**布局：**

创建只需一个链接或者一个按钮。

显示需要一个页面，类似文档。页面上方为举报正文，下面为此举报的处理过程（回复）

功能：

对文档或者发言进行举报。

**Db:**

使用5个coll，采用独立表而不是内嵌，传统思维，比较好处理

**Impeach**：存储举报信息，

**impeachDealer**: 举报信息处理人分配过程（单独表，记录处理人分配过程）

**impeachImages**：储举报信息中的图片，

**impeachAttachments**：存储举报信息中的图片，

**impeachComments**：一个用来存储举报的对话

**impeachResult**：存储举报的处理结果

**PS：impeachImages/ impeachAttachments同时为impeach/ impeachComments服务，所以不提供冗余的外键，指明imgae/attachment对应哪个impeach/comment了。**

**Impeach**:{

Title: string required:2~49

Content;string required; 1~1999 举报内容

impeachImagesId:[ objectId] optional 举报截图

impeachAttachmentsId: [objectId] optional 举报附件

impeachCommentsId: [objectId] optional 举报后的对话

impeachType: **String**(**enum**) required; enum 举报对象的类型（文档还是评论：article/comment）

impeachedArticleId: article\_objectId optional 如果举报文档，填入文档Id

impeachedCommentId: article\_objectId optional 如果举报评论，填入评论Id

impeachedUserId: user\_objectId required 被举报的文档/评论的发布者（以便页面显示给他/她看）。冗余字段，可以skip从impeachedArticleId或者impeachedCommentId推导的步骤

creatorId:objectId required; 举报人

dealerId: objectId optional 处理人 ref: adminUser

impeachStatus：**enum**:**String** required;enum 新建/提交/接受/分配/处理中/驳回/处理完

cDate: date required;

}

Impeach\_dealer: {

impeachId:object\_id

assignerId:

dealerId:

cDate

}

**Impeach\_Image**:{

name:string required;4~255 举报插入的图片

hahsName:string required; 40+4 图片的hash名

authorId: objectId required; 图片插入人（冗余字段，由server决定：发起举报的人）

size: int required

pathId: objectId required 图片存储的位置

cDate: date required; 创建日期

~~uDate/dDate~~

}

**Impeach\_Attachment**:{

Name: string required; 4~255

HashName: string required; 40~44

AuthorId: objectId required;

size: int required

pathId: objectId required 图片存储的位置

cDate: date required; 创建日期

~~uDate/dDate~~

}

**Impeach\_Comment**:{

ImpeachId: objectId required 对话所属的举报

Content:string, required;15～140 对话内容

AuthorId:objectId required 对话发起者（举报人/处理人）

impeachImagesId:[ objectId] optional 举报截图

impeachAttachmentsId: [objectId] optional 举报附件

cDate required

~~uDate/dDate~~

}