مسابقات ملی مهارت رشته تکنولوژی های وب

مسابقات کشوری ۱۴۰۲

ماژول سمت سرور

زمان: ۳ ساعت

معرفی

در این ماژول نیاز است که Backend سیستم مدیریت اشتراک فایل ابری را پیاده سازی کنید. در این بخش باید از فریمورک لاراول استفاده کنید و باید از دیتابیس mysql یا MariaDB استفاده گردد .

شما بایستی بخش احراز هویت سیستم جهت ایجاد کاربر و ورود به سیستم را پیاده سازی کرده و سپس بخش های مدیریت فایل‌ها را توسعه دهید .

۱- ثبت‌نام

یک API برای ثبت نام که ورودی های username و password میگیرد و کاربر را ثبت‌نام میکند.

متد این API از نوع POST است.

POST /api/login

Request body:

- username
- email
- password
- confirmPassword

Response 201:

```
{
    "message": "user registered"
}
```

برای همه ورودی ها باید ولیدیشن داشته باشید.
ورودی‌های username، password، confirmPassword و email اجباری هستند.
در صورتی که password و confirmPassword همخوانی نداشتن باید خطای 422 با مسیج درست نمایش داده شود.
پسورد باید حتما شامل حروف بزرگ، حروف کوچک، عدد و کاراکترهای خاص باشد در غیر این صورت خطای 422 با مسیج درست نمایش داده شود.
مقدار email باید تکراری نباشد و در صورت وارد شدن ایمیل تکراری، خطای 422 با مسیج درست نمایش داده شود.
در صورت نادرست بودن ولیدیشن، باید خطای 422 با مسیج درست نمایش داده شود.
باید جلوی race condition گرفته شود، اگر کسی پشت سر هم API ساخت دایرکتوری را فراخوانی کند، نباید به صورت داپلیکیت ساخته شود.

۲- ورود

یک API برای ورود که ورودی های email/username را به عنوان identity و password میگیرد و در صورت ورود موفق، یک توکن برمیگرداند.

در ورودی identity میتوان ایمیل یا username وارد شود.
توکن باید دارای مدت زمان ۲ دقیقه ای باشد و پس از آن منقضی شود.
متد این API از نوع POST است.
در صورت ورود ناموفق باید ارور مشخص برگرداند.
باید جلوی درخواست های مجدد لاگین (brude force) گرفته شود. در هر دقیقه فقط میتوان ۵ بار این api فراخوانی شود.

POST /api/login

Request body:

- identity
- password

Response 201:

```
{
        "token": "xxxxxxxx"
}
```

۳- ساخت دایرکتوری

یک API برای این که بشود دایرکتوری جدید درست کرد، که حتما نیاز به توکن کاربر برای ساخت است، در غیر این صورت، این API ارور authentication نمایش میدهد.

متد این API از نوع POST است.
ورودی این API مقدار directoryName است.
نام هر دایرکتوری باید یونیک باشد. در غیر این صورت ارور 422 با مسیج درست نمایش داده شود.
دایرکتوری ساخته شده باید برای همین کاربری باشد که لاگین کرده است.
مقدار status باید شامل حروف انگلیسی (با حرف کوچک) و اعداد باشد. در غیر این صورت ارور 422 با مسیج درست نمایش داده شود.
باید بشود دایکرتوری‌های تودرتو به صورت زیر ساخت:

dir1/dir2/dir3

که ورودی باید به این شکل باشد:

directoryName: dir1/dir2/dir3

در دیتابیس باید سازنده دایرکتوری مشخص باشد.
در دیتابیس باید تاریخ ساخت دایرکتوری مشخص باشد.
در خروجی به جز دیتای وارد شده باید آی‌دی تسک، آی‌دی سازنده تسک، تاریخ ساخت و تاریخ ویرایش نمایش دهد.

POST /api/directories

Request body:
- directoryName

Request header:
- Authorization: Token {USER_TOKEN}

Response 201:

```
{
        "data": {
                "id": "1",
                "name: "dir3",
                "type": "directory",
                "path": "dir1/dir2",
                "address": "dir1/dir2/dir3",
                "size": "0",
                "createdAt": "2023-08-06 10:11:12",
                "updatedAt": "2023-08-06 10:11:12"
        }
}
```

۴ - آپلود فایل

یک API برای این که بشود کاربر یک فایل آپلود کند، که حتما نیاز به توکن کاربر میباشد، در غیر این صورت، این API ارور نمایش میدهد.

متد این API از نوع POST است.
نام فیلد ورودی فایل، file میابشد.
کاربر میتواند چند فایل همزمان آپلود کند.
کاربر میتواند در دایرکتوری که میخواهد فایل خود را آپلود کند که باید مقدار ورودی directory را پر کند؛ در غیر این صورت، فایل در هیچ دایرکتوری و در روت پروژه قابل دسترس است.
در صورتی که عکس آپلود شود، حتما باید با فرمت های jpg، jpeg، png باشد. درغیر این صورت باید خطا به کاربر نمایش داده شود.
در ریسپانس باید نوع فایل نمایش داده شود.

- فایل عکس: image
- فایل ویدیویی: video
- فایل صوتی: audio
- فایل pdf: document
- فایل متنی: text

بعد از آپلود، آدرس فایل و حجم آن (به مگابایت) نمایش داده شود.

POST /api/upload

Response 201:

```
{
        "data": {
                "id": "2",
                "name: "my_music.mp3",
                "type": "audio",
                "path": "dir1/dir2",
                "address": "dir1/dir2/my_music.mp3",
                "size": "20mb",
                "createdAt": "2023-08-06 10:11:12",
                "updatedAt": "2023-08-06 10:11:12"
        }
}
```

۵- ویرایش نام فایل یا دایرکتوری

یک API برای این که بشود نام فایل یا دایرکتوری را ویرایش کرد.

متد این API از نوع PATCH است.
در صورتی که نام دایرکتوی تغییر کند، باید تمام فایل‌ها/دایرکتوری‌هایی که در این دایرکتوری هستند هم در جای درستی بمانند، برای مثال اگه در دایرکتوری dir1 فایل های a.mp3 و b.png قرار داشتند، در صورت تغییر نام دایرکتوری از dir1 به myfile، آدرس‌های این فایل‌ها از dir1/a.mp3 و dir1/b.png به myfile/a.mp3 و myfile/b.png تغییر کنند.
در خروجی باید تاریخ ویرایش بعه درستی نمایش دهد.

PATCH /api/files/{file_or_directory_id}/~rename

Request body:
- newName

Request header:
- Authorization: Token {USER_TOKEN}

Response 201:

```
{
        "data": {
                "id": "2",
                "name: "my_music2.mp3",
                "type": "audio",
                "path": "dir1/dir2",
                "address": "dir1/dir2/my_music.mp3",
                "size": "20mb",
                "createdAt": "2023-08-06 10:11:12",
                "updatedAt": "2023-08-06 10:11:12"
        }
}
```

۶- حذف فایل یا دایرکتوری

یک API برای این که بشود فایل یا دایرکتوری را حذف کرد، که حتما نیاز به توکن کاربر و آی‌دی تسک میباشد، در غیر این صورت، این API ارور نمایش میدهد.

متد این API از نوع DELETE است.
توجه کنید، در صورت پاک کردن یک دایرکتوری، تمام زیر مجموعه‌های آن، اعم از فایل و دایرکتوری باید حذف شوند.

DELETE /api/files/{file_or_directory_id}

Request header:

- Authorization: Token {USER_TOKEN}

Response 200:

```
{
    "message": "object deleted"
}
```

۷- لیست فایل‌ها/دایرکتوری‌ها

یک API برای این که بشود لیست فایل‌ها به همراه دایرکتوری‌های موجود در هر دایرکتوری را دریافت کرد، که حتما نیاز به توکن کاربر میباشد، در غیر این صورت، این API ارور نمایش میدهد.

متد این API از نوع GET است.
توجه کنید، هر کاربر فقط باید فایل‌های آپلود شده یا دایرکتوری‌های ساخته شده‌ی خودش را ببیند.
هر کاربر باید فایل‌ها یا دایرکتوری‌هایی که کاربری دیگر به آن دسترسی داده است را ببیند.
این API قابلیت سرچ با نام فایل/دایرکتوری را باید داشته باشد که به صورت query param باید مورد استفاده قرار گیرد:

GET /api/files?q=xxx

این API قابلیت sort کردن بر اساس تاریخ ساخت و حجم باید داشته باشد؛ sort باید قابلیت از آخر به اول و از اول به آخر داشته باشد:

GET /api/files?sortBy=createdAt(1) سورت به صورت آخر به اول
GET /api/files?sortBy=createdAt(-1) سورت به صورت اول به آخر
GET /api/files?sortBy=size(1) سورت به صورت آخر به اول
GET /api/files?sortBy=size(-1) سورت به صورت اول به آخر

باید قابلیت pagination در این api وجود داشته باشد، مقدار محدودیت تعداد فایل در هر صفحه با perPage و گرفتن لیست هر صفحه با page به صورت query param انجام شود.

GET /api/files?page=1&perPage=10

GET /api/files

Request header:
- Authorization: Token {USER_TOKEN}

Response 200:


```
{
    "data": [
        {
            "id": "1",
            "name: "dir3",
            "type": "directory",
            "path": "dir1/dir2",
            "address": "dir1/dir2/dir3",
            "size": "0",
            "createdAt": "2023-08-06 10:11:12",
            "updatedAt": "2023-08-06 10:11:12"
        },
        {
            "id": "2",
            "name: "my_music.mp3",
            "type": "audio",
            "path": "dir1/dir2",
            "address": "dir1/dir2/my_music.mp3",
            "size": "20mb",
            "createdAt": "2023-08-06 10:11:12",
            "updatedAt": "2023-08-06 10:11:12"
        }
```

```
        ]
}
```

۸- دانلود کردن فایل

دانلود کردن همه فایل ها برای همه کاربران و مهمانان آزاد است، اما اگر در موقع دانلود، توکن کاربر وارد نشود، به منظور کاربر مهمان تلقی میشود و **سرعت دانلود** برای این کاربر باید بر روی **50kb** در ثانیه **محدود** شود.

متد این API از نوع GET است.
آدرس فایل ها به این صورت است که، اگر فایل music.mp3 در دایرکتوری dir1 قرار داشته باشد، لینک دانلود به شرح زیر است:

GET /download/dir1/music.mp3

در صورتی که آدرس داده شده برای یک دارکتوری باشد، باید تمام فایل‌ها و دایرکتوری‌های زیرمجموعه‌ی این دایرکتوری **زیپ** شوند و برای کاربر قابل دانلود باشد.

GET /download/{file_path}

Request header:

- Authorization: Token {USER_TOKEN}

۹- دسترسی یک فایل یا دایرکتوری به یک کاربر

یک API برای این که بشود به هر کاربری که میخواهیم دسترسی به فایل‌ها/دایرکتوری‌های انتخابی خودمان بدهیم، که حتما نیاز به توکن کاربر میباشد، در غیر این صورت، این API ارور نمایش میدهد.

متد این API از نوع POST است.
برای دسترسی دادن فایل مشخص شده به کاربری که میخواهیم، باید ایمیل آن کاربر را در ورودی وارد کنیم. در غیر این صورت باید خطای ۴۲۲ نمایش دهیم.
اگر کاربر در سیستم وجود نداشت باید خطای ۴۲۲ با عنوان مناسب نمایش دهیم.
در صورتی که دسترسی یک دایکتوری را به آن کاربر میدهیم، علاوه بر خود دایکتوری، باید بر فایل‌ها یا دایرکتوری‌های زیرمجموعه آن هم دسترسی داده شود.
بعد از دسترسی دادن به کاربر، باید فایل‌ها/دایرکتوری‌های دسترسی داده شده در لیست فایل‌ها/دایرکتوری‌های کاربر قابل رویت باشند.

POST /api/files/{file_or_directory_id}/~access-to-user

Request body:
- userEmail

Request header:
- Authorization: Token {USER_TOKEN}

Response 200:
{
"message": "set access to files or directories"
}

پیام خطا

خطای احرازهویت:

Response 401:

```
{
        "message": "unauthorized"
}
```

خطای ولیدیشن:

Response 422:

```
{
        "message": "validation error",
        "errors": [
                "field x most be required",
                "field y most be numeric",
                "field z should be one of '1', '2'",
        ]
}
```

خطای پیدا نکردن:

Response 404:

```
{
        "message": "task/story not found"
}
```

ساخت دیتای اولیه

باید یک کاربر با مشخصات زیر به صورت پیشفرض در سیستم وجود داشته باشد:

- username: admin
- password: 1234
- email: admin@iranwsc.ir

برای اجرای این قابلیت از کامند زیر باید استفاده شود.

```
php artisan db:seed
```

تمامی ورودی‌ها (به جز آپلود) و تمام خروجی‌ها (به جز دانلود) به شکل JSON هستند.

توجه داسته باشید کدهای خروجی باید برابر با خروجی هر API باشد. در غیر این صورت نمره مربوطه گرفته نمیشود.

حتما باید از قابلیت های لاراول در این پروژه استفاده شود.

برای دیزاین دیتابیس باید از migration استفاده کنید.

در صورتی که apiهای خواسته شده، طبق دستورالمل فراخوانی نشوند، نمره آن بخش را نخواهید گرفت.

برای اجرای api ها میتوانید از postman استفاده کنید.

در پایان باید از postman خروجی JSON گرفته باشید. خروجی فایل را با اسم postman.json در کنار فایل های پروژه قرار دهید.

در نهایت فایل نهایی خود را در پوشه Module_ServerSide_XX که XX شماره سیستم شما میباشد در سرور قرار دهید .