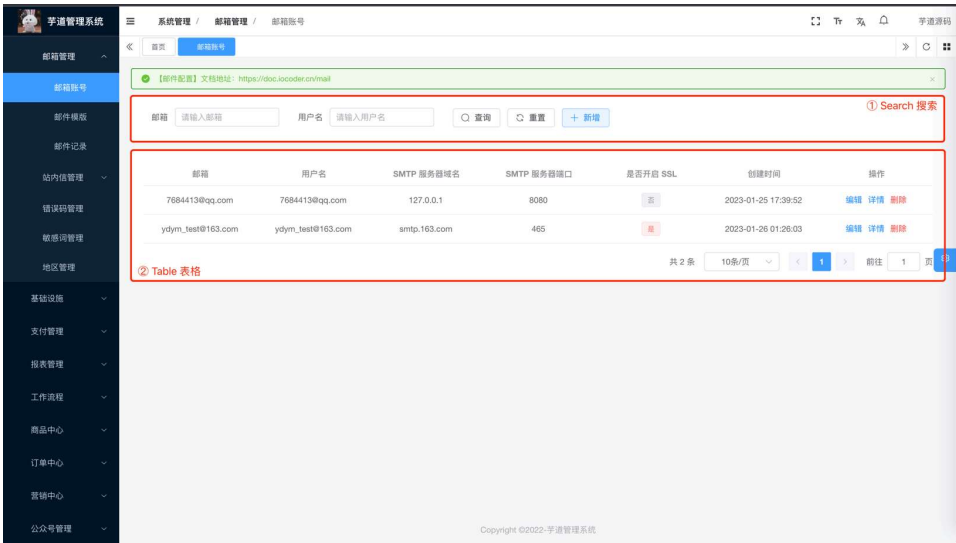
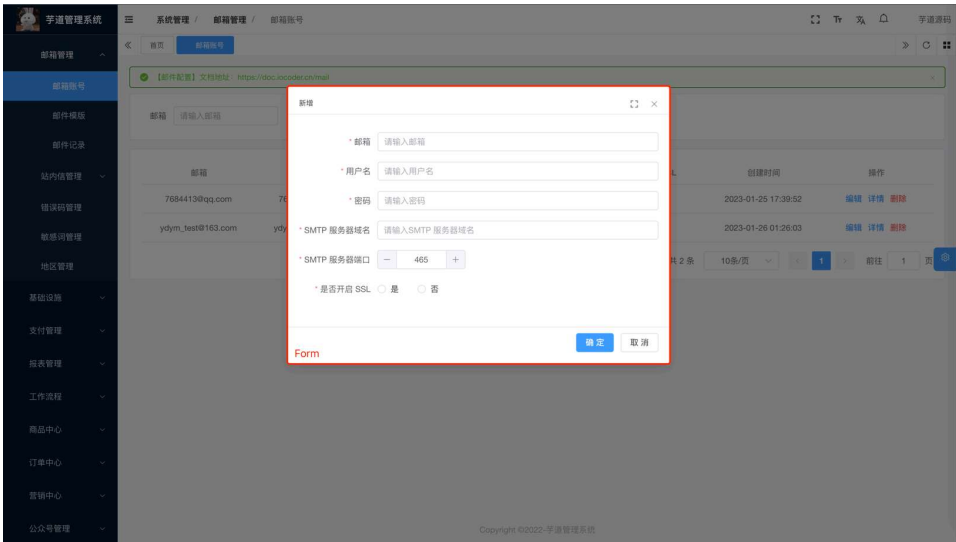
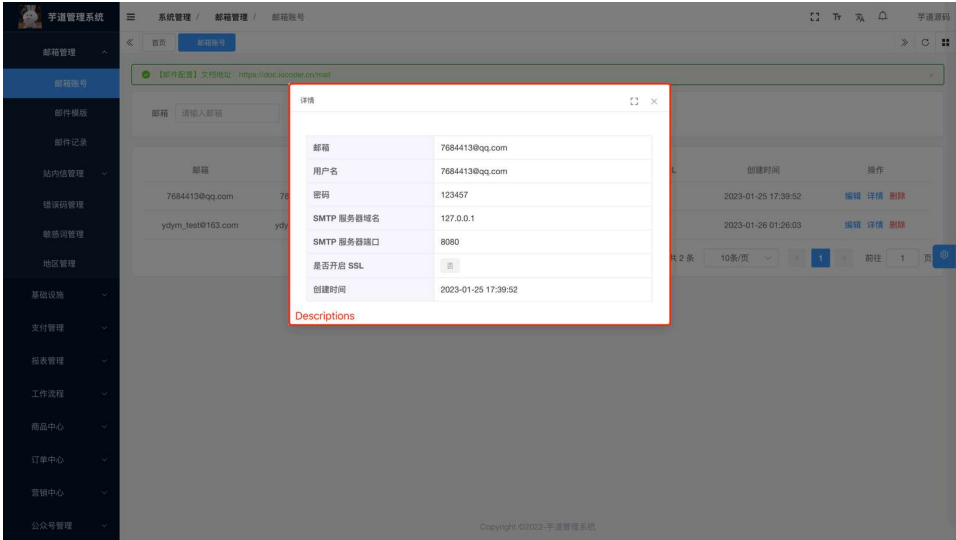


 CRUD 组件

管理后台的功能，一般就是 CRUD 增删改查，可以拆分 3 个部分：“列表”、“新增/修改”、“详情”，如下图所示：

部分	组件	示例
列表	Search + Table	
新增 / 修改	Form	

部分	组件	示例
详情	Descriptions	

1. 基础组件

涉及到 4 个前端基础组件，如下所示：

组件	文档
Search	查询组件
Table	表格组件
Form	表单组件
Descriptions	描述组件

2. CRUD 组件

由于以上 4 个组件都需要 Schema 或者 columns 的字段，如果每个组件都写一遍的话，会造成大量重复代码，所以提供 useCrudSchemas 来进行统一的数据生成。

- ① useCrudSchemas：位于 [src/hooks/web/useCrudSchemas.ts](#) 内
- ② useCrudSchemas 可以理解成一个 JSON 配置，示例如下：

► useCrudSchemas 示例

- ③ 字段的详细说明，可见 [useCrudSchemas 文档](#)。

3. 实战案例

项目的 [系统管理 -> 邮箱管理] 相关的功能，都使用 CRUD 实现，你可以自己去学习。

功能	代码
邮箱账号	src/views/system/mail/account
邮箱模版	src/views/system/mail/template
邮箱记录	src/views/system/mail/log

4. 常见问题

4.1 如何隐藏某个字段？

如 `formSchema` 不需要 `field` 为 `createTime` 的字段，可以使用 `form: { show: false }` 或 `isForm: false` 进行过滤，其他组件同理。

```
22 // CrudSchema: https://kailong110120130.gitee.io/vue-element-plus-admin-doc/hooks/useCrudSchemas.html
23 const crudSchemas = reactive<CrudSchema[]>([
24   {
25     label: '邮箱',
26     field: 'mail',
27     isSearch: true 搜索：默认隐藏，需要 true 显示
28   },
29   {label: '用户名'...},
34   {
35     label: '密码',
36     field: 'password',
37     isTable: false 表格：默认显示，需要 false 隐藏
38   },
39   {label: 'SMTP 服务器域名'...},
43   {label: 'SMTP 服务器端口'...},
51   {label: '是否开启 SSL'...},
68   {label: '创建时间'...},
69   {
70     label: '操作',
71     field: 'action',
72     isForm: false, 表单：默认显示，需要 false 隐藏
73     isDetail: false 详情：默认显示，需要 false 隐藏
74   }
75 ])
```

4.2 如何使用数据字典？

设置 `dictType` 字典的类型，和 `dictClass` 字典的数据类型。

```
51 {
52   label: '是否开启 SSL',
53   field: 'sslEnable',
54   dictType: DICT_TYPE.INFRA_BOOLEAN_STRING,
55   dictClass: 'boolean',
56   form: {
57     component: 'Radio'
58   }
59 }
```

4.3 如何使用 API 获取数据？

使用 `api` 来获取接口数据，需要主动 `return` 数据。

```
6 // 邮箱账号的列表
7 const accountList = await MailAccountApi.getSimpleMailAccountList() 1. API 加载数据
8
9 // 表单校验
10 > export const rules = reactive({...})
11
12 // CrudSchema: https://kailong110120130.gitee.io/vue-element-plus-admin-doc/hooks/useCrudSchemas.html
13 const crudSchemas = reactive<CrudSchema[]>([
14   {label: '模板编码'...},
15   {label: '模板名称'...},
16   {label: '模板标题'...},
17   {label: '模板内容'...},
18   {
19     label: '邮箱账号',
20     field: 'accountId',
21     width: '200px',
22     formatter: (_, Recordable, __: TableColumn, cellValue: number) => { 2.1 列表的格式化
23       return accountList.find((account) => account.id === cellValue)?.mail
24     },
25     search: {
26       show: true,
27       component: 'Select',
28       api: () => accountList, 2.2 搜索的使用
29       componentProps: {
30         optionsAlias: {
31           labelField: 'mail',
32           valueField: 'id'
33         }
34       }
35     },
36     form: {
37       component: 'Select',
38       api: () => accountList, 2.3 表单的使用
39       componentProps: {
40         optionsAlias: {
41           labelField: 'mail',
42           valueField: 'id'
43         }
44       }
45     }
46   }
47 ],
48 },
49 }
```

4.4 如何结合 Slot 自定义？

如果想要自定义，可以结合 Slot 来实现。具体有哪些 Slot，阅读对应基础组件的文档。

```
MailLogDetail.vue
1 <template>
2   <Dialog title="详情" v-model="dialogVisible" :scroll="true" :max-height="500">
3     <Descriptions :schema="allSchemas.detailSchema" :data="detailData">
4       <!-- 展示 HTML 内容 -->
5       <template #templateContent="{ row }">
6         <div v-html="row.templateContent"></div>
7       </template>
8     </Descriptions>
9   </Dialog>
10 </template>
```

[← 配置读取](#)

[国际化→](#)



Theme by **Vdoing** | Copyright © 2019-2024 芋道源码 | MIT License