1. Model: 其下所有函数都是对具体数据的操作。
2. CRUDGloalSetting：放到misc中，因为其下函数是通用操作而不是具体数据操作。

Async/await

1. 异步函数最终必须返回一个promise：

1.1 如果在async中调用其他函数，且要根据函数的结果(rc>0)，判断是否要返回，则必须加上await，以便rc>0时返回一个promise。

如下例子，sanityInput不是异步函数，但是为了在async中返回一个promise是，还是要使用await。

**let** sanitizedInputValue=**await *unifiedHelper***.sanityInput(req.**body**.**values**,***inputRule***[eCurrentColl],**false**,***maxFieldNum***[eCurrentColl])**if**(sanitizedInputValue.**rc**>0){**return** res.**json**(***unifiedHelper***.returnResult(sanitizedInputValue))  
}

1.2 如果最终返回的是一个函数，也需要加await

如下：returnResult是同步函数，只是更改数据的结构，但是因为是最终返回的值，也要变成promise

**return await** res.**json**(***unifiedHelper***.returnResult(populateResult))

1. mongoose的操作返回promise，即使使用了callback。为了nodejs不报错误（未处理的promise error），使用async/await的方式调用mongooose的操作，并加上catch来捕获错误并**产生对应的error code（通过Promise.reject返回）**。

2. 中间层调用的时候，如果是同步函数，使用await调用，则判断rc后，需要通过Promise.reject显示返回错误，如果是异步函数通过await，不用做任何操作，await会隐式直接返回reject。

3。 顶层函数通过then.catch处理最终的结果

**Null的处理**

Client端，可以通过设置field的value为null告知server端，字段需要删除。

如果在create的时候，有field为null，server端会

1. 检查对应的field rule，是否require为true

2. 如果通过validate（require=false），那么会通过validateFunc.constructCreateCriteria将对应的null field删除，确保无多余的字段传入

综上，**client可以使用同一的构建语句，构建带null的字段**