15-3 分享相关 API

创建分享

- 创建数据迁移表

```bash
npx sequelize migration:generate --name=share
```

- 定义 database / migrations / 目录下生成的数据迁移文件

JavaScript | 复制代码

```javascript
"use strict";

/** @type {import('sequelize-cli').Migration} */
module.exports = {
  async up(queryInterface, Sequelize) {
    const { INTEGER, STRING, DATE, ENUM, TEXT } = Sequelize;
    return queryInterface.createTable("share", {
      id: {
        type: INTEGER(20),
        primaryKey: true,
        autoIncrement: true,
      },
      sharedurl: {
        type: STRING,
        allowNull: true,
        defaultValue: "",
        comment: "分享链接",
      },
      file_id: {
        type: INTEGER,
        allowNull: false,
        defaultValue: 0,
        comment: "文件id",
        references: {
          model: "file",
          key: "id",
        },
        onDelete: "cascade",
        onUpdate: "restrict", // 更新时操作
      },
      iscancel: {
        type: INTEGER(1),
        allowNull: false,
        defaultValue: 0,
        comment: "是否取消分享",
      },
      user_id: {
        type: INTEGER,
        allowNull: false,
        defaultValue: 0,
        comment: "用户id",
        references: {
          model: "user",
          key: "id",
        },
```

```
          onDelete: "cascade",
          onUpdate: "restrict", // 更新时操作
        },
        created_time: DATE,
        updated_time: DATE,
      });
    },

    async down(queryInterface, Sequelize) {
      return queryInterface.dropTable("share");
    },
  };
```

- 执行数据库变更

```
Bash | 复制代码
npx sequelize db:migrate
```

- model 包创建 share.js 数据模型

JavaScript | 复制代码

```javascript
"use strict";
const crypto = require("crypto");
module.exports = (app) => {
  const { STRING, INTEGER, DATE, ENUM, TEXT } = app.Sequelize;

  const Share = app.model.define("share", {
    id: {
      type: INTEGER(20),
      primaryKey: true,
      autoIncrement: true,
    },
    sharedurl: {
      type: STRING,
      allowNull: true,
      defaultValue: "",
      comment: "分享链接",
    },
    file_id: {
      type: INTEGER,
      allowNull: false,
      defaultValue: 0,
      comment: "文件id",
      references: {
        model: "file",
        key: "id",
      },
      onDelete: "cascade",
      onUpdate: "restrict", // 更新时操作
    },
    iscancel: {
      type: INTEGER(1),
      allowNull: false,
      defaultValue: 0,
      comment: "是否取消分享",
    },
    user_id: {
      type: INTEGER,
      allowNull: false,
      defaultValue: 0,
      comment: "用户id",
      references: {
        model: "user",
        key: "id",
      },
      onDelete: "cascade",
```

```
        onUpdate: "restrict", // 更新时操作
      },
      created_time: DATE,
      updated_time: DATE,
    });

    Share.associate = function (models) {
      // 关联文件
      Share.belongsTo(app.model.File);
    };

    return Share;
  };
```

- 控制器：app/controller/share.js

JavaScript | 复制代码

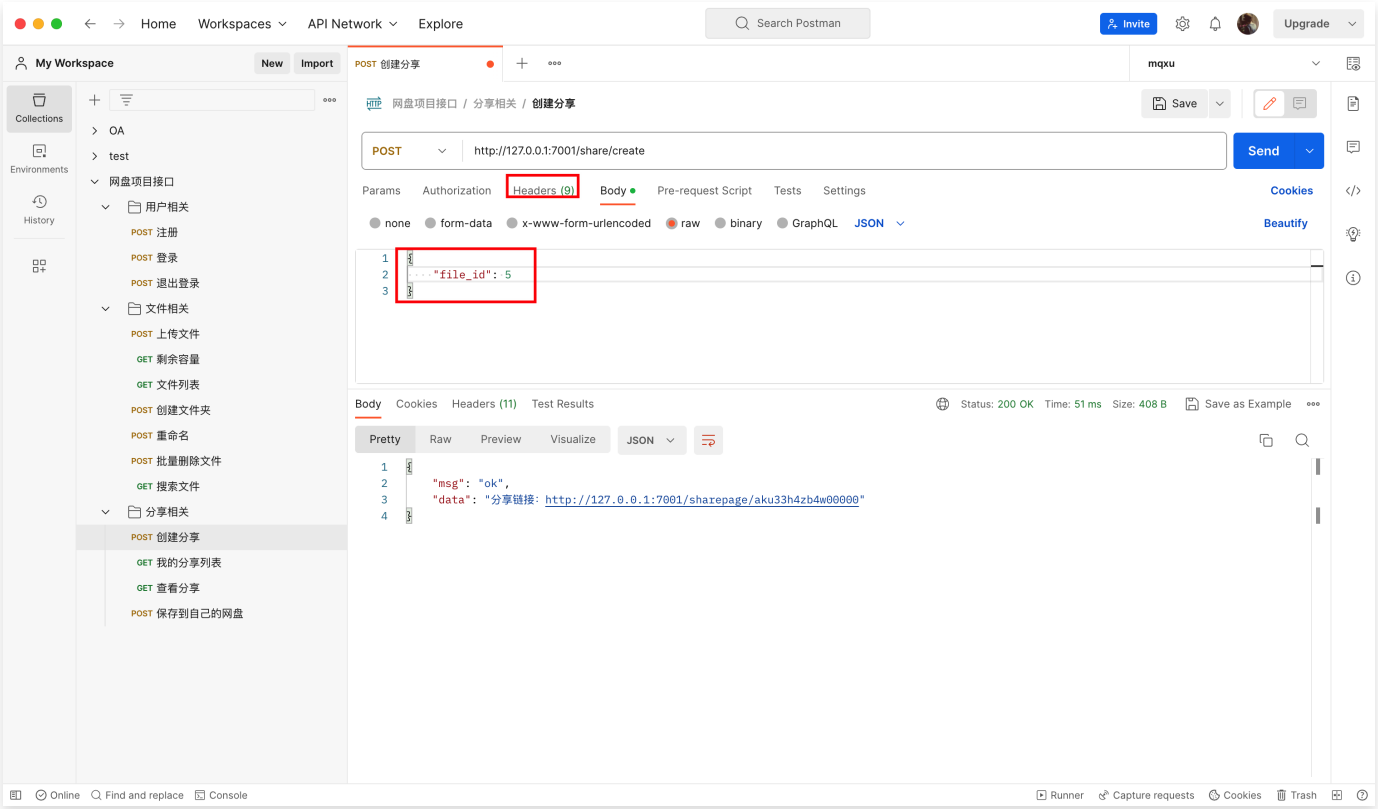
```javascript
"use strict";

const Controller = require("egg").Controller;

class ShareController extends Controller {
  // 创建分享
  async create() {
    const { ctx, app, service } = this;
    let user_id = ctx.authUser.id;

    ctx.validate({
      file_id: {
        type: "int",
        required: true,
        desc: "文件ID",
      },
    });

    let { file_id } = ctx.request.body;

    let f = await app.model.File.findOne({
      where: {
        id: file_id,
        user_id,
      },
    });

    if (!f) {
      return ctx.throw(404, "文件不存在");
    }

    let sharedurl = ctx.genID(15);

    let s = await app.model.Share.create({
      sharedurl,
      file_id,
      iscancel: 0,
      user_id,
    });

    let url = "http://127.0.0.1:7001/sharepage/" + sharedurl;
    ctx.apiSuccess("分享链接: " + url);
  }
}

```

```
46  module.exports = ShareController;
```

- 路由

```
JavaScript | 复制代码

1  // 创建分享
2  router.post("/share/create", controller.share.create);
```

测试，带 token 传需要分享的文件的主键




```
commit ba22dd0f0dedffd70c7ef26f10e0a8120e405898 (HEAD -> main)
Author: mqxu <moqi1977@gmail.com>
Date:   Tue Jul 4 23:48:41 2023 +0800

    创建分享
```

我的分享列表

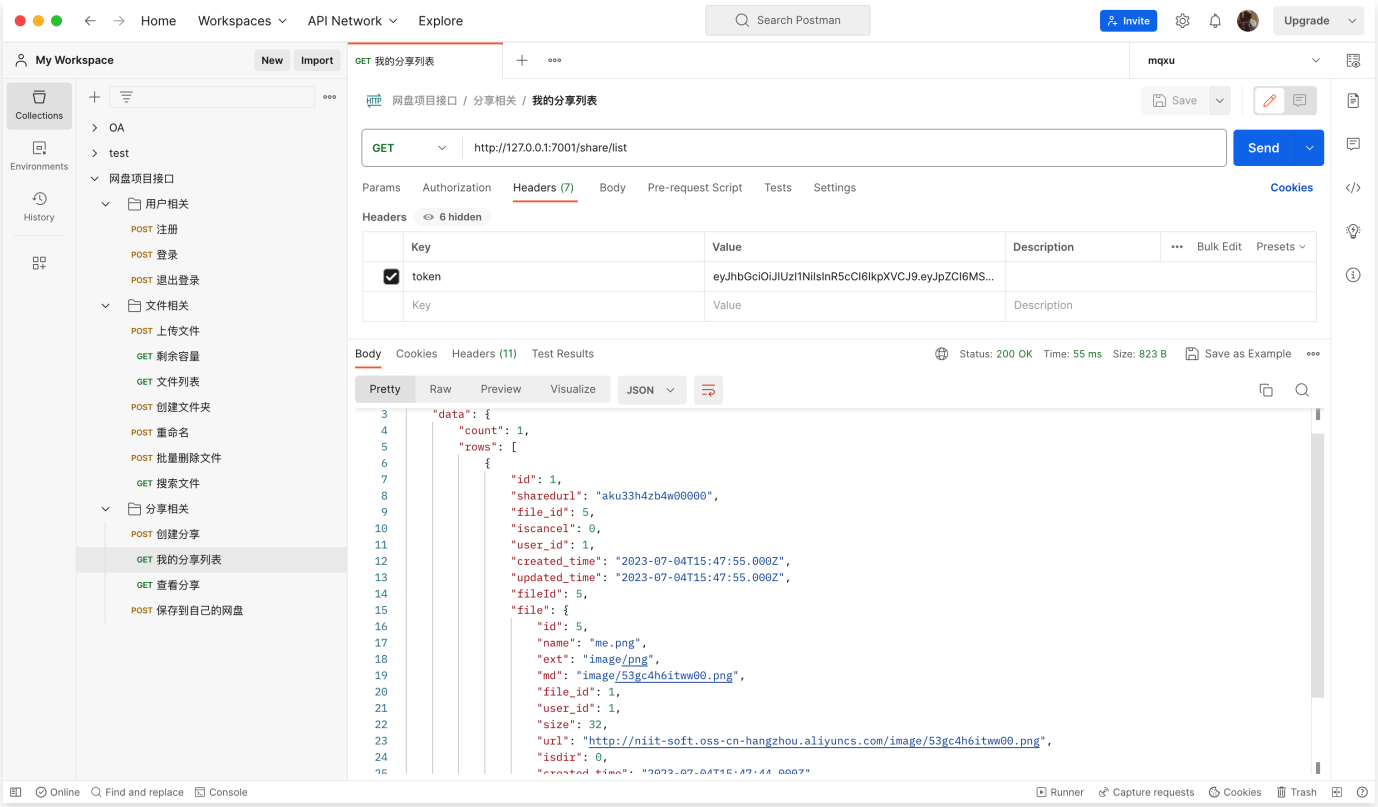
- 控制器：app/controller/share.js

```JavaScript
    // 我的分享列表
    async list() {
      const { ctx, app } = this
      const user_id = ctx.authUser.id

      let list = await app.model.Share.findAndCountAll({
        where: {
          user_id,
        },
        include: [
          {
            model: app.model.File,
          },
        ],
      })

      ctx.apiSuccess(list)
    }
```

- 路由：app/router.js

```JavaScript
// 我的分享列表
router.get('/share/list', controller.share.list);
```

- 测试带 token 请求，可以看到刚才分享过的文件




```
commit dabe25e7b6ebd6ab8f04466ef607b30e08ac1a04 (HEAD -> main)
Author: mqxu <moqi1977@gmail.com>
Date:   Tue Jul 4 23:52:00 2023 +0800

    我的分享列表
```

查看分享

- 控制器：app/controller/share.js

JavaScript | 复制代码

```javascript
// 查看分享
 async read() {
   const { ctx, app, service } = this
   let sharedurl = ctx.params.sharedurl
   if (!sharedurl) {
     return ctx.apiFail('非法参数')
   }

   let file_id = ctx.query.file_id

   // 分享是否存在
   let s = await service.share.isExist(sharedurl)

   let where = {
     user_id: s.user_id,
   }

   if (!file_id) {
     where.id = s.file_id
   } else {
     where.file_id = file_id
   }

   let rows = await app.model.File.findAll({
     where,
     order: [['isdir', 'desc']],
   })

   ctx.apiSuccess(rows)
 }
```

- 服务：app/service/share.js

JavaScript | 复制代码

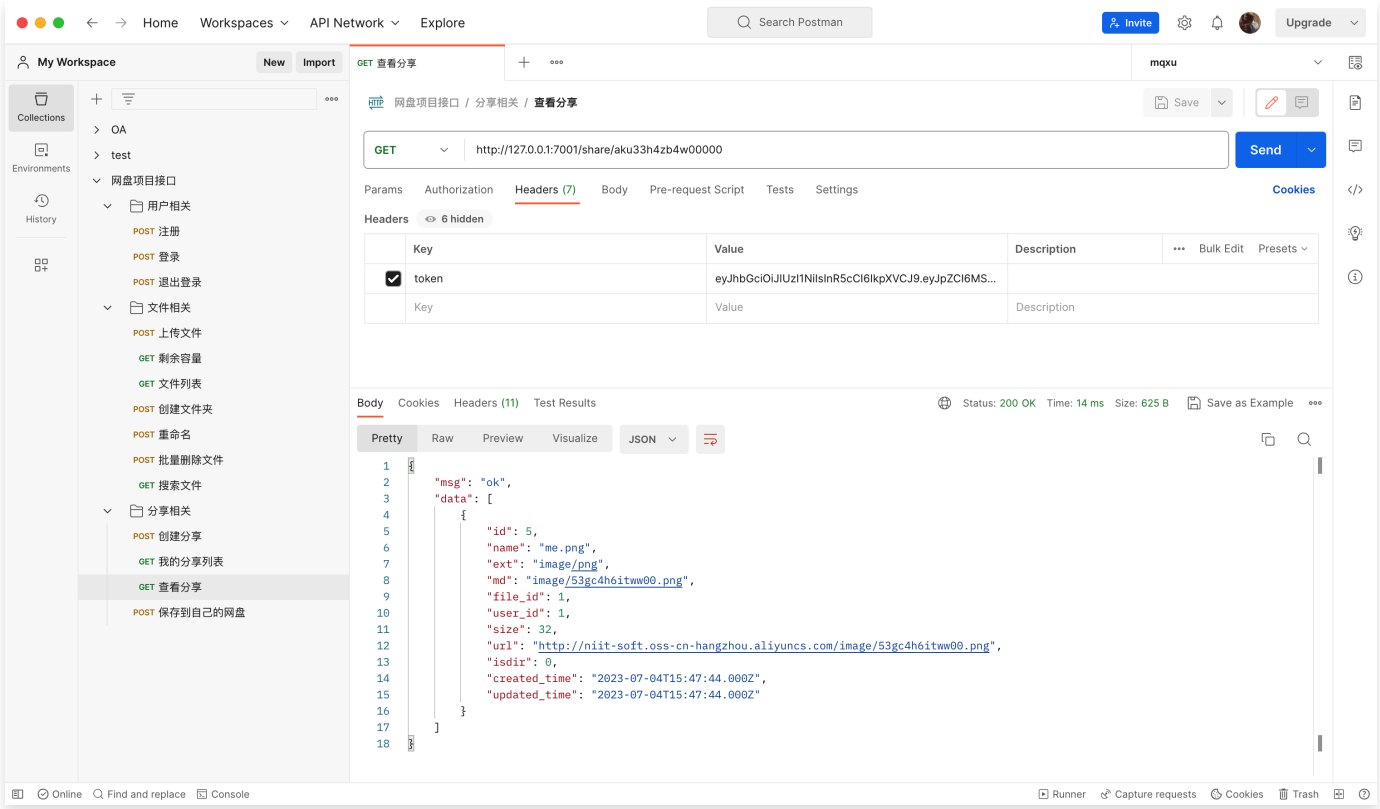
```
'use strict'

const Service = require('egg').Service

class ShareService extends Service {
  async isExist(sharedurl, options = {}) {
    let s = await this.app.model.Share.findOne({
      where: {
        sharedurl,
        iscancel: 0,
      },
      ...options,
    })

    if (!s) {
      return this.ctx.throw(404, '该分享已失效')
    }

    return s
  }
}

module.exports = ShareService
```

- 路由：app/router.js

JavaScript | 复制代码

```
// 查看分享
router.get('/share/:sharedurl', controller.share.read);
```

- 测试，带上之前创建成功的分享链接作为路径参数




```
commit 5cb75d55088d95eaeadb8925d010ffd3d8498f98 (HEAD -> main)
Author: mqxu <moqi1977@gmail.com>
Date:   Tue Jul 4 23:56:43 2023 +0800

    查看分享
```

保存到自己到网盘

控制器：app/controller/share.js

JavaScript | 复制代码

```
// 保存到自己的网盘
 async saveToSelf() {
   const { ctx, app, service } = this;
   let current_user_id = ctx.authUser.id;

   ctx.validate({
     dir_id: {
       type: "int",
       required: true,
       desc: "目录ID",
     },
     sharedurl: {
       type: "string",
       required: true,
       desc: "分享标识",
     },
   });

   let { dir_id, sharedurl } = ctx.request.body;

   // 分享是否存在
   let s = await service.share.isExist(sharedurl, {
     include: [
       {
         model: app.model.File,
       },
     ],
   });
   if (s.user_id === current_user_id) {
     return ctx.apiSuccess("本人分享，无需保存");
   }

   // 文件是否存在
   if (dir_id > 0) {
     await service.file.isDirExist(dir_id);
   }

   // 查询该分享目录下的所有数据
   let getAllFile = async (obj, dirId) => {
     let data = {
       name: obj.name,
       ext: obj.ext,
       md: obj.md,
       file_id: dirId,
       user_id: current_user_id,
```

```
          size: obj.size,
          isdir: obj.isdir,
          url: obj.url,
        };
        // 判断当前用户剩余空间
        if (ctx.authUser.total_size - ctx.authUser.used_size < data.size) {
          return ctx.throw(400, "你的可用内存不足");
        }

        // 直接创建
        let o = await app.model.File.create(data);

        // 更新user表的使用内存
        ctx.authUser.used_size = ctx.authUser.used_size + parseInt(data.size
    );
        await ctx.authUser.save();

        // 目录
        if (obj.isdir) {
          // 继续查询下面其他的数据
          let rows = await app.model.File.findAll({
            where: {
              user_id: obj.user_id,
              file_id: obj.id,
            },
          });
          rows.forEach((item) => {
            getAllFile(item, o.id);
          });

          return;
        }
      };

      await getAllFile(s.file, dir_id);

      ctx.apiSuccess("ok");
    }
```

- 路由：app/router.js

JavaScript | 复制代码

```
// 保存到自己网盘
router.post('/share/save_to_self', controller.share.saveToSelf);
```

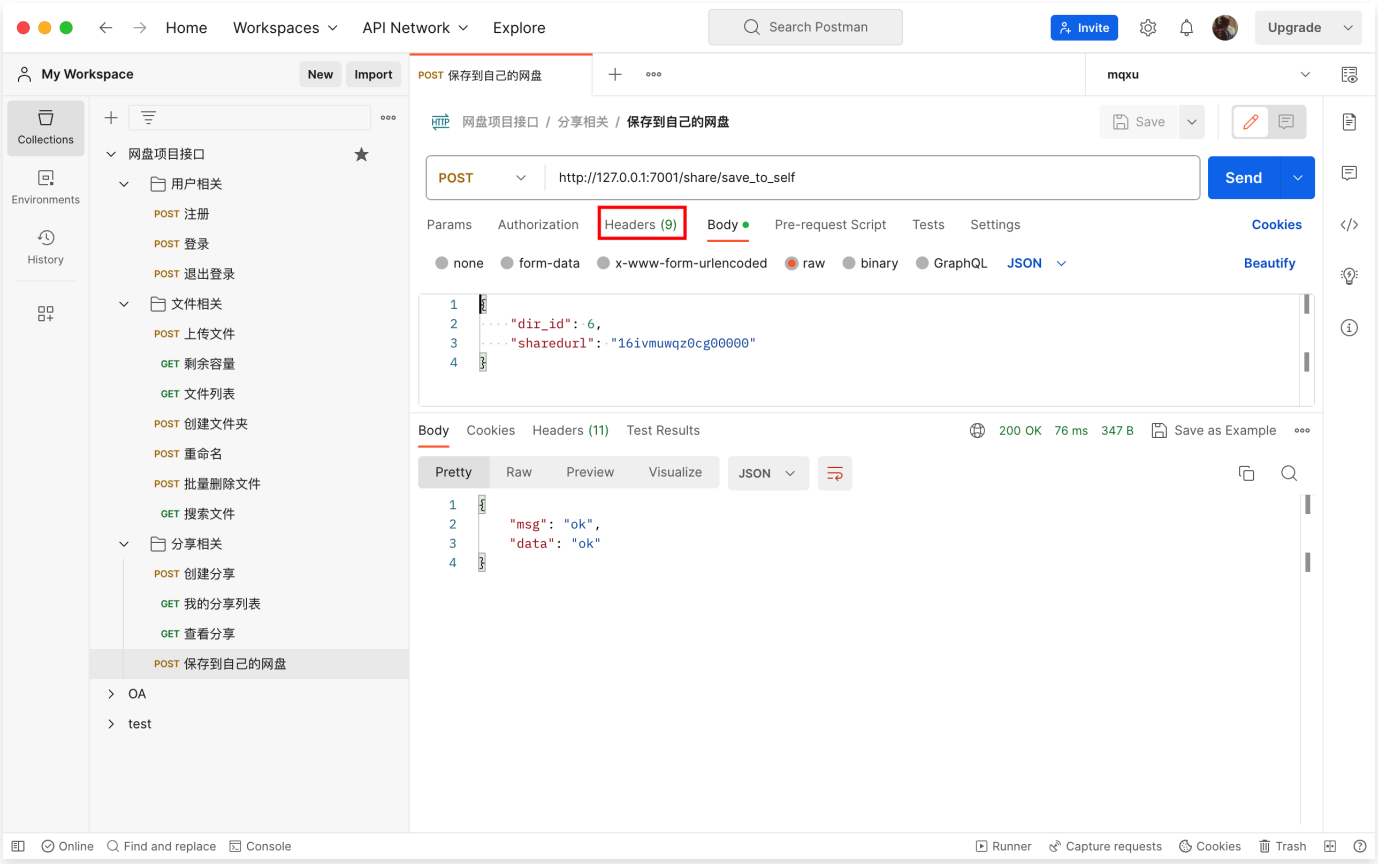
测试

注册一个用户，登录，后续操作都带上这个用户的 token

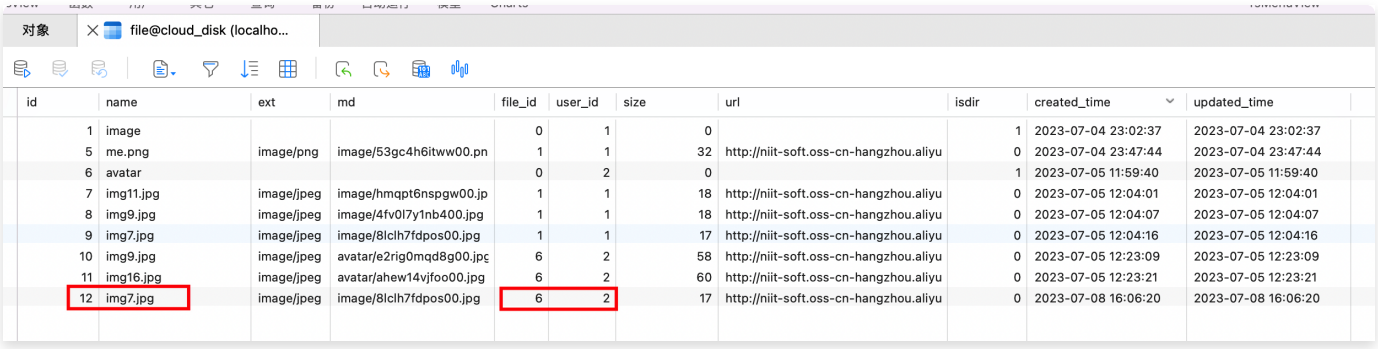
新建一个目录

用这个 2 号用户发起“保存到自己网盘“的请求，需要传递两个参数

- 保存到自己哪个目录
- 分享的 url



数据库



id	name	ext	md	file_id	user_id	size	url	isdir	created_time	updated_time
1	image			0	1	0		1	2023-07-04 23:02:37	2023-07-04 23:02:37
5	me.png	image/png	image/53gc4h6itww00.pn	1	1	32	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-04 23:47:44	2023-07-04 23:47:44
6	avatar			0	2	0		1	2023-07-05 11:59:40	2023-07-05 11:59:40
7	img11.jpg	image/jpeg	image/hmqpt6nspgw00.jp	1	1	18	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-05 12:04:01	2023-07-05 12:04:01
8	img9.jpg	image/jpeg	image/4fv0l7y1nb400.jpg	1	1	18	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-05 12:04:07	2023-07-05 12:04:07
9	img7.jpg	image/jpeg	image/8lclh7fdpos00.jpg	1	1	17	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-05 12:04:16	2023-07-05 12:04:16
10	img9.jpg	image/jpeg	avatar/e2rig0mqd8g00.jpg	6	2	58	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-05 12:23:09	2023-07-05 12:23:09
11	img16.jpg	image/jpeg	avatar/ahew14vjfoo00.jpg	6	2	60	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-05 12:23:21	2023-07-05 12:23:21
12	img7.jpg	image/jpeg	image/8lclh7fdpos00.jpg	6	2	17	http://niit-soft.oss-cn-hangzhou.aliyu	0	2023-07-08 16:06:20	2023-07-08 16:06:20

```
commit e19b7fc22c5e936f7056a079b434f9fabfb15a9c (HEAD -> main)
Author: mqxu <moqi1977@gmail.com>
Date:   Sat Jul 8 16:08:54 2023 +0800

    保存到自己到网盘
```