگر userToken از گذشته مانده باشد، تابع refreshUser را صدا می‌کند. اطلاعات این تابع پس از ارسال درخواست و دریافت پاسخ، اطلاعات کاربر را در sharedpreferences ذخیره می‌کند که عبارتست از فضایی کوچک برای ذخیره‌سازی اطلاعات کلید-مقدار[۱]. این اطلاعات با باز و بسته کردن برنامه و حتی پاک کردن حافظه موقت نهان[۲]، از بین نمی‌روند. البته برای افزایش سرعت برنامه با هربار باز کردن برنامه، این اطلاعات به حافظه موقت برنامه منتقل می‌شوند تا دسترسی راحت‌تر شود.

---

۱ Key-Value

۲ cache

Home.dart

این ویجت ساختار کلی ظاهر را دارد که دارای چهار BottomNavigationBar و یک Drawer است.

Profile.dart

تابع getProjects و تابع getThingProjects تعداد پروژه‌ها و اشیا داخل پروژه‌های کاربر را روی دکمه‌ها نمایش می‌دهد. تابع getThingsData نیز داده‌های دریافتی توسط اشیا کاربر را نمایش می‌دهد و تابع dataExtractor داده‌های JSON را به صورت درست نمایش می‌دهد.

dataConverter تاریخ میلادی را به جلالی تبدیل می‌کند. تابع timePicker نیز ساعت دریافتی از API را به صورت تمیز شده نمایش میدهد.

سایر توابع که از نوع Getter هستند، اطلاعات مربوط به کاربر را که بدو ورود دریافت و به صورت global در کل برنامه قابل دسترسی هستند را می‌یابند.

Projects.dart

برای render کردن لیست دریافتی از تابع getProjects، از ListView.builder فلاتر استفاده کردم. برای گزینه‌های امکانات، از DropdownButton استفاده کردم. برای دکمه + که پروژه اضافه می‌کند از FloatingActionButton استفاده کردم. تابع deleteProject نیز شناسه پروژه را می‌گیرد و آن را حذف و صفحه را ریفرش میکند. تابع displaySnackBar_ نیز در صورت موفقیت‌آمیز بودن حذف، یک اسنک بار نمایش میدهد و مینویسد *حذف با موفقیت انجام شد*. سایر گزینه‌های عملیاتی با MaterialPageRoute به صفحه مورد نظرشان منتقل می‌شوند.

Project_create.dart

تابع اصلی createProject است که داده‌های نوشته شده در projectNameController و projectDetailsController را به API پست می‌کند.

Project_manage.dart

تابع updateProject نام و توضیحات پروژه را برای API ارسال می‌کند.

Project_view.dart

برای پیاده‌سازی قسمت تاریخ‌گزین PersianDatePicker از کتابخانه shamsi_date استفاده کردم. تابع getThingsData نیز همانطور که قبلا اشاره شد داده های اشیا پروژه را می‌گیرد.

Thing.dart

برای نمایش سه تب جدا داخل این صفحه از TabBar استفاده کردم. که هر تب با TabBarView به ویجت مورد نیاز وصل می‌شود.

Thing_profile.dart

تابع deleteThingProfile با گرفتن شناسه پروفایل اشیا، به API درخواستی برای حذف آن می‌فرستد. تابع displaySnackBar_ نیز در صورت موفقیت‌آمیز بودن حذف، یک اسنک بار نمایش میدهد و مینویسد *حذف با موفقیت انجام شد*. تابع getThingProfiles نیز اطلاعات مربوط به پروفایل اشیا را می‌گیرد.

Thing_profile_create.dart

تابع اصلی createThingProfile است که داده‌های نوشته شده در Controller ها را به API پست می‌کند.

Thing_view.dart

getThingProfileDetails جزئیات اطلاعات مربوط به پروفایل اشیا را می‌گیرد.

Gateway.dart

تابع getGateways لیست گذرگاه‌ها را از API می‌گیرد.

Cart.dart

برای نمایش دو تب جدا داخل این صفحه از TabBar استفاده کردم. که هر تب با TabBarView به ویجت مورد نیاز وصل می‌شود.

Current_package.dart

تابع getCurrentPackage، بسته‌ی فعلی کاربر را از API می‌گیرد و زمان باقیمانده بسته را با تابع packageRemainTime حساب می‌کند.

Buy_package.dart

تابع getPackages لیست بسته‌های موجود برای خرید کاربر را دریافت می‌کند.