****



**بهمن 1400**

**تهیه‌کنندگان: تیم فنی مرکز نوآوری**

**تاییدکننده: ساناز وحیدی**

**مستند تولید سانیار**

فهرست مطالب

[نسخ 2](#_Toc96176566)

[چکیده 3](#_Toc96176567)

[هدف 3](#_Toc96176568)

[مقدمه 4](#_Toc96176569)

[server 5](#_Toc96176570)

[بخش اول (cms) 8](#_Toc96176571)

[بخش دوم(sunyar) 10](#_Toc96176572)

[بخش سوم (config) 12](#_Toc96176573)

[بخش چهارم (utility) 13](#_Toc96176574)

[مخزن گیت 13](#_Toc96176575)

[کتابخانههای نصب شده 13](#_Toc96176576)

[مشخصات زیرساخت 14](#_Toc96176577)

[Client 15](#_Toc96176578)

[local storage 25](#_Toc96176579)

[Interceptor 27](#_Toc96176580)

[کتابخانههای نصب شده 28](#_Toc96176581)

# نسخ

|  |  |
| --- | --- |
| نسخه | تغییرات |
| V1.0.0 | - |
| V1.1.0 | 1- Add login and register for admin & portal client  2- Add authorization function for web services (midleware) and implament it on admin & portal client  3- Resolve bugs in client & server  4- Add user & role section to web-admin client  5- Change component of plans ui in portal client  6- Add logout btn and name for portal client  7- add sql injection for cms subsystem |

# چکیده

سانیار، یک بستر کمک مالی است که با فناوری دفترکل توزیع شده پشتیبانی می شود. با بهره گیری از این فناوری، فرایند کمک در قالب طرح های خیرخواهانه سازمان های مردم نهاد در دسترس نیکوکاران قرار می گیرد. هر فردی که قصد کمک دارد، از پنجره واحد سانیار وارد پرتال های سازمان های مردم نهاد شده و این موارد را انتخاب و مشاهده خواهد کرد. به این ترتیب امکان کمک مهیا می شود و مؤسسات مردم نهاد خود عملیات تسویه را انجام می دهند. سانیار سوابق کمک های مالی را به صورت دائمی و شفاف ثبت و نگهداری می کند.

# هدف

پروژه سانیار در راستای تحقق اهداف زیر گام برمی دارد:

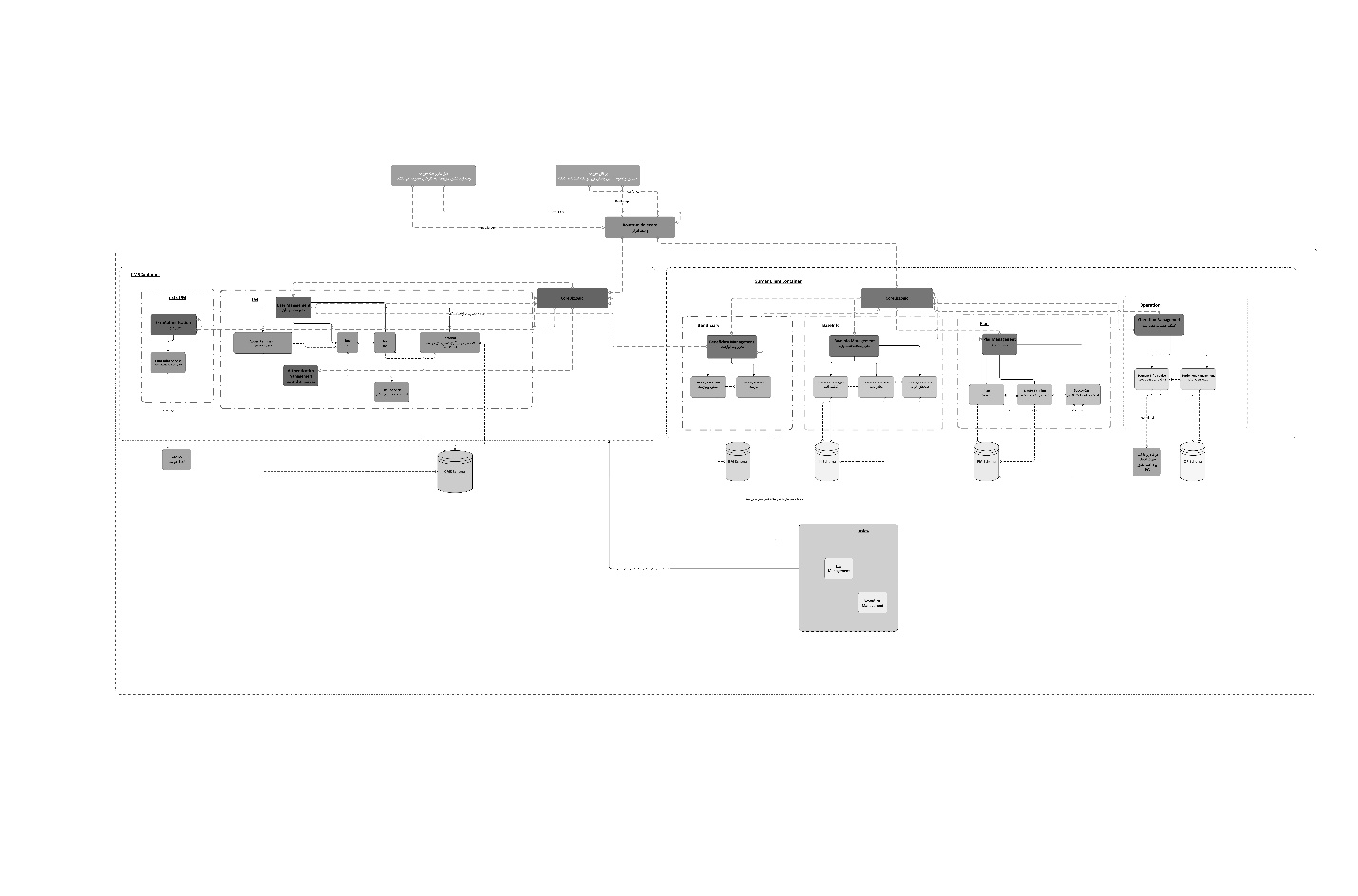
* افزایش شفافیت و جلب اعتماد مردم نسبت به مؤسسات مردم نهاد
* افزایش فعالیت و کمک های انسان دوستانه
* مدیریت هویت در مؤسسات مردم نهاد
* مدیریت توزیع کمک ها
* هماهنگی بین مؤسسات مردم نهاد و خیریه ها با پیاده سازی شبکه
* گسترش شبکه مدیریت هویت بین سازمان ها
* اکوسیستم سازی و مدیریت کلان

# مقدمه

از آنجا که وجود دو سیستم کاملا مستقل در پیاده­سازی نرم افزار نشانگر معماری میکرو سرویس است. اما با توجه به مفروضات مطرح شده توسط تیم کسب و کار و معماری کلان ارائه شده (شکل-1) توسط طراح مبتنی بر اینکه زیرسیستم مدیریت محتوا (cms) در آینده قابلیت جداسازی از زیرسیستم سانیار را داشته باشد و از طرفی با درنظر گرفتن محدودیت خیریه­ها در نصب و راه­اندازی سیستم کلی، نمای کلی فولدربندی پروژه جهت برآورده ساختن موارد ذکر شده در زیر مطابق با شکل-2 ارائه گردیده است که در ادامه به تشریح هر بخش به صورت مجزا می پردازیم.

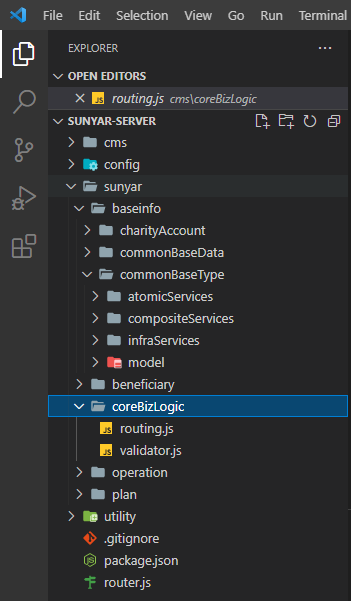
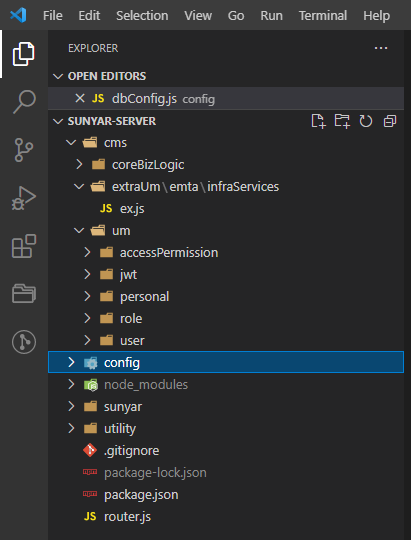
* پیاده سازی به صورت Monolithic
* جابجایی در حد دو سیستم مستقل
* روش توسعه نرم­افزار DDD(Domain Driven Design)
* طراحی ماژولار (مطابق با شکل-1)
* معماری سرویس گرا (مطابق با مستندات پیاده­سازی ارائه شده)

# server

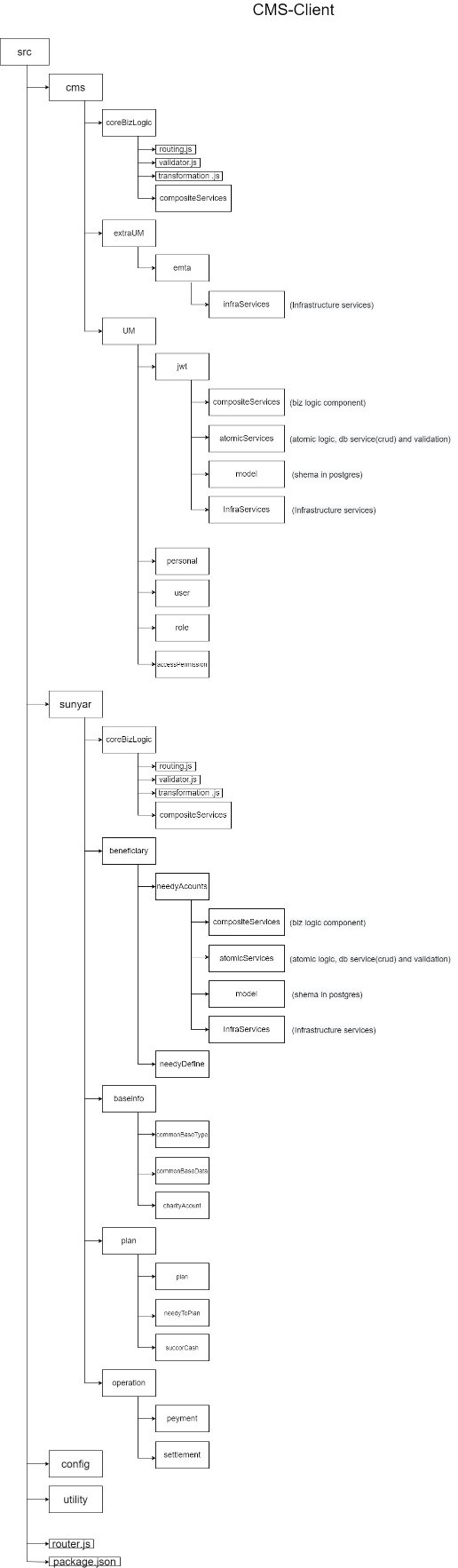


شکل-1-معماری کلان

ملاحظه: همانطور که در شکل 2 نشان داده شده است علاوه بر پوشه cms، sunyar، config، utility، فایل router.js و فایل package.json که به منزله شناسنامه پروژه می باشد به منظور ایجاد یک سرور و هدایت تمامی ریکوست های ارسالی به سمت سرور به گیت وی هر زیرسیستم بر اساس مسیر مشخص شده در ریکوست، هر دو فایل در root اصلی پروژه قرار گرفته اند.



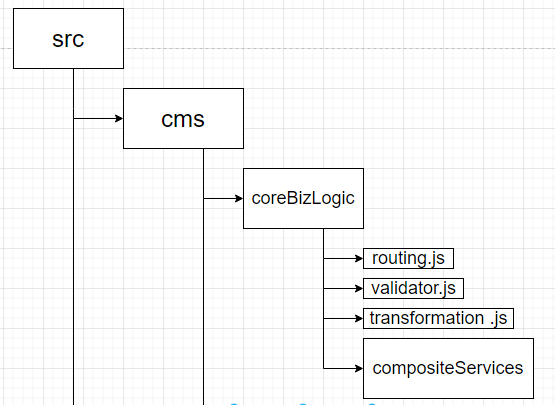
شکل-2-ساختار فولدربندی پروژه در محیط کد – sunyar و cms

****

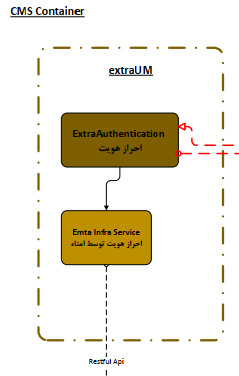
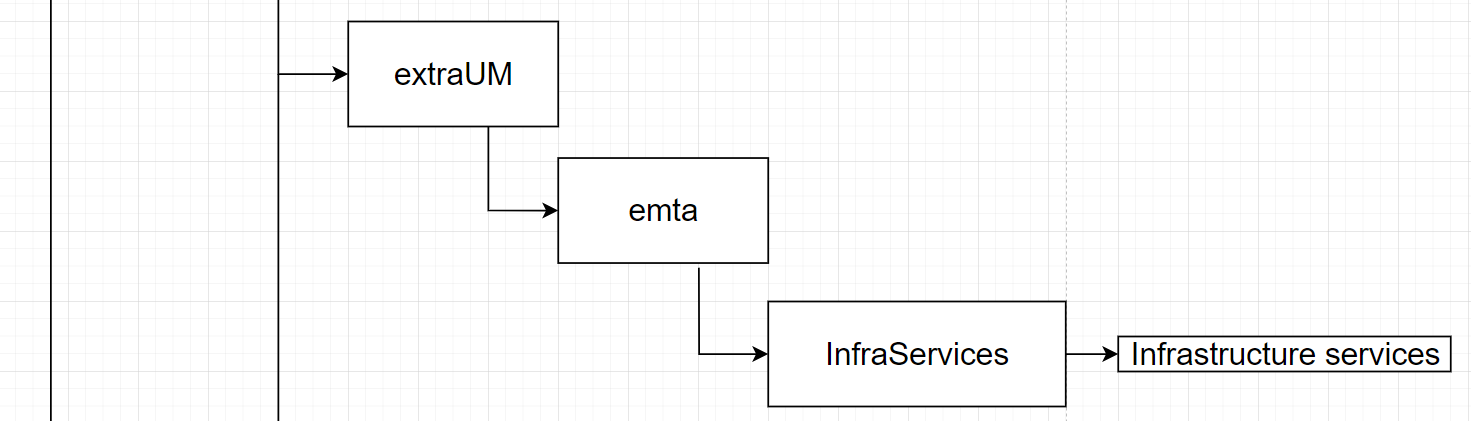
شکل-3-نمای کلی ساختار فولدربندی

# بخش اول (cms)

پوشه دامین coreBizLogic از آنجایی که بیزینس خاص خود را دارد نحوه پوشه­بندی داخل آن با بقیه دامین ها متفاوت می باشد. پوشه مذکور دربردارنده فایل routing.js مربوط به زیرسیستم cms است، درون این فایل عملیات authentication و authorization مربوط به پرسنل داخلی خیریه­ها به همراه اعتبارسنجی مقادیر ورودی در سطح اول انجام می­گیرد. پوشه compositeServices وظیفه صدا زدن سرویس­های بیزینسی موجود در باقی دامین­ها را دارد.

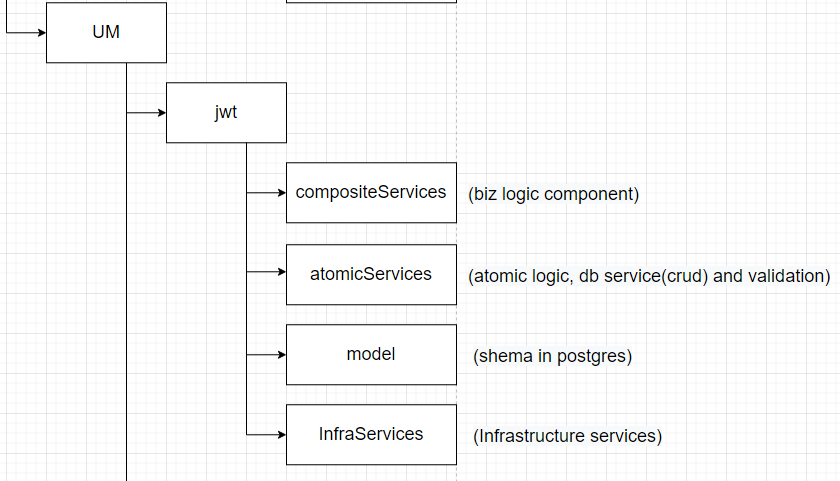


شکل-4-فولدر coreBizLogic زیرسیستم cms

دامین extraUM به منظور احراز هویت بیرونی سیستم تحت پروتکل Oauth توسط سامانه امتا جهت ورود خیرین و در آینده نیازمندان به سیستم درنظر گرفته شده است.

شکل-5-دامین extraUM

در دامین UM نیز کامپوننت jwt به منظور احراز هویت پرسنل خیریه­ها بر اساس jwt قرار داده شده است.

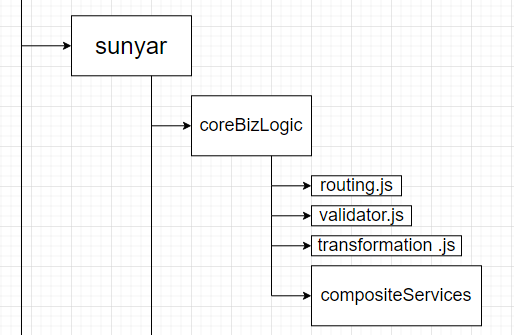


شکل-6-دامین UM

ایجاد دسترسی به سرویس های هر بخش مستلزم دانستن نقش های فرد لاگین شده یا ماهیت او می باشد. در صورتی که فرد ادمین باشد به تمامی سرویس ها دسترسی دارد، اگر نقش مددکار باشد به سرویس های تعریف نیازمند و تعریف طرح دسترسی دارد، اگر نقش حسابدار باشد به تسویه ها و همچنین اگر ماهیت فرد خیر باشد می تواند طرح هایی که در بخش پورتال نیاز به لاگین دارند دسترسی داشته باشد. عملکرد به این صورت است که توکن jwt که حاوی آیدی کاربر و نام کاربری است، دیکُد می شود و اطلاعات نقش ها و ماهیت آن فرد در صورت وجود در دیتابیس برای سرویس های دیگر سرور فرستاده می شود. دسترسی ها می توانند یک آرایه باشند به عبارت دیگر یک سرویس می تواند چندین دسترسی مختلف برای آن تعریف کرد. تمام اطلاعات نقش ها و ماهیت افراد در صورت وجود به پارامتر req.context.auth اضافه می شوند که شامل فیلد های roles و personType و user می باشد که استخراج این اطلاعات بر عهده تابع authenticateByJwt (میدلور اصلی) است. تابع authorizeRequest نقش اصلی در تعیین دسترسی ها را دارد به این صورت که به عنوان میدلور بر روی هر سرویس قرار می گیرد. ورودی آن اطلاعات فرد لاگین شده و همچنین دسترسی های مجاز برای سرویس است، در صورتی که اطلاعات فرد با این دسترسی های همخوانی نداشته باشد سرور پیغام خطای عدم دسترسی برای درخواست می فرستد.

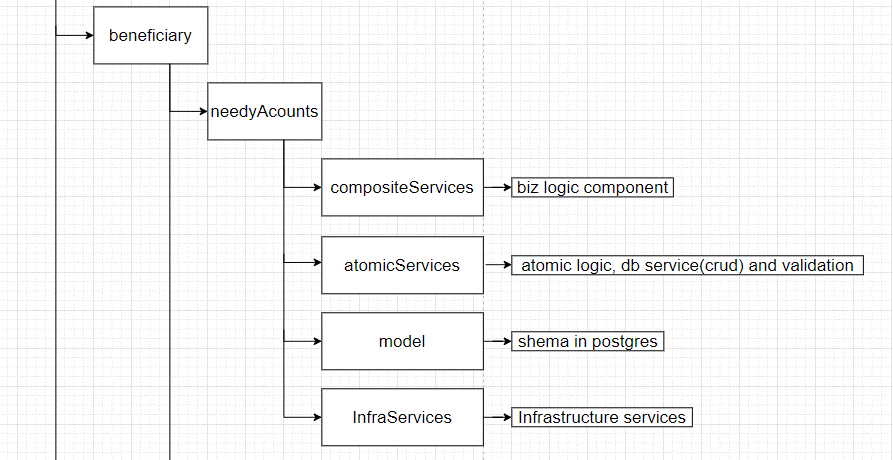
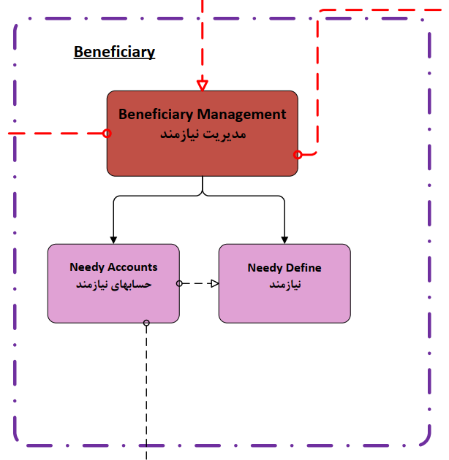
# بخش دوم(sunyar)

پوشه دامین coreBizLogic از آنجایی که بیزینس خاص خود را دارد نحوه پوشه بندی داخل آن با بقیه دامین ها متفاوت می باشد. پوشه مذکور دربردارنده فایل routing.js مربوط به زیرسیستم سانیار است، درون این فایل اعتبارسنجی مقادیر ورودی در سطح اول انجام می­گیرد و همانند زیرسیستم cms دارای پوشه compositeServices می­باشد.



شکل-7- coreBizLogic سانیار

با توجه معماری کلان پروژه، داخل پوشه sunyar به ازای هر دامین یک پوشه و داخل آن به ازای هر کامپوننت یک پوشه مجزا در نظر گرفته شده است. لایه کامپوننت خود مطابق با شکل 2 به عنوان نمونه که مربوط به دامین beneficiary است شامل چهار بخش مجزا می شود.



شکل-8-دامین beneficiary

:compositeServices درون پوشه compositeServices تمامی فرآیندهای مربوط به بیزینس نوشته شده و از سرویسهای crud دیتابیس پیاده سازی شده در پوشه atomicServices استفاده می شود.

atomicServices این پوشه مربوط به کلیه سرویسهای پایه (crud دیتابیس) و همچنین دربرگیرنده ولیدیشن مربوط به بیزینس می باشد.

ملاحظه: سرویسهای پایه به هیچ وجه نباید درون compositeServices قرار داده شوند

:model در این پوشه متد تعریف جداول دیتابیسی مربوط به کامپوننت دربردارنده توسط ORM به همراه پارامترهای ورودی فراخوانی می شود.

:InfraServices تمامی سرویس های خارجی و زیرساختی اعم از چک سلامت یا بیدار بودن دیتابیس، بالا بودن سرور و ... در این پوشه قرار می گیرند. به طور مثال اگر در کامپوننت مشخص شده در شکل-2، قصد استفاده از سرویسی موجود در cms را داشته باشیم، جای فراخوانی آن درون این پوشه است. این پوشه هیچ ربطی به منطق کسب و کار ندارد.

دسته­بندی مابقی دامین ها و کامپوننت های موجود در زیرسیستم سانیار نیز همانند نمونه آورده شده در بالا می باشند.

# بخش سوم (config)

فایل های مربوط به پیکربندی دیتابیس به تفکیک Dev (هیچ دیتایی در آن درست نمی باشد) Stage (دیتای عملیاتی و واقعی اما بر روی سرور تست می باشد و صرفا جهت تمرین است) و Operation (همه چیز برای محیط عملیاتی درنظر گرفته می شود)

پیکربندی:Dev کلیه جداول دیتابیس با استفاده از ORM در حالت Drop & Create تعریف می­شوند ( به صورت سیستمی)

پیکربندی :Stageکلیه جداول دیتابیس با استفاده از ORM در حالت چک تعریف می­شوند (به صورت دستی)

پیکربندی :Operationکلیه جداول دیتابیس با استفاده از ORM در حالت چک تعریف می­شوند (به صورت دستی)

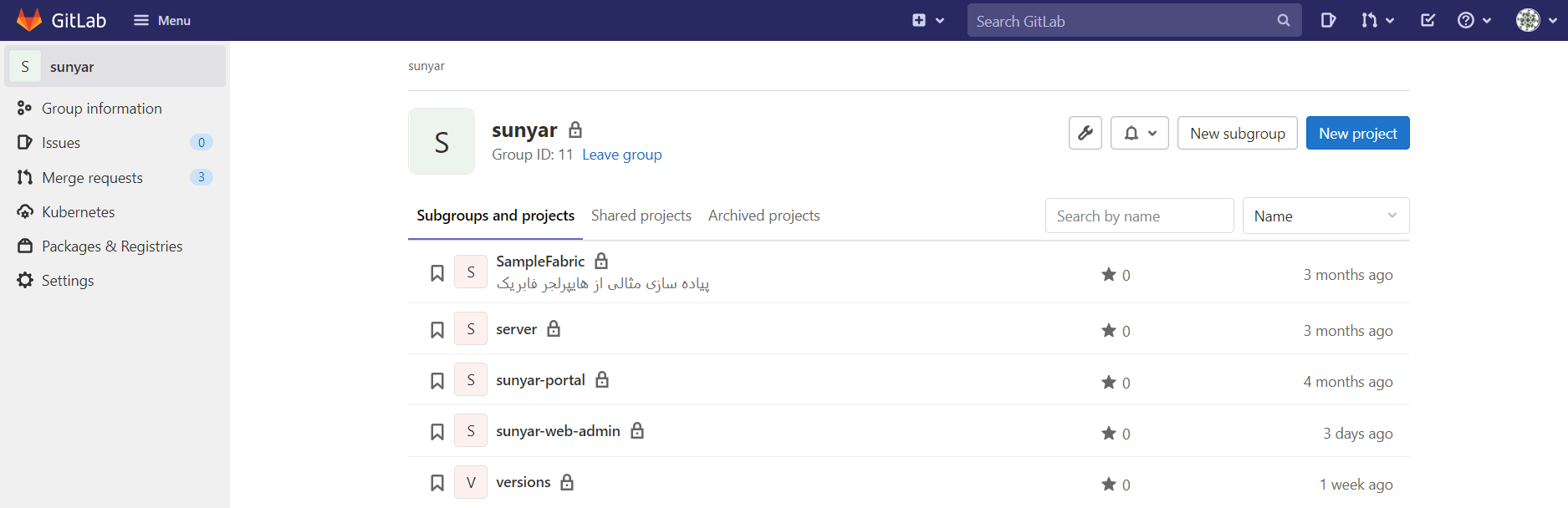
# بخش چهارم (utility)

کلیه فایل هایی که به صورت مشترک به هر دو زیرسیستم سانیار و cms سرویس می دهند و حالت جنرال دارند، مانند لاگ(پیاده سازی در فایل ضمیمه آورده شده است) و توابع پیاده سازی شده جهت انجام ولیدیشن در این پوشه قرار می گیرند.

# مخزن گیت

آدرس سرور مربوط به مخزن گیت سانیار به شرح زیر می باشد:

<https://192.168.9.71/sunyar>



شکل-9- ریپازیتوری گیت سانیار

که در آن یک گروه با نام sunyar و سه پروژه با نام­های زیر در داخل آن قرار گرفته است

* sunyar-backend
* sunyar-frontend
* sunyar-core

# کتابخانه­های نصب شده

"app-root-path": "^3.0.0",

"cookie-parser": "^1.4.5",

"crypto-js": "^4.1.1",

"express": "^4.17.1",

"joi": "^17.4.1",

"morgan": "^1.10.0",

"nodemon": "^2.0.7",

"pg": "^8.7.1",

"sequelize": "^6.6.5",

"uuid": "^8.3.2",

"winston": "^3.3.3"

# مشخصات زیرساخت

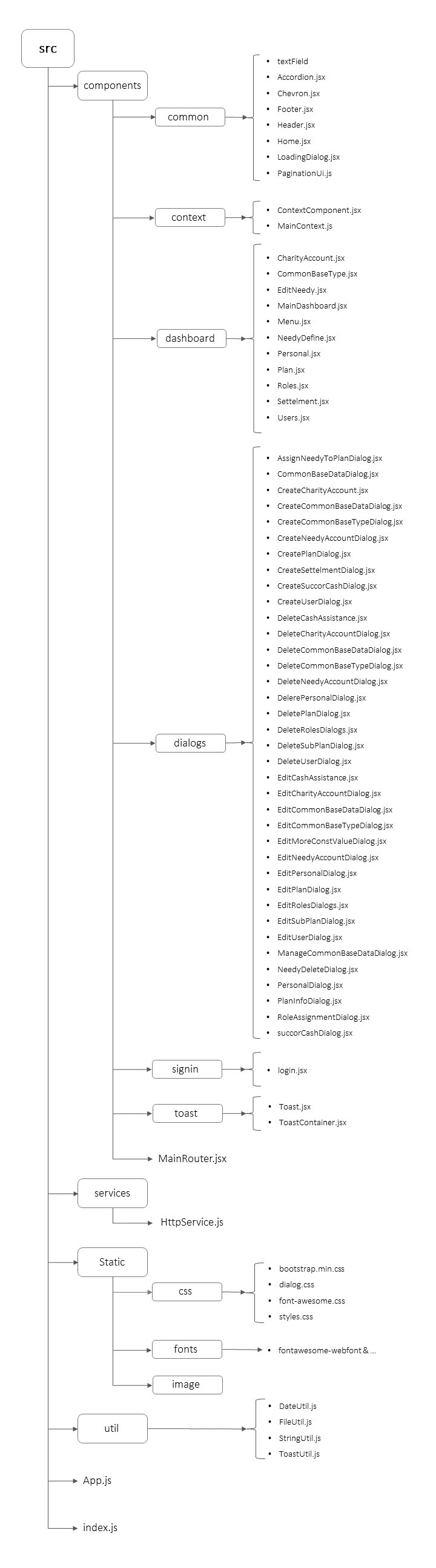
سرور تست: لینوکسی(CentOS)

* Storage: 60 g
* CPU: 3 core
* RAM: 2 g

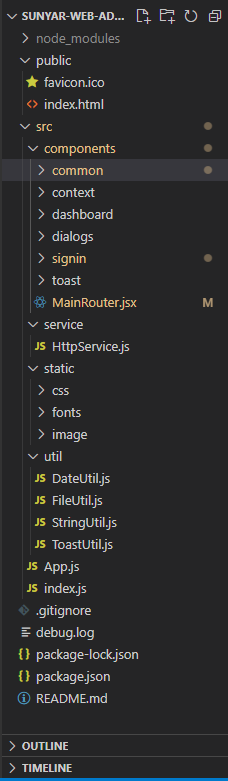
سرور دیتابیس: ویندوزی

* Storage: 30 g
* CPU: 5 core
* RAM: 2 g

# Client



شکل-1-ساختار فولدربندی ادمین پروژه sunyar



شکل-2-ساختار فولدربندی ادمین پروژه sunyar در محیط کد

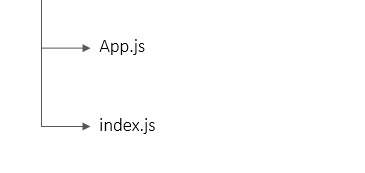
ساختاربندی کلی در ادمین سانیار به این گونه است که تمامی کامپوننت ها در دو فولدر کلی با نام‌های public و src قرار گرفته است.

فولدر public شامل آیکون سانیار و یک فایل html با نام index است که کل پروژه نهایتا در این فایل و در یک div با Id به نام root رندر می‌شود.

سایر کامپوننت ها همگی در فولدر src تعریف شده‌اند.

فولدر src شامل 4 فولدر components، service، static و util و دو فایل js به نام‌های App و index است.

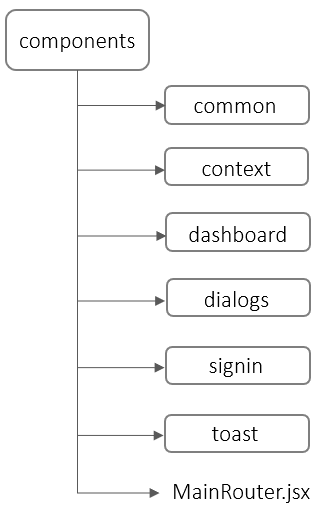
فایل App.js به عنوان parent تمامی کامپوننت ها است و تمامی فایل‌های css در این کامپوننت والد import شده است تا در تمامی کامپوننت‌های فرزند قابل دسترسی باشد.



شکل-3-ساختار فولدربندی ادمین پروژه sunyar \_برشی از src

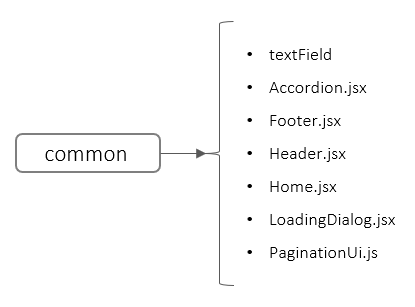
در App.js یک فانکشن به نام App تعریف شده و کامپوننت MainRouter را return می‌کند که نقش تعیین rout ها را بر عهده دارد و یک فایل index.js که وظیفه رندر کردن App را دارد.

فولدر components همانطور که در شکل 4 نمایش داده شده است، شامل شش فولدر و یک فایل mainRouter.jsx است که در این فایل rout های هر کامپوننت مشخص شده است.



شکل-4-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت components

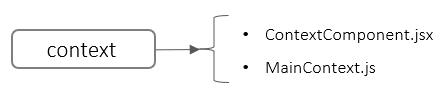
فولدر اول common شامل کامپوننت های معمول استفاده شده در پروژه است که زیر شاخه های آن در شکل 5 نمایش داده شده است.



شکل-5-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت common

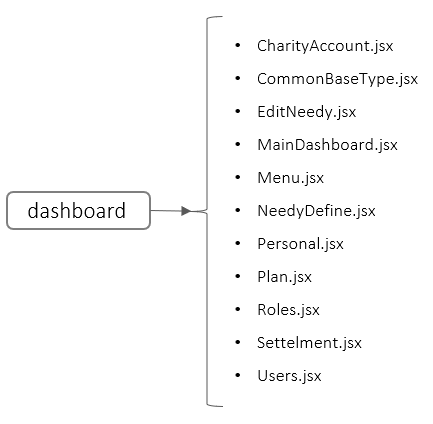
کامپوننت‌هایی مانند Header، Footer، Home و LoadingDialog که در تمامی routها قابل مشاهده است. فولدر textField جهت داینامیک کردن قسمت Login و Register به پروژه اضافه شده و paginationUi نیز در جداولی که سمت کلاینت پیجینیشن آنها انجام شده کنترل شده مورد استفاده قرار گرفته است. Accordion کامپوننتی است که در منو سایت ادمین و در ساختار درختی طرح در قسمت تعریف طرح و تسویه مورد استفاده قرار گرفته است.

فولدر context که در شکل 6 قابل مشاهده است، شامل دو فایل ContextComponent و MainContext است که برای آن دسته از stateهایی استفاده شده که در تمامی کامپوننت های پروژه به آن نیاز داشتیم و به منزله ظرفی برای اینگونه stateها است، مانند LoadinDialog که در تمامی ریکوست ها مورد استفاده قرار گرفته است.



شکل-6-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت context

فولدر dashboard شامل تمامی navLink های موجود در منو سایت ادمین است و تمامی کامپوننت های موجود در منو همانطور که در شکل 7 قابل روئیت است، در این فولدر تعریف شده است.



شکل-7-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت dashboard

CharityAccount کامپوننتی است که در آن میتوان یک حساب خیریه با داشتن اطلاعاتی چون نام بانک، نام شعبه، صاحب حساب، شماره کارت، نام حساب و شماره حساب تعریف کرد و لیستی از حساب های خیریه‌ای که تا کنون تعریف شده است را مشاهده کرد.

CommonBaseType کامپوننتی است که در آن مقادیر ثابت تعریف می‌شود، به طور مثال برای هر بانک یک مشخصه تعریف میشود و آن بانک از طریق کد یا مشخصه قابل شناسایی است.

EditNeedy برای تغییر اطلاعات نیازمند تعریف شده است. هر وقت نیاز بود اطلاعات نیازمند از قبیل نام، کد ملی، شماره شناسنامه و ... تغییر کند، در این قسمت می‌توان این کار را انجام داد.

MainDashboard کامپوننتی است که تمامی rout های navlink ها در آن تعریف شده است.

در کامپوننت Menu تمام موارد موجود در منوی سایت از قبیل اطلاعات پایه، مدیریت کاربران، مدیریت نیازمند، مدیریت طرح ها و کمک ها و مدیریت حسابداری مشخص شده است.

در کامپوننت NeedyDefine می‌توان نیازمند را با اطلاعات مربوطه تعریف کرد و لیستی از نیازمندانی که تا به حال تعریف شده است را مشاهده کرد و یا اقدام به تغییر اطلاعات و یا حذف یک نیازمند کرد.

Personal برای مشاهده لیستی از پرسنل تعریف شده است که این پرسنل میتواند خیر و یا پرسنل سایت باشد و قابلیت حذف و تغییر و یا افزودن پرسنل جدید در این کامپوننت تعبیه شده است.

در کامپوننت Plan تمام طرح هایی که برای کمک تعریف شده به صورت درختی قابل مشاهده است و همچنین قابلیت افزودن و تغییر یا حذف یک شاخه و یا زیر شاخه در هر کدام از طرح ها در این کامپوننت تعریف شده است.

Roles کامپوننتی است که لیستی از نقش های تعریف شده در آن قابل مشاهده است. این نقش‌ها به صورت استاتیک تعریف شده و شامل سه نقش ادمین، مددکار و حسابدار است که با توجه به نقش تعریف شده دسترسی های خاص برای هر نقش در نظر گرفته شده است.

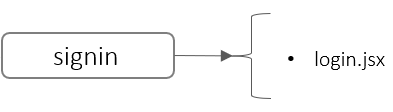
در Settelment تسویه و کمک های خیرین قابل مشاهده است. در این کامپوننت می‌توان از طریق تاریخ، نام نیازمند، نام طرح و یا نام خیر اطلاعات را واکشی کرد. تسویه را در همین کامپوننت با استفاده از دکمه افزودن می‌توان انجام داد ولی کمک های خیرین از طریق پرتال سایت قابل افزودن است و در این کامپوننت صرفا لیستی از کمک‌ها قابل مشاهده است.

Users نیز مربوط به کاربران سایت ادمین است و در آن قابلیت افزودن کاربر جدید، حذف و یا ویرایش در نظر گرفته شده است. همچنین در این کامپوننت می‌توان هر یک از سه نقش تعریف شده را به هر یک از کابران اختصاص و یا نقش آن‌ها را تغییر داد.



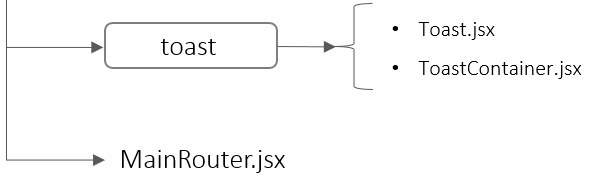
شکل-8-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت dialogs

در تمامی کامپوننت های منو برای هر یک از قابلیت های حذف، اضافه، ویرایش، تخصیص نقش، تخصیص نیازمند به طرح و ... برای نمایش از دیالوگ استفاده شده است. فولدر Dialogs شامل تمامی این دیالوگ های تعریف شده است و برای هر کدام یک کامپوننت در این فولدر در نظر گرفته شده است.



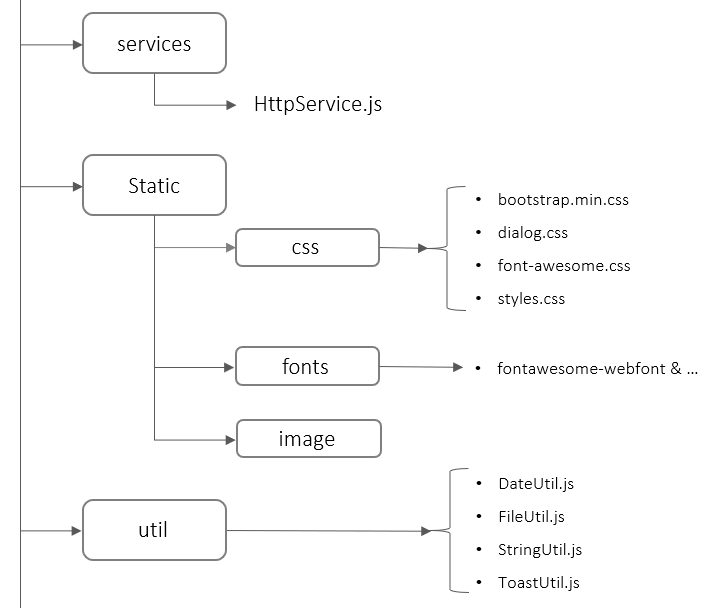
شکل-9-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت signin

Signin شامل کاموپوننت های مربوط به لاگین و رجیستر است که همانطور که در شکل 9 قابل مشاهده است در اینجا فقط کامپوننت login تعریف شده است.



شکل-10-ساختار فولدربندی ادمین پروژه sunyar \_کامپوننت toast و mainRouter

در toast کامپوننت های مربوط به هشدارها و خطاها تعریف شده است و استایل دهی و نمایش آن‌ها در این کامپوننت‌ها مشخص شده است و MainRouter نیز شامل rout های اصلی سایت است.



شکل-11-ساختار فولدربندی ادمین پروژه sunyar \_بخشی از src

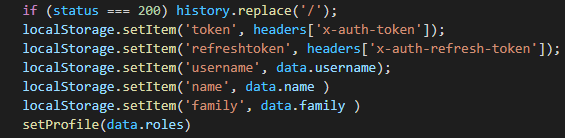
همانطور که در شکل 11 قابل مشاهده است در این بخش تمامی کامپوننت های مربوط به httpservice، css، font، image و Util ها وجود دارد که بر حسب لزوم در سایر کامپوننت ها از آن ها استفاده شده است.

# local storage

**local storage** این امکان را به ما می‌دهد که در مرورگر خود، داده‌هایی را با مدت زمان نامحدود ذخیره کنیم. **local storage**سرور را مشغول نمی‌کند، درنتیجه باعث کاهش سرعت نمی‌شود و فضای بیشتری را در اختیار ما قرار می‌دهد.

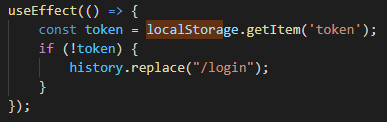
در پروژه سانیار در بخش هایی برای ذخیره داده ها در مرورگر از **local storage** استفاده شده است که به شرح زیر می‎باشد:

در قسمت ادمین سایت در کامپوننت Login برای ذخیره نام و نام خانوادگی، نام کاربری و احراز هویت از local storage استفاده شده است.



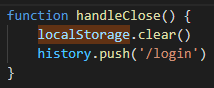
شکل-12-ذخیره اطلاعات در localstorage در کامپوننت Login

در کامپوننت MainRouter برای کنترل دسترسی کاربر، ابتدا چک می‌شود که کاربر لاگین کرده است یا خیر و در صورت لاگین نبودن، کاربر به صفحه لاگین ارجاع داده میشود.



شکل-13-ذخیره اطلاعات در localstorage در کامپوننت MainRouter

در کامپوننت Header یک دکمه خروج در نظر گرفته شده است که با کلیک کردن روی آن، کل اطلاعات local storageپاک می‌شود.



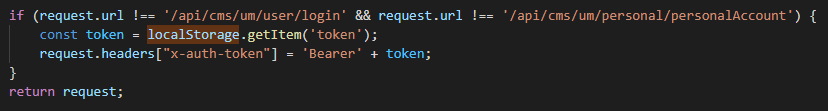
شکل-14-ذخیره اطلاعات در localstorage در کامپوننت Header

همچنین جهت نمایش نام و نام خانوادگی کاربری که لاگین کرده است، اطلاعات از local storage فراخوانی می‌شود.



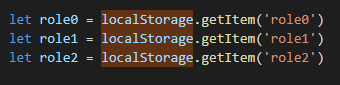
شکل-15-ذخیره اطلاعات در localstorage در کامپوننت Header

در Httpservice برای قرار دادن توکن هر کاربر بعد از لاگین درheader تمام ریکوست های بعدی از localstorage استفاده شده است.



شکل-16-ذخیره اطلاعات در localstorage در کامپوننت Httpservice

در کامپوننت menu برای داینامیک کردن منوها با توجه به دسترسی و نقش هر فرد از localstorage استفاده شده است. در اینجا 3 نقش ادمین، مددکار و حسابدار تعریف شده است که ادمین تمام قسمت های منو را می‌تواند ببیند و عدد 1 برای نقش ادمین در نظر گرفته شده است. مدد کار فقط قسمت های مربوط به مدیریت طرح ها و کمک ها و مدیریت نیازمند را می‌تواند مشاهده کند که برای آن عدد 2 و برای حسابدار که تنها به مدیریت حسابداری دسترسی دارد عدد 3 در نظر گرفته شده است.



شکل-17-ذخیره اطلاعات در localstorage در کامپوننت menu

# Interceptor

در httpservice و در interceptor ریکوست هایی که از کلاینت به سرور زده می‌شود مکانیزم token و refreshToken اعمال شده است که token هر فردی که لاگین میکند، در header تمامی ریکوست‌های بعد قرار داده شده است. هر توکن یک expire time دارد و وقتی که  expireشد دوباره از طریق ارسال refresh token یک token و refresh token جدید دریافت می‌کند تا وقتی که expire time رفرش توکن نیز تمام شود و کاربر به سمت صفحه لاگین هدایت شده و دوباره لاگین شود و مراحل احراز هویت طی شود.

# کتابخانه­های نصب شده

    "axios": "^0.21.1",  
    "bootstrap": "^5.0.0-beta3",  
    "jalali-moment": "^3.3.10",  
    "popper.js": "^1.16.1",  
    "react": "^17.0.1",  
    "react-dom": "^17.0.1",  
    "react-multi-date-picker": "3.3.0",  
    "react-router": "^5.2.0",  
    "react-router-dom": "^5.3.0",  
    "react-scripts": "4.0.3",  
    "react-table": "^6.8.2",  
    "web-vitals": "^1.1.1"